



D3.4

Final Outcomes of New Learning Paradigms and Distributed AI

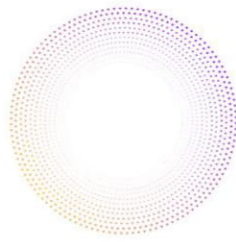
Project Title	AI4Media - A European Excellence Centre for Media, Society and Democracy
Contract No.	951911
Instrument	Research and Innovation Action
Thematic Priority	H2020-EU.2.1.1. - INDUSTRIAL LEADERSHIP - Leadership in enabling and industrial technologies - Information and Communication Technologies (ICT) / ICT-48-2020 - Towards a vibrant European network of AI excellence centres
Start of Project	1 September 2020
Duration	48 months



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 951911

info@ai4media.eu

www.ai4media.eu



Deliverable title	Final Outcomes of New Learning Paradigms and Distributed AI
Deliverable number	D3.4
Deliverable version	1.0
Previous version(s)	-
Contractual date of delivery	31/08/2024
Actual date of delivery	12/09/2024
Deliverable filename	D3.4_AI4Media.pdf
Nature of deliverable	Report
Dissemination level	Public
Number of pages	214
Work Package	WP3
Task(s)	T3.1, T3.2, T3.3, T3.4, T3.5, T3.6, T3.7, T3.8
Partner responsible	CEA
Author(s)	Adrian Popescu (CEA), Nicu Sebe (UNITN), Marco Formentini (UNITN), Hannes Fassold (JR), Matthew Barthet (UM), Antonios Liapis (UM), Niccolò Biondi (UNIFI), Federico Pernici (UNIFI), John Violos (CERTH), Symeon Papadopoulos (CERTH), Ioannis Sarridis (CERTH), Emmanouil Krasanakis (CERTH), Nikolaos Giatsoglou (CERTH), Christoforos Papastergiou (CERTH), Vasileios Mezaris (CERTH), Nikolaos Kaparinos (CERTH), Ioannis Patras (QMUL), Ioannis Maniadis Metaxas (QMUL), Christos Tzelepis (QMUL), Ioannis Pitas (AUTH), Ioanna Valsamara (AUTH), Anestis Kaimakamidis (AUTH), Artur Garcia (BSC), Fabrizio Sebastiani (CNR), Alejandro Moreo (CNR), Andrea Esuli (CNR), Lorenzo Volpi (CNR), Andrea Pedrotti (CNR)
Editor	Adrian Popescu (CEA)
Project Officer	Evangelia Markidou

Abstract	This document presents the final outcomes of the research on new learning paradigms and distributed AI, reporting the progress in tasks T3.1 to T3.8 during the period M37-M48. Specifically, this document builds upon and presents the updates of the work presented in previous deliverables D3.1, D3.2, and D3.3. For each task we present the technological and research advances, as well as relevant publications and published software, repositories, or other resources. Finally, we discuss the limitations of current technologies and our plans for further development and research.
Keywords	Artificial Intelligence, media, machine learning, deep learning, lifelong learning, online learning, manifold learning, disentangled feature representation, transfer learning, domain adaptation, deep quality diversity, learning to count, quantum assisted learning

Copyright

© Copyright 2024 AI4Media Consortium

This document may not be copied, reproduced, or modified in whole or in part for any purpose without written permission from the AI4Media Consortium. In addition to such written permission to copy, reproduce, or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced. All rights reserved.



Contributors

NAME	ORGANIZATION
Adrian Popescu	CEA
Nicu Sebe	UNITN
Marco Formentini	UNITN
Hannes Fassold	JR
Matthew Barthet	UM
Antonios Liapis	UM
Niccolò Biondi	UNIFI
Federico Pernici	UNIFI
John Violos	CERTH
Symeon Papadopoulos	CERTH
Ioannis Sarridis	CERTH
Emmanouil Krasanakis	CERTH
Nikolaos Giatsoglou	CERTH
Christoforos Papastergiopoulos	CERTH
Vasileios Mezaris	CERTH
Nikolaos Kaparinos	CERTH
Ioannis Patras	QMUL
Ioannis Maniadis Metaxas	QMUL
Christos Tzelepis	QMUL
Ioannis Pitas	AUTH
Ioanna Valsamara	AUTH
Anestis Kaimakamidis	AUTH
Artur Garcia	BSC
Fabrizio Sebastiani	CNR
Alejandro Moreo	CNR
Andrea Esuli	CNR
Lorenzo Volpi	CNR
Andrea Pedrotti	CNR

Peer Reviews

NAME	ORGANIZATION
Ioannis Patras	QMUL
Matthew Barthet	UM





Revision History

Version	Date	Reviewer	Modifications
0.1	10.07.2024	Adrian Popescu	First draft with contributions from all partners
0.2	25.07.2024	Adrian Popescu	Second draft with contributions from all partners
0.3	30.07.2024	Adrian Popescu	Draft sent to internal reviewers
0.4	01.09.2024	Adrian Popescu	Updated version based on internal reviews
0.5	12.09.2024	Adrian Popescu	Final version
1.0	12.09.2024	Filareti Tsalakanidou	Final version ready for submission

The information and views set out in this report are those of the author(s) and do not necessarily reflect the official opinion of the European Union. Neither the European Union institutions and bodies nor any person acting on their behalf.





Table of Abbreviations and Acronyms

Abbreviation	Meaning
3D	Tri-dimensional
ACC	Adjusted Classify and Count
AE	Auto-encoder
AI	Artificial Intelligence
ANN	Artificial Neural Network
AP	Average Precision
API	Application Programming Interface
APP	Artificial Prevalence Protocol
AUC	Area Under Curve
BA	Balanced Accuracy
BA^2	Budget-Aware Adapters
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-Term Memory
BR	Binary Relevance
CAM	Class Activation Maps
CAP	Classifier Accuracy Prediction
CBIR	Content-Based Image Retrieval
CC	Classify and Count
CMA-ME	Covariance Matrix Adaptation MAP-Elites
CIL	Class-incremental Learning
CL	Curriculum Learning
class-iNCD	class-incremental NCD
CLS	Classify Token
CNN	Convolutional Neural Network
CoReS	Compatible Representations via Stationarity
CPSS	Cross-Patch Style Swap
CPU	Central Processing Unit
CSD	Contrastive Supervised Distillation
CUDA	Compute Unified Device Architecture
CV	Computer Vision
DeiT	Data-efficient Image Transformer
DA	Domain Adaptation
DAS	Design Alternatives Sampling
DCCL	Dynamic Conceptual Contrastive Learning
DCG	Dynamic Conception Generation
DCL	Dual-level Contrastive Learning
DCNN	Deep Convolutional Neural Network





Abbreviation	Meaning
deepQD	deep-learning-based QD
DeLeNoX	Deep Learning Novelty eXplorer
DG	Domain Generalization
DIGA	Dynamically Instance-Guided Adaptation
DNN	Deep Neural Network
DoF	Depth of Field
EC	Evolutionary Computation
EU	European Union
EER	Equal Error Rate
EFCIL	Exemplar-Free Class-Incremental Learning
F1	F1-score
FAR	False Acceptance Rate
FE	Feature Extraction
FER	Facial Expression Recognition
FFN	Feed-Forward Networks
FFT	Fast Fourier Transform
FLOP	Floating-point Operations per Second
FPS	Frames per Second
FT	Fine Tuning
GCD	Generalized Category Discovery
GCN	Graph Convolutional Network
GenAI	Generative AI
GNN	Graph Neural Network
GPT	Generative Pre-Trained
GPU	Graphics Processing Unit
HED	Holistically-Nested Edge Detection
HJ	Hamilton–Jacobi equation
IBU	Iterative Bayesian Unfolding
IEC	Interactive Evolutionary Computation
IoU	Intersection over Union
IS	Inception Score
JGNN	A Java library for creating and training lightweight Graph Neural Networks
JS	Jensen–Shannon divergence
KD	Knowledge Distillation
KDE	Kernel Density Estimation
KL	Kullback-Leibler
KSA	Knowledge Self-Assessment
LDA	Latent Dirichlet Allocation
LLE	Local Linear Embedding





Abbreviation	Meaning
LLM	Large Language Model
LMM	Large Multimedia Model
LQ	Learning to Quantify
LSTM	Long-Short Term Memory
mAP	mean Average Precision
MDL	Multi-Domain Learning
MELiTA	Map-Elites with Transverse Assesment
MI	Mutual Information
mIoU	mean Intersection-over-Union
MLM	Masked Language Modeling
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
MTCNN	Multi-Task Cascaded Convolutional Neural Network
NAS	Neural Architecture Search
NCD	Novel Category Discovery
NEAT	NeuroEvolution of Augmenting Topologies
NLP	Natural Language Processing
NPP	Natural Prevalence Protocol
NQ	Network Quantification
NSGA-II	Non-dominated Sorting Genetic Algorithm II
NSLC	Novelty Search with Local Competition
OOD	Out-of-Distribution
OIDD	Out Of Distribution Detection
OQ	Ordinal Quantification
OT	Optimal Transport
P@k	Precision@k
PACC	Probabilistic Adjusted Classify and Count
PCC	Probabilistic Classify and Count
PCG	Procedural Content Generation
PDE	Partial Differential Equation
PIL	Python Imaging Library
PINN	Physical Informed Neural Network
PL	Preference Learning
PPS	Prior Probability Shift
PT	Photometric Transformation
QD	Quality Diversity
QoI	Quality of Inference
RGB	Red Green Blue
RL	Reinforcement Learning
ROC	Receiver Operating Characteristic





Abbreviation	Meaning
RUN	Regularized Unfolding
S2D-Dec	Sparse2Dense mesh Decoder
SF-OCDA	Source-Free Open Compound Domain Adaptation
SFDA	Source-Free Domain Adaptation
SGD	Stochastic Gradient Descent
SLD	Saerens-Latinne-Decaestecker
SOTA	State of the Art
SRVF	Square Root Velocity Function
SSL	Self-supervised Learning
SVM	Support Vector Machine
t-SNE	Distributed Stochastic Neighbor Embedding
TTDA	Test-time Domain Adaptation
TTDA-Seg	Test-Time Domain Adaptation in Semantic Segmentation
TPRD	True Positive Rate Disparity
TTS	Text-to-Speech
UB	Upper Bound
UCB	Upper Confidence Bound
UC	Use Case
UC-ME	User Controller MAP-Elites
UDA	Unsupervised Domain Adaptation
ULA	Universal Local Attractor
UMAP	Uniform Manifold Approximation and Projection
URL	Uniform Resource Locator
USC	User Selection Criterion
VAE	Variational Autoencoder
VidLMs	Video Language Models
ViT	Visual Transformer
VGG	Visual Geometry Group
WP	Work Package
XAI	eXplainable AI
XNQ	eXtreme Network Quantifier





Contents

1	Executive Summary	22
2	Introduction	25
3	Concise descriptions of the presented works	28
3.1	Lifelong and on-line learning (Task 3.1)	28
3.1.1	Introduction	28
3.1.2	Overview	28
3.2	Manifold learning and disentangled feature representation (Task 3.2)	29
3.2.1	Introduction	29
3.2.2	Overview	29
3.3	Transfer learning (Task 3.3)	29
3.3.1	Introduction	29
3.3.2	Overview	30
3.4	Neural Architecture Search (Task 3.4)	30
3.4.1	Introduction	30
3.4.2	Overview	30
3.5	AI at the Edge, decentralised and distributed learning (Task 3.5)	31
3.5.1	Introduction	31
3.5.2	Overview	31
3.6	Deep quality diversity (Task 3.6)	32
3.6.1	Introduction	32
3.6.2	Overview	33
3.7	Learning to count (Task 3.7)	34
3.7.1	Introduction	34
3.7.2	Overview	34
3.8	Quantum assisted reinforcement learning (Task 3.8)	35
3.8.1	Introduction	35
3.8.2	Overview	36
4	Lifelong and on-line learning (Task 3.1)	37
4.1	Dynamic Conceptual Contrastive Learning for Generalized Category Discovery	37
4.1.1	Introduction and methodology	37
4.1.2	Experimental results	39
4.1.3	Conclusions	40
4.1.4	Relevant publications	41
4.1.5	Relevant software/datasets/other outcomes	41
4.1.6	Relevance to AI4Media use cases and media industry applications	41
4.2	A reality check on pretraining for exemplar-free class-incremental learning	41
4.2.1	Introduction and methodology	41
4.2.2	Comparison to State-of-the-art Methods	42
4.2.3	Conclusion	44
4.2.4	Relevant publications	45
4.2.5	Relevance to AI4media use cases and media industry applications	45
4.3	Stationary Representations: Optimally Approximating Compatibility and Implications for Improved Model Replacements	45
4.3.1	Introduction and methodology	45





4.3.2	Experimental results	47
4.3.3	Conclusion	50
4.3.4	Relevant publications	50
4.3.5	Relevant software and/or external resources	50
4.3.6	Relevance to AI4media use cases and media industry applications	50
4.4	Collaborative Knowledge Distillation via a Learning-by-Education Node Community	50
4.4.1	Introduction and methodology	50
4.4.2	Experimental results	55
4.4.3	Conclusion	56
4.4.4	Relevant publications	56
4.4.5	Relevance to AI4media use cases and media industry applications	56
4.5	Efficient DNN knowledge assessment in a multi-agent Teacher-Student DNN environment	57
4.5.1	Introduction and methodology	57
4.5.2	Experimental results	59
4.5.3	Conclusion	61
4.5.4	Relevant publications	61
4.5.5	Relevance to AI4Media use cases and media industry applications	61
5	Manifold learning and disentangled feature representation (Task 3.2)	63
5.1	Flow Factorized Representation Learning	63
5.1.1	Introduction and methodology	63
5.1.2	Experiments	64
5.1.3	Conclusions and Limitations	66
5.1.4	Relevant publications	66
5.1.5	Relevant software/datasets/other outcomes	66
5.1.6	Relevance to AI4Media use cases and media industry applications	67
5.2	Parts of Speech–Grounded Subspaces in Vision-Language Models	67
5.2.1	Introduction and methodology	67
5.2.2	Experimental results	68
5.2.3	Conclusions	69
5.2.4	Relevant publications	70
5.2.5	Relevant software/datasets/other outcomes	70
5.2.6	Relevance to AI4Media use cases and media industry applications	70
5.3	Improving Fairness using Vision-Language Driven Image Augmentation	70
5.3.1	Introduction and methodology	70
5.3.2	Experimental results	72
5.3.3	Conclusions	73
5.3.4	Relevant publications	73
5.3.5	Relevant software/datasets/other outcomes	73
5.3.6	Relevance to AI4Media use cases and media industry applications	74
6	Transfer learning (Task 3.3)	75
6.1	Dynamically Instance-Guided Adaptation	75
6.1.1	Introduction and methodology	75
6.1.2	Experiments	77
6.1.3	Conclusion	78
6.1.4	Relevant publications	79





6.1.5	Relevant software/datasets/other outcomes	79
6.1.6	Relevance to AI4Media use cases and media industry applications	79
6.2	Assessing Fine-Grained Modality Alignment in Video-Language Models	79
6.2.1	Introduction and methodology	79
6.2.2	Experiments	83
6.2.3	Conclusion	84
6.2.4	Relevant Publications	84
6.2.5	Relevant software/datasets/other outcomes	85
6.2.6	Relevance to AI4Media use cases and media industry applications	85
7	Neural Architecture Search (Task 3.4)	86
7.1	Budget-Aware Pruning for Multi-Domain Learning	86
7.1.1	Introduction and methodology	86
7.1.2	Experiments	87
7.1.3	Conclusion	90
7.1.4	Relevant publications	90
7.1.5	Relevance to AI4media use cases and media industry applications	90
8	AI at the Edge, decentralised and distributed learning (Task 3.5)	91
8.1	Frameworks for graph analysis and learning at the edge	91
8.1.1	Introduction an methodology	91
8.1.2	Results	94
8.1.3	Conclusions	96
8.1.4	Relevant publications	96
8.1.5	Relevant software	96
8.1.6	Relevance to AI4Media use cases and media industry applications	97
8.2	GNN architectures trained at the edge	97
8.2.1	Introduction and methodology	97
8.2.2	Results	100
8.2.3	Conclusions	102
8.2.4	Relevant publications	102
8.2.5	Relevant software	102
8.2.6	Relevance to AI4media use cases and media industry applications	102
8.3	Genetic Algorithms For Federated Learning	102
8.3.1	Introduction	102
8.3.2	Methodology	103
8.3.3	Experiments	105
8.3.4	Conclusions	106
8.3.5	Relevant Publications	107
8.3.6	Relevance to AI4Media use cases and media industry applications	107
8.4	InDistill: Information flow-preserving knowledge distillation for model compression	107
8.4.1	Introduction	107
8.4.2	Methodology	108
8.4.3	Experimental setup	110
8.4.4	Results	110
8.4.5	Conclusions	112
8.4.6	Relevant publications	112
8.4.7	Relevant software	112
8.4.8	Relevance to AI4Media use cases and media industry applications	112





8.5	Towards Optimal Trade-offs in Knowledge Distillation for CNNs and Vision Transformers at the Edge	112
8.5.1	Introduction	112
8.5.2	Methodology	113
8.5.3	Experimental setup	114
8.5.4	Results	115
8.5.5	Conclusions	117
8.5.6	Relevant publications	117
8.5.7	Relevance to AI4Media use cases and media industry applications	117
8.6	FPGM pruning for lightweight mobile face detection models	117
8.6.1	Introduction and methodology	118
8.6.2	Algorithm	118
8.6.3	Experimental results	120
8.6.4	Conclusion	120
8.6.5	Relevant publications	120
8.6.6	Relevant software	120
8.6.7	Relevance to AI4Media use cases and media industry applications	122
8.7	Porting Large Language Models to Mobile Devices for Question Answering	122
8.7.1	Introduction	122
8.7.2	LLM framework for on-device inference	122
8.7.3	Model selection and prompt format	123
8.7.4	Experiments	123
8.7.5	Conclusion	123
8.7.6	Relevant publications	123
8.7.7	Relevance to AI4media use cases and media industry applications	124
8.8	AdaFamily: A family of adaptive gradient methods for training	124
8.8.1	Introduction	124
8.8.2	Algorithm	125
8.8.3	Experimental results	126
8.8.4	Conclusion	127
8.8.5	Relevant publications	128
8.8.6	Relevant software	128
8.8.7	Relevance to AI4Media use cases and media industry applications	128
9	Deep quality diversity (Task 3.6)	129
9.1	Multimodal Quality Diversity in Creative Domains	129
9.1.1	Introduction and methodology	129
9.1.2	Experimental results	131
9.1.3	Conclusion	131
9.1.4	Relevant publications	132
9.1.5	Relevance to AI4Media use cases and media industry applications	132
9.2	Large Language Models for Automated Game Generation	132
9.2.1	Introduction and methodology	132
9.2.2	Experimental results and discussion	134
9.2.3	Conclusion	134
9.2.4	Relevant publications	134
9.2.5	Relevance to AI4media use cases and media industry applications	134
9.3	Quality Diversity in Dynamic Environments	134
9.3.1	Introduction and methodology	134





9.3.2	Experimental results	135
9.3.3	Conclusion	136
9.3.4	Relevant publications	137
9.3.5	Relevant software/datasets/other outcomes	137
9.3.6	Relevance to AI4media use cases and media industry applications	137
9.4	Quality-Diversity Search for Constrained Structure Design	137
9.4.1	Introduction and methodology	137
9.4.2	Experimental results	138
9.4.3	Conclusion	139
9.4.4	Relevant publications	139
9.4.5	Relevance to AI4media use cases and media industry applications	139
10	Learning to count (Task 3.7)	140
10.1	Measuring fairness under unawareness of sensitive attributes: A quantification-based approach	140
10.1.1	Introduction	140
10.1.2	Methodology	141
10.1.3	Conclusion	143
10.1.4	Relevant publications	144
10.1.5	Relevant software/datasets/other outcomes	144
10.2	Binary quantification and dataset shift: An experimental investigation	144
10.2.1	Introduction	145
10.2.2	Methodology	145
10.2.3	Conclusion	147
10.2.4	Relevant publications	148
10.2.5	Relevant software/datasets/other outcomes	148
10.3	Kernel density estimation for multiclass quantification	148
10.3.1	Introduction	148
10.3.2	Methodology	149
10.3.3	Conclusion	150
10.3.4	Relevant publications	151
10.3.5	Relevant software/datasets/other outcomes	151
10.4	Quantification using permutation-invariant networks based on histograms	151
10.4.1	Introduction	151
10.4.2	Methodology	152
10.4.3	Conclusion	154
10.4.4	Relevant publications	154
10.4.5	Relevant software/datasets/other outcomes	154
10.5	Quantifying query fairness under unawareness	154
10.5.1	Introduction	155
10.5.2	Methodology	155
10.5.3	Conclusion	158
10.5.4	Relevant publications	158
10.5.5	Relevant software/datasets/other outcomes	158
10.5.6	Relevance to AI4Media use cases and media industry applications	158
10.6	Learning to quantify graph nodes	158
10.6.1	Introduction	158
10.6.2	Methodology	159
10.6.3	Conclusion	161





10.6.4	Relevant publications	161
10.6.5	Relevant software/datasets/other outcomes	162
10.6.6	Relevance to AI4Media use cases and media industry applications	162
10.7	Using quantification to predict classifier accuracy under prior probability shift	162
10.7.1	Introduction	162
10.7.2	Methodology	163
10.7.3	Conclusion	165
10.7.4	Relevant publications	165
10.7.5	Relevant software/datasets/other outcomes	166
10.7.6	Relevance to AI4media use cases and media industry applications	166
10.8	Other contributions related to Learning to Quantify (Task 3.7)	166
10.8.1	Summary	166
10.8.2	Relevant publications	167
10.8.3	Relevant software/datasets/other outcomes	167
11	Quantum Reinforcement Learning (Task 3.8)	168
11.1	Quantum circuit depth optimization using RL	168
11.1.1	Introduction and methodology	168
11.1.2	Methodology	168
11.1.3	Experimental results	169
11.1.4	Conclusion	169
11.1.5	Relevant publications	169
11.1.6	Relevance to AI4Media use cases and media industry applications	170
12	Ongoing Work and Conclusions	172
12.1	Ongoing work	172
12.1.1	Lifelong and on-line learning (Task 3.1)	172
12.1.2	Manifold learning and disentangled feature representation (Task 3.2)	172
12.1.3	Transfer learning (Task 3.3)	172
12.1.4	Neural Architecture Search (Task 3.4)	172
12.1.5	AI at the Edge, decentralised and distributed learning (Task 3.5)	173
12.1.6	Deep quality diversity (Task 3.6)	173
12.1.7	Learning to count (Task 3.7)	173
12.1.8	Quantum Reinforcement Learning (Task 3.8)	173
12.2	Conclusions	173





List of Tables

2	Results on generic image recognition datasets.	39
3	Results on fine-grained datasets.	40
4	Comparisons of LENC with competing CKD methods, for incoming data streams \mathcal{D}^s of sizes 1,000 and 5,000. The average test accuracy (%) and deviation of the student nodes is reported.	56
5	Out-of-Distribution (OOD) detection experimental results. The metric used to evaluate the performance of the OOD detector is the AUCROC. The D_{in} is the In-Distribution dataset, while D_{out} is the Out-of-Distribution dataset. The results presented are the mean values from ten independent experiments.	60
6	Minimum number of data required to ensure the reliability of the OOD detection module for each training dataset, D_{in} , and the percentage in relation to the training dataset size.	61
7	Equivariance error \mathcal{E}_k and log-likelihood $\log p(\mathbf{x}_t)$ on Shapes3D [1].	65
8	Results of the classification tasks with age as protected characteristic. We train the model with the original training data (baseline), with the original data plus synthetic (baseline-sampling), and with the proposed VLBC-. We compare with <i>weighting</i> [2], LfF [3], and CGL-FairHSIC [4].	73
9	Comparison with state-of-the-art methods in terms of mIoU. The best score for each column is highlighted . CS: CityScapes, BDD: BDD100K, MA: Mapillary, IDD: IDD, CC: Cross-City. *: Use an extra augmented sample during adaptation. Avg.: Mean of mIoUs over five target domains.	77
10	Overview of data and foiling methods used in each test in ViLMA.	80
11	Pairwise ranking accuracy (acc_r) performance of 12 Video-Language Models on the ViLMA benchmark on the proficiency (P), main (T), and combined (P+T) tasks. In the combined task P+T , a success in T only counts if P is also successful. The final column Avg. is the taskwise average of combined scores P+T among each task. Best (second-best) model per metric are marked in boldface (underlined).	83
12	Computational complexity, number of parameters, accuracy per domain, S , S_O and S_P scores on the Visual Domain Decathlon, best in bold, second best underlined	89
13	GNN training resources on the Cora dataset.	95
14	Average measure values for competing graph filters for community recommendation. Best method in bold.	95
15	Comparison of <i>pygrank</i> with alternatives for node ranking algorithms.	96
16	Comparing the accuracy of different types and training schemes of base algorithms and their combination with the diffusion of p2pGNN. Accuracy is computed after 1,000 time steps and averaged across 5 peer-to-peer simulation runs.	100
17	Average Nemenyi ranks of GNN architectures across 12 predictive tasks.	101
18	Genetic Algorithms' models Accuracy Results	105
19	Top-1 accuracy on CIFAR-100.	110
20	Classification evaluation on ImageNet. ResNet34 and ResNet18 are employed as teacher and student networks.	111
21	Metric learning and classification evaluation on ImageNet and CIFAR-100. ResNet50 as a teacher and ResNet18 as a student are selected for the ImageNet dataset, while ResNet32 \times 4 as a teacher and ResNet32 as a student are selected for the CIFAR-100 dataset.	111
22	Knowledge Distillation between ViTs & CNNs	115





23	Exploring Knowledge Distillation with Various Sizes of CNN Students & Image Resolutions	116
24	Enhancement in Accuracy with Fine-Tuning on CNNs & ViTs	117
25	Comparative MAP results on EXTD between the proposed pruning approach (FPGM-based) and our baseline.	120
26	Comparative MAP results on EResFD between the proposed pruning approach (FPGM-based) and our baseline.	121
27	Comparative MAP results on EResFD between universal FPGM pruning with FPGM pruning that uses pruning rates optimized through Bayesian optimization.	121
28	Results (measured as test error, in percent) on the <i>CIFAR-10</i> dataset for different models. The best and second-best result for each model is marked in orange and blue.	127
29	Results (measured as test error, in percent) on the <i>CIFAR-100</i> dataset for different models. The best and second-best result for each model is marked in orange and blue, respectively.	128
30	Main types of dataset shift discussed in the literature. For the type of dataset shift on the row, symbol “ \neq ” indicates that the distribution on the column is assumed to change across L and U , while symbol “ $=$ ” indicates that the distribution is assumed to remain invariant. The last column indicates the section of the present work where this type of shift is discussed in detail.	146





List of Figures

1	Diagram of the proposed Dynamic Conceptual Contrastive Learning (DCCL). Samples from the conceptions should be close to each other. For example, samples from the same classes (bus) at the class level, samples belonging to the transportation (bus and bicycle) at the super-class level, and samples from trains with different colors at the sub-class level. Our DCCL potentially learns the underlying conceptions in unlabeled data and produces more discriminative representations.	37
2	(a) Overview of our dynamic conceptual contrastive learning framework. We first extract features by ViT backbone network, then cluster the features to generate conceptional labels and initialize conception representations, and learn representations by joint instance-level and conception-level objectives. During the training process, the initialization of conception representations and dual-level representation learning are performed alternately, in which the conception buffer is updated every iteration to keep the consistency of the changing instance features and conceptional representations. (b) Illustration of the proposed conception consolidation. Without consolidating the relationships of conceptions by label information, Infomap tends to over-cluster data and thus provides the supervision that has a high risk to over-correct affinities between neighbor instances.	38
3	Visualization of features distributions of the unlabeled set of the Pet [5] dataset. (a)-(d) are the results generated from DINO [6], GCD [7], our DCCL without \mathcal{L}^D and full DCCL, in turn. (e) is a visualization of false samples that are easy to be incorrectly clustered.	40
4	Comparison of SL and tReX training strategies for four EFCIL algorithms (NCM, DSLDA, FeTrIL, FeCAM). Feature extractors are trained on the initial data subset of each dataset (Casia, Inat, Landmarks), i.e. with either 100 (b100t9) or 500 classes (b500t10).	43
5	Average incremental accuracy of FeCAM [8] for various feature extractors: Res-net50 (RN50) model trained using vanilla supervised learning (SL) or tReX on the initial classes (init) and on the initial subset of classes augmented by 1000 classes generated using Stable Diffusion v2.1. (init+gen), supervised RN50 trained on ImageNet-1k, RN50 trained with CLIP on LAION-400m, ViT-S trained with DINOv2 on LVD-142m, supervised ViT-S trained on ImageNet-21k with AugReg. The best results are highlighted in black. <i>Supervised learning on the initial data subset outperforms strong pretraining in four cases out of six.</i>	44
6	Average incremental accuracy as a function of the initial accuracy. Linear regressions fitted on the average incremental accuracies of four EFCIL algorithms obtained with models pretrained on large external datasets (full, black line) or the first subset of classes with or without synthetic augmentation (dashed, red line).	44
7	Improved Asynchronous Model Compatible Lifelong Learning Representation (IAM-CL ² R pronounced “ <i>I am clear</i> ”). In the process of lifelong learning, a model is sequentially fine-tuned and asynchronously replaced with improved third-party models that are pre-trained externally. Stationary representations ensure seamless retrieval services and better performance, without the need to reprocess gallery images. . .	46





8	Implementation details of fine-tuning and model replacement in IAM-CL ² R. In the third-party pre-trained model ϕ^* , class labels are assigned from left to right. Conversely, in the current fine-tuned model ϕ_t^* , class labels are assigned from right to left. This simple convention maintains distinct class labels for pre-training and fine-tuning, preventing any overlap between them. The d -Simplex fixed matrix \mathbf{W} can be seen as a common interface between the two learning processes.	47
9	Average multi-model Accuracy (AA_t) evaluated across 31 tasks using CIFAR100R/10, showing: (a) model replacements at tasks 11 and 21 (indicated by yellow markers); (b) no model replacement.	48
10	Plots of Average multi-model Accuracy (AA_t) for 31 tasks on CIFAR100R/10, showing the impact of model replacements with different network architectures at tasks 11 and 21.	49
11	LENC node architecture.	53
12	Integration of an Out-of-Distribution (OOD) Detection Module in each DNN agent to optimize evaluation in a multi-agent system. The selection of the Teacher agent is based on task-relevant In-Distribution (ID) data. The DNN agent possessing the most ID data is selected to function as a Teacher agent. We notate as $idx(\cdot)$ the function that returns the index of its argument and idx^* its output.	59
13	Relationship between AUCROC scores and sample size for the dataset Fashion MNIST [9] as D_{in} and the MNIST [10] as D_{out}	60
14	Illustration of our flow factorized representation learning: at each point in the latent space we have a distinct set of tangent directions ∇u^k which define different transformations we would like to model in the image space. For each path, the latent sample evolves to the target on the potential landscape following dynamic optimal transport.	64
15	Exemplary latent flow results on Shapes3D [1]. The transformations from top to bottom are Floor Hue, Wall Hue, Object Hue, and Scale, respectively. The images of the top row are from the supervised experiment, while the bottom row is based on the weakly-supervised experiment.	65
16	CLIP represents multiple visual modes of variation in an embedding (e.g. the ‘object’ and its ‘appearance’). The learnt PoS subspaces more reliably separate the constituent visual components.	67
17	Killing the CLIP representations’ component in the adjective subspace provides a way to block the imitation of artists’ styles in CLIP-based text-to-image models.	69
18	Projecting onto the orthogonal complement of custom visual theme subspaces erases <i>specific</i> appearances from CLIP-based text-to-image models’ images.	69
19	Overview of the proposed Vision-Language Bias Control (VLBC) method for controlling the bias in facial image datasets. Given a real training set \mathcal{X}_r and a downstream task, we find the under-represented protected characteristic (e.g., black in <i>skin colour</i>) by computing the sample statistics. Based on this, we select which images from a synthetic dataset \mathcal{X}_s (generated using the DiffAE [11] generator \mathcal{G} and pseudo-labelled by a pre-trained network on 41 attributes f [12,13]) to use for augmentation. Then, the selected images are manipulated by our augmentation module (ContraCLIP+DiffAE), pre-trained on text prompts defining the desired protected characteristic. In this example, we manipulate/augment the selected images based on <i>skin colour</i> . Note that the original labels of the augmented images (i.e., corresponding to the attribute class at hand) do not change. Finally, the augmented dataset \mathcal{X}_a is used along with the original real dataset \mathcal{X}_r for training downstream tasks.	71





20	Top: Illustration of test-time domain adaptive semantic segmentation (TTDA-Seg). Bottom: Comparison with different TTDA methods. The proposed DIGA is a holistic method that has the properties of effectiveness (distribution&semantic adaptation and avoid unstable training&error accumulation) and efficiency (backward-free).	75
21	Illustration of the implementation of our DIGA. Given a source model, our DIGA can be readily equipped with only access to the BN layers, classifier head and feature head.	76
22	Qualitative comparison of segmentation results.	78
23	In standard adapters, the amount of parameters from the domain-specific models (indicated in colored \mathcal{C}) is equal to or greater than the backbone model (due to the mask represented in black). Budget-Aware Adapters can reduce the number of parameters required for each domain (unused parameters are denoted in gray). However, the whole model is needed at test time if handling distinct domains (colored areas share few parameters). Our model encourages different domains to use the same parameters (colored areas share most of the parameters). Thus, when handling multi-domains at test time, the unused parameters can be pruned without affecting the domains.	87
24	Overview of our strategy for sharing parameters among domains. Colors represent data (i.e., weights, masks, etc), therefore, the colored squares denote the input data for each operation as well as its resulting output.	88
25	NeuraLang code (left) and the JGNN code that prepares to train it on a dataset (right).	92
26	Example convergence of train, validation, and test set losses of ULA (left) and GCNRNI (right).	101
27	Interactive chat application on a smartphone	124
28	The MELiTA process in a simplified feature map for this use case, with grey cells occupied by elites. From one selected elite E , the changed image (e'_V) produces three candidate solutions from elites E, R_1, R_2 . Based on their CLIP score, the ordered list of candidates is $\vec{L} = \{R'_2, E', R'_1\}$. Since $q(R'_2) > q(R_2)$ the candidate R'_2 (that merges the image from E' and text from R_2) replaces R_2 . If $q(R'_2) \leq q(R_2)$ then E' would occupy the empty cell at (5,0). Dotted lines denote temporary individuals that are lost after this parent selection.	130
29	Metrics of the archives after 2000 selections in MAP-Elites and MELiTA. Box plots summarize values from 10 runs per title.	131
30	The text and image assets generation sequence. Pixel dimensions for images are also shown. Arrows delineate the inputs used to generate the respective assets. Outputs from block #1 were also used in all image prompts (arrows not shown).	133
31	Screenshots showing the dungeon crawling (a and c) and card combat sections (b and d) of CrawLLM with an ancient Egypt (Theme #1) and a pirate theme (Theme #2).	133
32	The complete trajectory of the same lander in the dynamic Lunar Lander environment, before an environment shift (yellow) and after (red).	136
33	Visualisation of all feasible solutions found in a single run of the FI-MAP-Elites algorithm on the Complex Test Case. Meshes in yellow have a higher fitness (i.e lower elastic energy). All feasible designs are found in the first 13 by 14 cells (BC1 by BC2) of the feature map; empty cells beyond this range are omitted.	138





34	<p>Problem representations obtained with traditional class-wise histograms (first row) and with our proposed mechanism based on GMMs (second row) on a 3-class sentiment problem (the dataset is called “wb” and belongs to the “Tweets” group described). The first three columns (A,B,C) show the model representations for the training sets L_1, L_2, and L_3, while the last column (D) shows the representation for the test set U. The quantification problem is framed as the task of reconstructing (D) as a convex linear combination of (A,B,C).</p>	149
35	<p>Learnable histogram layer with hard binning and learnable bin centers and widths. The individual components are common operations used in DL frameworks that we use to compute Equation 21.</p>	153
36	<p>We show the differences between node classification and network quantification on a partially unlabeled graph where nodes may belong to the “blue” or “green” classes. unlabeled nodes are shown in gray. Node classification (a) is performed by a node classifier C, which takes as input the partially unlabeled graph and returns as output an isomorphic graph where the class of the unlabeled nodes has been predicted. In contrast, network quantification (b) uses a quantifier Q, which takes as input subsets of unlabeled nodes (in light pink shades) and returns as output their class distribution. In this work, we study network quantification under prior probability shift.</p>	159
37	<p>Box plot of the number of gates reduced by the agent once trained. Tests are performed for two-different tasks and across several circuit sizes, both in terms of number of qubits and average circuit depth. Statistics are drawn from one-thousand executions of random circuits for each circuit size and task. (a) Reduced single-qubit and two-qubit gates. Different shades of green depict different amount of qubits. (b) Number of two-qubit gates reduced. Different shades of blue depict different amount of qubits.</p>	171





1. Executive Summary

This deliverable presents the research outcomes obtained between M37 and M48 as a result of the activities carried out in WP3 (New Learning Paradigms & Distributed AI), and more specifically in Task 3.1 (Lifelong and On-line Learning), Task 3.2 (Manifold Learning and Disentangled Feature Representation), Task 3.3 (Transfer Learning), Task 3.4 (Neural Architecture Search), Task 3.5 (AI at the Edge, Decentralised and Distributed Learning), Task 3.6 (Deep Quality Diversity), Task 3.7 (Learning to Count) and Task 3.8 (Quantum assisted reinforcement learning). All activities address problems that are central in the Machine Learning community and with methodologies that are at the forefront of the developments in the field. This is reflected by the fact that a large number of the works presented here have resulted in publications in prestigious and authoritative international journals and conferences in the field. Beyond this, and to increase the impact in the field, many works provide software. We make explicit references to the corresponding publications and/or software provided by each partner and establish connections of the presented work with the WP8 Use Cases and media industry related applications in general.

Below, we give a concise motivation and overview of the work in each task – more detailed explanations are found in the relevant sections.

- The **lifelong and on-line learning (Task 3.1)** address the problem of training models which evolve gradually as new data are ingested. This is central to the media industry since new concepts and events occur continually and the underlying Machine Learning models used for their automatic analysis need to be updated continually to ensure an up-to-date processing. This poses certain challenges since it is necessary to ensure a balance between stability and plasticity, two properties which account for the performance obtained for past and new data at each stage of the lifelong or on-line learning processes.

More specifically, the contributions presented in Section 4 of this deliverable include: (a) using contrastive learning for generalized category discovery, (b) analyzing the role of pre-training for class-incremental learning, (c) improving the stationarity of representation for continual image retrieval, (d) using generative representation learning for causal timeseries forecasting, (e) introducing a collaborative version of knowledge distillation via a learning-by-education approach, and (f) assessing the efficiency of DNN knowledge in a multi-agent environment.

- The **manifold learning and disentangled feature representation (Task 3.2)** addresses the problem of learning representations of the data that are meaningful for performing generative and discriminative tasks. This includes generating easily synthetic data, such as faces with the desired expression, manipulating images of people so as not to be identifiable and finding better metrics for comparing images so as to perform search and retrieval.

More specifically, the contributions presented in Section 5 of this deliverable include: (a) introducing a flow-factorized representation learning that combines disentanglement and symmetry in the representation space, (b) adding parts-of-speech grounded subspaces in vision-language models, and (c) improving fairness using vision-language driven image augmentations.

- The **transfer learning (Task 3.3)** addresses the problem of reusing previously generated models for tasks that are different than the original ones, tackling the problem of catastrophic forgetting. Considering the huge amount of data, human effort, and computational power needed to train these models, being able to reuse them is of paramount importance.

More specifically, the contribution presented in Section 6 of this deliverable targets test-time domain adaptive semantic segmentation by leveraging each instance to dynamically adapt its segmentation in a non-parametric manner.





- The **neural architecture search (Task 3.4)** addresses the automatic optimization of deep learning architectures. In particular, it leverages past search experiences on different datasets to accelerate a new search. This overcomes the big problem of traditional NAS methods which start for every new search from scratch.

More specifically, the contribution presented in Section 7 of this deliverable leverages parameter sharing or differentiable architecture search in the scope of transfer NAS and explores its applicability for multi-objective NAS and tasks beyond image classification.

- The **AI at the edge, decentralized, and distributed learning (Task 3.5)** focuses on the application of AI directly on edge devices and servers, for model inference and training. This is in contrast to the current dominant paradigm of deployment at centralized infrastructures. AI at the edge is attractive due to the increased costs of aggregating computational resources and data at cloud infrastructures, as well as the privacy and confidentiality requirements of end user data.

More specifically, the contributions presented in Section 8 of this deliverable include: (a) proposing a framework for graph analysis and learning at the edge, (b) training GNN architectures efficiently at the edge, (c) studying genetic algorithms for federated learning, (d) adapting flow-preserving knowledge distillation for model compression, (e) optimizing knowledge distillation trade-offs for CNNs and transformers at the edge, (f) filter pruning for lightweight mobile face detection, (g) porting large language models to mobile devices for question answering, and (h) introducing AdaFamily, an optimization algorithm for training deep neural networks.

- The **deep quality diversity (Task 3.6)** studies ways of handling deceptive search spaces by finding a maximally diverse collection of individuals (with respect to a space of possible behaviors) in which each member is as high performing as possible.

More specifically, the contributions presented in Section 9 of this deliverable include: (a) studying the role of multimodal quality diversity in creative domains, (b) using large language models for automated game generation, (c) adapting quality diversity algorithms to solve dynamic optimization problems, and (d) using quality-diversity search for constrained structure design.

- **Learning to count (Task 3.7)** addresses the problem of training (under the supervised learning paradigm) estimators of quantities. The main categories of sub-tasks falling under this problem are LQ, which is concerned with training unbiased estimators of class prevalence (i.e., learning to estimate, given a sample of objects, the percentage of objects that belong to a given class), and “Learning to count objects”, which concerns learning to estimate the number of objects (which may be inanimate objects, such as cars, but may also be animate objects, such as people or animals) in visual media, such as still images or video frames.

More specifically, the contributions presented in Section 10 of this deliverable include: (a) measuring fairness under unawareness of sensitive attributes with a quantification based approach, (b) investigating binary quantification and dataset shift, (c) introducing kernel density estimation for multiclass quantification, (d) using permutation-invariant networks based on histograms for quantification, (e) quantifying query fairness under unawareness, (f) learning to quantify graph nodes, and (g) using quantification to predict classifier accuracy under prior probability shift.

- **Quantum Reinforcement Learning (Task 3.8)** studies the role of conventional and classical algorithms in reinforcement learning. Particular attention is given to the assumptions made by the algorithms and their usability in real-world conditions.





More specifically, in this work we report a method to optimize quantum circuit depth via reinforcement learning. This contribution is presented in Section 11 of this deliverable.

In summary, the work presented in this deliverable has resulted in:

- 26 conference articles (such as NeurIPS, CVPR, ICCV, ICML) and 4 journal articles (JAIR, DMKD),
- 20 open-source software and tools publicly available (e.g., in GitHub),
- 6 preprints that are currently submitted for publication.

The remainder of this deliverable is structured as follows. In Section 2, we introduce each WP3 task and we give an overview of the contributions of each partner. In Section 3, we provide concise descriptions of the presented works, while detailed descriptions of contributions are given for each task in Section 4 (Task 3.1), Section 5 (Task 3.2), Section 6 (Task 3.3), Section 7 (Task 3.4), Section 8 (Task 3.5), Section 9 (Task 3.6), Section 10 (Task 3.7), and Section 11 (Task 3.8). All the methods presented in this deliverable can be applied to media-related areas and applications. Indeed after describing each method, we also present their relevance to WP8 Use Cases. Finally, Section 12 concludes the deliverable by summarizing the work covered as well as presenting the ongoing work regarding each task addressed in this deliverable.





2. Introduction

The goal of WP3 is to investigate new learning paradigms, looking beyond current achievements in deep learning and focusing, among others, on topics such as lifelong and continuous learning, manifold and transfer learning, deep quality diversity, and learning to count. In the following, we briefly discuss the challenges related to each of these research topics.

Whilst standard deep learning methods typically assume that all the required training data are readily available, this often poses an unrealistic condition in practice since real-world application-related data often arrive in streams, while their characteristics may vary over time. **Lifelong learning and on-line learning** are two closely related research areas that aim to train models which evolve gradually as new data are ingested. Advances in these fields are in dire need in AI4Media in order to keep up with the dynamic nature of news and media content since new events constantly appear in them, and the models used for their automatic analysis need to be updated regularly to ensure up-to-date processing. The contributions of the AI4media partners target important open problems in this research area: (1) the automatic discovery of novelty, (2) the use and adaptation of large pre-trained models, (3) the stationarity of continual representations, and (4) the knowledge transfer from large to small models. The current contributions of the AI4Media project regarding lifelong and online learning methodologies are given in Section 4. During the reporting period, three conference papers were accepted, and their pre-prints were under review.

In recent years, **manifold and disentangled feature representation learning** have risen as a prominent research area addressing the problem of finding meaningful representation schemes for both the generative and the discriminative learning paradigms. Dataset biases can be reduced by studying the structure of latent spaces of generative methods (such as GANs) by discovering semantic paths that govern the generation process and, therefore, generating in a controllable manner synthetic data [14]. Similarly, finding directions in the latent space can help model modes of variation and disentangle different types of transformations [15,16]. Advances in both generative and discriminative regimes are particularly useful in media generation and visual content analysis. The current contributions of the AI4Media project regarding manifold and disentangled representation methodologies are given in Section 5. During the reporting period, four conference papers were accepted, and three pre-prints were under review.

Taking into consideration the vast amount of data, human labour, and computational power that is needed in training modern Deep Learning models, in recent years, practitioners and researchers have devised **Transfer Learning** techniques that allow to reuse and benefit from previously generated models for various purposes. This is particular useful to the media industry and the use cases of AI4Media, since transfer learning methods provide solutions to analyze/adapt the visual content (by virtue of being able to generalize under domain-gap), discover new visual content, and adapt accordingly. Beyond practical reasons, Transfer Learning poses an important scientific challenge, as it forces researchers to explore the internal knowledge representation of deep models and unveil their structure and how learning is being conducted before being able to reuse them for diverse purposes. Advances in this field have potential relevance for key aspects of Deep Learning, not only explainability and interpretability, but also efficiency and footprint reduction, as well as deployment of AI powered systems in real world scenarios. These are relevant to other work packages of AI4Media – i.e., regarding explainable and interpretable AI (WP4) and learning from scarce real-world data (WP5). The current contributions of the AI4Media project regarding transfer learning methodologies are given in Section 6. During the reporting period, two conference papers were accepted.

Deep learning methods are very successful in solving tasks in machine translation, image and speech recognition. However, the search for suitable architectures is a time-consuming, arduous, and error-prone task. In this task, we focus on **transfer neural architecture search** (TNAS),





which should leverage past search experiences on different datasets to accelerate a new search. This overcomes the big problem of traditional NAS methods which start for every new search from scratch. Specifically, we focus on how to leverage parameter sharing or differentiable architecture search in the scope of TNAS and explore its applicability for multi-objective NAS and tasks beyond image and text classification. The current contributions of the AI4Media project regarding NAS methodologies are given in Section 7. During the reporting period, one conference paper was accepted.

We also focus on emerging **AI at the edge, distributed and decentralized technologies** that are in contrast with the dominant paradigm of deployment at centralized architectures. One of the objectives of this task is developing efficient strategies and algorithms for distributed heterogeneous training. We worked on (1) collaborative learning including federated, distributed and gossip learning, (2) model compression to reduce the size and execution time of AI models and (3) in-device processing to facilitate execution of AI operations. The current contributions of the AI4Media project regarding AI at the edge, distributed and decentralized technologies are given in Section 8. During the reporting period, eight conference and two journal papers were accepted.

Finding a maximally diverse collection of individuals (regarding a space of possible behaviours) in which each member is performing as high as possible is an important research field, finding applications, among other areas, in media content that have strict quality requirements (such as games). **Quality-Diversity (QD)** methods have been recently appearing in the EC literature as a way of handling such deceptive search spaces. Drawing inspiration from natural evolution, which – unlike the objective-based optimization tasks to which EC is typically applied – is primarily open-ended, QD algorithms re-introduce a notion of localized quality among individuals with the same behavioural characteristics. QD algorithms aim to obtain balance between their individuals' quality and their population's diversity, and thus media content with strict quality requirements, such as games that are start-to-end playable, are the ideal ground for advancing quality-diversity. The developed QD methods are useful in the media industry and the AI4Media use cases by providing tools for (i) generating diverse content without requiring ad-hoc designer-specified directions for this diversity, and (ii) modelling the subjective human game players experiences to dynamically adapt the game according to the predicted user's emotional responses. The contributions of the AI4Media project regarding deep quality diversity methodologies are given in Section 9. During the reporting period, four conference papers were accepted.

Datasets that are being used by the research community and the industry when applying machine learning models typically exhibit shift, i.e., the joint distribution of the independent and the dependent variables is not the same in the training data and in the unlabelled data for which predictions must be obtained. When this happens, estimating the prevalence of the classes of interest in the unlabelled data is difficult, since “traditional” learning methods assume these prevalence values to stay approximately constant. **“Learning to Count”** is concerned with developing techniques for estimating quantities in unlabelled data possibly affected by dataset shift, where these quantities can be the prevalence values (i.e., relative frequencies) of the classes of interest (as needed in applications such as monitoring consensus for a certain policy or political candidate in social media) or the number of physical objects in instances of visual media (such as estimating car park occupancy from surveillance camera images, or monitoring traffic volumes from road cameras). The contributions of the AI4Media project regarding learning-to-count methodologies are given in Section 10. During the reporting period, five conference and two journal papers were accepted, and two pre-prints were under review.

Quantum computers are a new resource to perform computations using Physical principles, and some devices are already available. However, due to the limited capabilities of these pioneering systems, they are restricted to a small number of processing tasks. To overcome this limitation, researchers are proposing a hybrid approach based on the combination of Quantum computers





with conventional resources. Here, we present some results in this direction, using the advances in Reinforcement Learning to reduce the requirements on Quantum devices by reducing the size of Quantum circuits. The contributions of the AI4Media project regarding Reinforcement Learning for Quantum Computation are given in Section 11. During the reporting period, one pre-print was under review.

To summarize, the contributions presented in this deliverable but also previous ones address problems that are central to the Machine Learning community, providing methodologies that are at the forefront of the developments in the field. The partners' activities led to a significant number of high-quality and diverse works that have been published in some of the most prestigious and authoritative international journals and conferences in the field.





3. Concise descriptions of the presented works

In the following, we briefly summarise the outcomes of each WP3 task for the period M37-M48. These works are then presented in detail in sections 4-11.

3.1. Lifelong and on-line learning (Task 3.1)

3.1.1. Introduction

Standard deep learning methods assume that all training data are available at once. This hypothesis is often unrealistic since application-related data arrive in streams, and their characteristics shift over time. Lifelong learning and on-line learning are two closely related research topics whose purpose is to train models which evolve gradually as new data are ingested. This learning process is challenging because it is necessary to ensure a balance between stability and plasticity, two properties which account for the performance obtained for past and new data at each stage of the lifelong or on-line learning processes. Advances in these fields are particularly needed in AI4Media in order to keep up with the dynamic nature of media content (e.g., breaking news). New concepts and events occur continually in them and the underlying models used for their automatic analysis need to be updated continually to ensure an up-to-date processing.

We report the research outcomes of Task 3.1 in detail in Section 4.

3.1.2. Overview

The partners involved in Task 3.1 tackle different open challenges in lifelong and on-line learning. The contributions are summarized below and then discussed in more detail in the following subsections.

In Subsection 4.1, UNITN proposes a novel dynamic conceptional contrastive learning (DCCL) framework to effectively leverage the underlying relationships between unlabeled samples for learning discriminative representation for generalized category discovery (GCD). The DCCL approach consistently achieves superior performance over state-of-the-art GCD algorithms on both generic and fine-grained tasks.

In Subsection 4.2, CEA investigates the impact of initial training strategies on continual learning. Focus is put on (1) comparing pre-training versus initial training and (2) studying the effect of enriching the initial dataset with synthetic data. Results are nuanced for the first research question because no strategy is best in all scenarios. As for synthetic data, they help when the initial dataset is small but their positive effect fades otherwise.

In Subsection 4.3, UNIFI addresses the stationarity of deep representations. This property is desirable for the reuse of the embeddings extracted with previous models when introducing new ones in the processing pipeline of dynamic retrieval systems. The contribution includes a theoretical justification of previous contributions and the proposal of a loss function that preserves higher-order dependencies when updating deep representations.

In Subsection 4.4, AUTH presents a method for efficient knowledge transfer between DNNs, where a specialized Student can learn from one or multiple Teacher networks. This method introduces an online Collaborative Knowledge Distillation (CKD) technique that achieves state-of-the-art results. Interconnected nodes can continuously learn without forgetting through peer-to-peer connections.

In Subsection 4.5, AUTH explores a novel self-assessment mechanism designed specifically for environments with multiple DNN agents to evaluate their knowledge efficiently through an Out-of-Detection pipeline.





3.2. Manifold learning and disentangled feature representation (Task 3.2)

3.2.1. Introduction

In recent years, manifold and disentangled feature representation learning have risen as a prominent research area addressing the problem of finding meaningful representation schemes for both the generative and the discriminative learning paradigms. Studying the structure of latent spaces of generative methods (such as GANs) by discovering semantic paths that govern the generation process, and therefore generating in a controllable manner synthetic data that can reduce dataset biases [14]. Similarly, finding directions in the latent space can help model modes of variation and disentangle different types of transformations [15, 16]. Advances in both generative and discriminative regimes are particularly useful in media generation and visual content analysis.

We report the research outcomes of Task 3.2 in detail in Section 5.

3.2.2. Overview

Within this task partners are contributing in fundamental aspects of manifold learning and disentangled feature representation, with the use cases of the project in mind. The contributions are summarized below and then discussed in more detail in subsequent subsections.

In Subsection 5.1, UNITN proposes a novel way of modeling paths in the latent space that correspond to distinct transformations in the image space. The latent flow paths are generated by the gradient field of some learned potentials following fluid mechanical dynamic Optimal Transport (OT), and the modeling allows for novel understandings of both *disentanglement* and *equivariance*.

In Subsection 5.2, QMUL proposes a way to better isolate representations of the different visual properties of an image/text in CLIP vision-language models. They propose a subspace learning methodology that captures specific target variation. The learnt subspaces offer a range of practical benefits including blocking the synthesis of custom visual themes in CLIP-based text-to-image synthesis.

In Subsection 5.3, QMUL and UNITN study how to incorporate Vision-Language driven image augmentations to improve the fairness of existing (biased) datasets and, consequently, the fairness of discriminative models trained on such datasets. For doing so, they propose to incorporate the power of a pre-trained diffusion model that can modify sensitive facial attributes (e.g., age or skin colour) from a pool of synthetic facial images. The manipulated faces (with respect to the desired sensitive attributes) are used to make the original dataset fairer and mitigate the bias present on a downstream model trained on the original dataset. Quantitative results show how the augmented images help the model improve the overall accuracy, the aforementioned metric, and the disparity of equal opportunity.

3.3. Transfer learning (Task 3.3)

3.3.1. Introduction

Transfer Learning is an emerging field among Deep Learning practitioners that seeks to reuse and exploit previously generated models for different purposes. Considering the huge amount of data, human effort and computational power needed to train these models, being able to reuse them is of paramount importance. Beyond practical reasons, Transfer Learning poses a scientific challenge of relevance, as it forces researchers to question the internal knowledge representation of deep models. Indeed, to understand how to reuse deep representations, one must first understand how are these representations learned, and how are they internally structured. Advances in this field





have potential relevance for key aspects of Deep Learning, such as explainability and interpretability, efficiency and footprint reduction, and real world deployment of AI powered systems. We report the research outcomes of Task 3.3 in detail in Section 6.

3.3.2. Overview

Within this task partners are contributing in fundamental aspects of Transfer Learning, coordinately so that their advances can contribute to one another, and with the use cases of the project in mind.

In Subsection 6.1, UNITN introduces an efficient backward-free approach for test-time domain adaptive semantic segmentation (TTDA-Seg) which can be implemented within one forward propagation with a light computation cost. The approach is effective in adapting the model in both distribution and semantic aspects. The method is easy to implement and is model-agnostic, allowing it to be readily injected into existing models. The experiments conducted on three source and five target domains based on driving benchmarks and show that our method produces new state-of-the-art performance for TTDA-Seg.

In Subsection 6.2, CNR proposes a novel benchmark for the evaluation of fine-grained linguistic and temporal grounding capabilities of Video and Language Models focusing on the alignment between the heterogeneous feature spaces derived from texts and videos. The benchmark is structured into five action-based tasks, each designed around an action-related linguistic phenomenon. By assessing a series of state-of-the-art VidLMs and Image-and-Language Models, the benchmark highlights that current Video and Language Models do not consistently align the two modalities, obtaining scores comparable to those obtained by Image Language Models, working only with static images.

3.4. Neural Architecture Search (Task 3.4)

3.4.1. Introduction

Deep learning models are becoming indispensable tools in many industries. Implementing them, however, requires a high level of expertise in neural network architectures. Additionally, the architectures one would select tend to vary significantly, depending on data domains and target AI tasks. To make matters worse, once adequate architectures are selected, making the best architectural choices, such as the hyperparameter training process selection, is a tedious task that relies on trial-and-error. In particular, experts need to rely on their past experiences, technical expertise and understanding of the target application domain. Finally, evaluating the performance of such models is typically reduced to experts in the selected area in which the model will operate. For all these reasons, the need to automate as much as possible the processes involved in building deep learning neural networks is critical in order to fully democratise access to such technology across all industries.

3.4.2. Overview

The recent outcomes of Task 3.4 are presented in detail in Section 7.1. The research conducted by UNITN addresses the multi-domain learning problem while taking into account a user-defined budget for computational resources. The solution is to prune a single model for multiple domains, making it more compact and efficient. This is done by encouraging the sharing of parameters among domains, allowing the pruning of the weights that are not used in any of them, reducing both the computational complexity and the number of parameters to values lower than the original baseline for a single domain. The presented results are competitive with other state-of-the-art methods





while offering good trade-offs between classification performance and computational cost according to the user's needs.

3.5. AI at the Edge, decentralised and distributed learning (Task 3.5)

3.5.1. Introduction

The integration of mobile devices and the Internet-of-Things (IoT) in every aspect of people's lives has created new kinds of media data (e.g., social network interactions, multimedia features, sensor readings) at unprecedented scales that reach big data and beyond [17–20]. Traditional AI training and deployment considers centralized web services that can be queried through one endpoint. However, services with computationally heavy AI struggle to gather and process the ever-expanding resources of the interconnected world [21–24], especially when large language models or computer vision systems require large amounts of resources to train and run. Even worse, real world data are rarely homogeneous (e.g., identically distributed), while concerns about their privacy and ownership are constantly being raised. These limitations of centralized computing have motivated new machine learning paradigms that cover heterogeneous global-scale data by moving parts or all of training and inference to decentralized edge devices, or by orchestrating distributed training schemes that leverage the computational capabilities of multiple devices. We report the research outcomes of Task 3.5 in detail in Section 8.

3.5.2. Overview

In Subsection 8.1, CERTH describes programming libraries that facilitate graph analysis and learning at the edge. In particular, they present a Java Graph Neural Network library (JGNN) for the implementation of machine learning models like Graph Neural Networks (GNNs), and pygrank, which is Python library for the creation of complex yet efficient node ranking algorithms. Together, the libraries support a wide range of graph algorithms across a wide variety of edge device hardware. JGNN simplifies model declarations by introducing a scripting language, called NeuraLang, that declares neural layers as functions in a few lines of code. For pygrank, approaches are provided to non-convex nature of graph-based objectives to select algorithm hyperparameters on-the-fly based on the actual data each edge device encounters.

In Subsection 8.2, CERTH presents two methodologies for GNN model training at the edge. The first methodology applies to peer-to-peer networks, where communication links coincide with social relations, and trains fragments of GNNs within each device so that the predictions it makes about its user approximates a hypothetical centralized equivalent. The second methodology tackles in-device training by offering a lightweight GNN that may lack expressive power compared to alternatives but in the end manages to better satisfy training objectives by leveraging a novel property we introduce for easy optimization that is called local attraction.

In Subsection 8.3, CERTH proposes three nature-inspired algorithms for federated-algorithms, namely, Federated Particle Swarm Optimization (FedPSO), Federated Ant Colony Optimization (FedACO), and Distributed Differential Evolution (DDE). These algorithms preserve privacy and reduce the communication overhead of traditional federated learning schemes, making them especially attractive for scenarios where data privacy and network optimization are paramount. The three algorithms are compared against the standard Federated Averaging (FedAvg) algorithms in a network of nodes with non-IID data distributions.

In Subsection 8.4, CERTH describes their work on compressing heavy CNN models. Specifically, they present InDistill, a model compression approach that combines knowledge distillation and channel pruning in a unified framework for the transfer of the critical information flow paths from





a heavyweight teacher to a lightweight student. Such information is typically collapsed in previous methods due to an encoding stage prior to distillation. By contrast, InDistill leverages a pruning operation applied to the teacher’s intermediate layers reducing their width to the corresponding student layers’ width. In that way, the forced architectural alignment enables the intermediate layers to be directly distilled without the need for an encoding stage. Additionally, InDistill adopts a curriculum learning-based training scheme considering the distillation difficulty of each layer and the critical learning periods in which the information flow paths are created.

In Subsection 8.5, CERTH performs a comprehensive investigation of knowledge distillation methods for AI vision models at the edge. In particular, it investigates four critical questions on the following topics: 1) different teacher and student model architectures, namely, CNNs and ViTs, 2) the capacity gap between teacher and student, 3) different resolution of the input images, and 4) fine-tuning of the student model. These questions are highly relevant to AI practitioners and involve not only the complexity of the distilled model but also the complexity of the knowledge distillation process itself, determining its practical execution on the edge.

In Subsection 8.6, CERTH proposes the Filter Pruning via Geometric Median algorithm for the pruning of two AI models for face detection. This approach leads to highly compact models that can be deployed at end devices. In addition, CERTH proposes a framework for adapting the pruning rate to different network layers based on Bayesian optimization. This framework exploits the variable compression potential of different layers, derives the optimal pruning rates for each layer independently of the actual pruning method, and outperforms the uniform pruning rate approach on a face detection task evaluated with mean average precision.

In Subsection 8.7, JR describes a methodology for the porting of an LLM application for question answering on mobile devices. This avoids sending queries to a cloud application, which can jeopardize user privacy, and can support advanced natural language processing applications at the mobile device like virtual assistants, language translation, text summarization and named entity recognition. From a technical perspective, JR proposes a toolchain based on a C+* framework instead of the more popular TensorFlow Lite scheme, which would entail high complexity and maintenance issues. The experimental results show that the LLM app runs correctly and fast enough for an interactive chat on a Samsung Galaxy S21 smartphone.

In Subsection 8.8, JR presents the AdaFamily training algorithm, a family of adaptive gradient methods that is a configurable blend of the AdamW, AdaBelief and AdaMomentum algorithms. The latter algorithms have proven to be advantageous variations of the standard Adam algorithm, and AdaFamily manages to combine their benefits. The experimental results show that AdaFamily outperforms the individual algorithms on an image classification task and the blending parameter offers further opportunities for optimization. Efficient optimization is important for in-device AI training and the AdaFamily algorithm can be extended for distributed training.

3.6. Deep quality diversity (Task 3.6)

3.6.1. Introduction

QD algorithms have been recently introduced to the EC literature as a way of handling deceptive search spaces. The goal of these algorithms is “to find a maximally diverse collection of individuals (with respect to a space of possible behaviours) in which each member is as high performing as possible” [25]. The inspiration for such approaches is natural evolution, which is primarily open-ended—unlike the objective-based optimization tasks to which EC is often applied. While the rationale of open-ended evolution has been previously used as an argument for genetic search for pure behavioural novelty, QD algorithms re-introduce a notion of (localized) quality among individuals with the same behavioural characteristics. QD algorithms attempt to balance between





their individuals' quality and their population's diversity, and thus media content which have strict quality requirements, such as games that are playable from start to finish, are the ideal arena for advancing quality-diversity.

The aim of Task 3.6 is to couple deep neural network architectures with divergent search for transforming exploration, aiming for both diverse and high quality outcomes. Experiments in this deepQD search approach during the reported period are aligned on two main directions:

D1: improve the definition of diversity based on learnt representations.

D2: promote diversity and quality in existing deep learning generative architectures for media.

We report the research outcomes of Task 3.6 in detail in Section 9.

3.6.2. Overview

Since the progress reported in Deliverable 3.3, we present one main updated contribution with respect to the research direction D1 described above, as well as three complementary research directions on Dynamic QD (Section 9.3), LLMs for automated game generation (Section 9.2), and constrained QD search for shell structures (Section 9.4) which can be merged in the future with deep QD.

In Subsection 9.1, UM describes a continuation of their work on the Map-Elites with Transverse assessment (MEliTA) algorithm, an extension of the MAP-Elites algorithm for generating multimodal artefacts using deep QD. The primary innovation of MEliTA lies in its inter-modal evaluation process where phenotypically similar elites share partial artefacts (e.g. an image or text) in order to promote more coherent pairings. MEliTA is tested on a bimodal use case for generating novel text and artworks for fictional game titles. For the text modality, two separate Generative Pre-Trained 2 (GPT-2) models were fine-tuned on a dataset of 72,000 Steam game titles and descriptions. The text can be mutated either partially or fully, and is characterized for the archive using topic modelling via the Latent Dirichlet Allocation (LDA) algorithm. For the image modality, MEliTA leverages Stable Diffusion (SD) to generate game cover arts which closely align with the title and description. Images are mutated using augmentation functions from the Torch Vision software library, as well as using an image-to-image SD model, and are characterized by their colourfulness and complexity. Results indicate that MEliTA can significantly improve text-to-image mappings within the solution space compared to typical MAP-Elites, and is a promising step forward for multimodal bottom-up orchestration.

Subsection 9.2 is a potentially interesting use-case extension for UM's previous work in Section 9.1, which spans more than two modalities and involves re-theming games to new settings and narratives using LLMs. For this study, UM introduces an original game called CrawLLM, which combines dungeon crawling gameplay with card-based combat. The structure of the gameplay is used to guide the construction of prompts for LLMs to generate new themes, stories, characters, and locations, which in turn direct the generation of other modalities such as visual assets and sound. In this initial proof of concept, a demo of the game is shown running 20 pre-generated and non-curated themes, with a plan to expand this study into an automated game asset generation pipeline using UM's MEliTA QD approach.

In Subsection 9.3, UM describes an approach to performing QD search in dynamic environments, where the fitnesses of individuals are irregularly influenced by external factors and require re-evaluation to avoid suboptimal solutions. They introduce a new and generalizable dynamic QD framework that keeps an archive of past solutions updated when shifts in the environment are detected. Furthermore, they present a novel characterization of dynamic environments that can be easily applied to existing benchmarks with minor changes. Finally, UM presents a test case using





Map-Elites and Covariance Matrix Adaptation MAP-Elites (CMA-ME) on the dynamic sphere and dynamic lunar lander environments. Results across different quality metrics show dynamic QD is able to efficiently highlight the tradeoff between accuracy and computational complexity in dynamic environments, and this work can act as a basis for further research on the problem.

Finally, Subsection 9.4 describes UM’s work on the exploration of design spaces using a constrained version of QD search for generating shell structures. Specifically, they introduce a Feasible-Infeasible variant of the Map-Elites algorithm called FI-MAP-Elites. This approach maintains and evolves two separate archives, one for feasible solutions and one for infeasible solutions, ensuring that the algorithm can explore the entire design space while still satisfying design constraints. In order to make this algorithm more easily accessible, UM have packaged it into a tool within the Rhino/Grasshopper environment, and conducted a case study of parametric models of shell structures with varying levels of complexity. Results show that FI-Map-Elites is capable of producing a more diverse array of feasible solutions compared to current single-objective optimization available within Rhino. This could potentially offer engineers a broader range of design alternatives to choose from during the conceptual design phase.

3.7. Learning to count (Task 3.7)

3.7.1. Introduction

In machine learning and data mining, estimating the frequencies of classes in sets of unlabelled data is usually known as *quantification*, and the task of training models that do so is known as *learning to quantify*. This task, which has received increased attention in recent years, has applications in many fields where data are analysed at the “macro” (i.e., population) level, and has seen the development of techniques that allow much more accurate prediction of class frequencies than can be obtained by simply “classifying and counting”. In Task 3.7, several efforts have been carried out that involve learning to quantify, including novel methods for learning to quantify (Subsections 10.3, 10.4, 10.6), novel protocols for testing quantification methods (Subsection 10.2), and novel applications of quantification to important tasks, such as estimating classifier accuracy on out-of-distribution data (Subsection 10.7), estimating the fairness of classifiers (Subsection 10.1), and estimating the fairness of rankers (Subsection 10.5).

3.7.2. Overview

In Subsection 10.1, CNR tackle the problem of measuring group fairness under unawareness of sensitive attributes, by using techniques from quantification, a supervised learning task concerned with directly providing group-level prevalence estimates (rather than individual-level class labels). They identify five important factors that complicate the estimation of fairness under unawareness and formalize them into five different experimental protocols under which they assess the effectiveness of different estimators of group fairness. They also consider the problem of potential model misuse to infer sensitive attributes at an individual level, and demonstrate that quantification approaches are suitable for decoupling the (desirable) objective of measuring group fairness from the (undesirable) objective of inferring sensitive attributes of individuals.

In Subsection 10.2, CNR carry out an experimental analysis of how current quantification algorithms behave under different types of dataset shift, in order to identify limitations of current approaches and hopefully pave the way for the development of more broadly applicable methods. They do this by proposing a fine-grained taxonomy of types of dataset shift, by establishing protocols for the generation of datasets affected by these types of shift, and by testing existing quantification methods on the datasets thus generated.





In Subsection 10.3, CNR propose a new representation mechanism for multiclass quantification based on multivariate densities that they model via kernel density estimation (KDE). The experiments they have carried out show that their method, dubbed KDEy, yields superior quantification performance with respect to previous distribution-matching approaches to quantification. They also investigate the KDE-based representation within the maximum-likelihood framework and show that KDEy often shows superior performance with respect to the expectation-maximization method for quantification, arguably the strongest contender in the quantification arena to date.

In Subsection 10.4, CNR propose HistNetQ, a novel neural architecture for quantification that relies on a permutation-invariant representation based on histograms, that is specially suited for quantification problems. Their experiments, carried out in the only quantification competition held to date, show that HistNetQ outperforms other deep neural architectures devised for set processing, as well as the state-of-the-art quantification methods. Furthermore, HistNetQ offers two significant advantages over traditional quantification methods: i) it does not require the labels of the training examples but only the prevalence values of a collection of training bags, making it applicable to new scenarios; and ii) it is able to optimize any custom quantification-oriented loss function.

In Subsection 10.5, CNR propose to use quantification methods (i.e., techniques specifically devised for the estimation of class proportions under dataset shift) for reliably measuring fairness under unawareness in IR systems. Their experiments on the TREC 2022 Fair Ranking Track collection show that quantification techniques significantly enhance the accuracy of determining group prevalence in rankings, which leads to more accurate fairness measurements. They find that their method (which extends beyond binary sensitive groups) significantly outperforms existing baselines for multiple sensitive attributes.

In Subsection 10.6, CNR introduce XNQ, a novel NQ method that synergizes the flexibility and efficiency of the unsupervised node embeddings computed by randomized recursive GNNs, with an Expectation-Maximization algorithm that provides a robust quantification-aware adjustment to the output probabilities of a calibrated node classifier. In an extensive evaluation, they find that our approach consistently and significantly improves on the best network quantification methods to date, thereby setting the new state of the art for this challenging task. Simultaneously, the proposed method provides a training speed-up of up to 10x-100x over other graph-based methods.

In Subsection 10.7, CNR deal with the problem of predicting classifier accuracy on unseen data affected by prior probability shift (PPS), an important type of dataset shift. They propose QuAcc, a method built on top of quantification algorithms robust to PPS, i.e., algorithms devised for estimating the prevalence values of the classes in unseen data affected by PPS. QuAcc is based on the idea of viewing the cells of the contingency table (on which classifier accuracy is computed) as classes, and of estimating, via a quantification algorithm, their prevalence values on the unseen data labelled by the classifier. They perform systematic experiments in which they compare the prediction error incurred by QuAcc with that of state-of-the-art classifier accuracy prediction (CAP) methods.

3.8. Quantum assisted reinforcement learning (Task 3.8)

3.8.1. Introduction

Quantum computing has emerged in the last years as a new technology with the potential to provide enormous computational power using the principles of Quantum Mechanics. As a novel field, much work is still required to bring in practice the advances postulated by the theoretical advances in defining Quantum algorithms. While prototypes of Quantum computers already exist and are operational, limitations in the current technological capabilities to build a large scale device are the main limiting factor to fully expand this alternative to current computational practices. Due to these





limitations, current research has focused on hybrid approaches where Quantum and conventional computers collaborate together to solve a computational problem. Among the resources available, Reinforcement Learning has allowed the optimal control of Quantum devices.

In Section 11, we describe the approach to use hybrid algorithms to fully exploit the capabilities of current and near-future Quantum computers by proposing their interaction with current algorithms. These can be executed in a regular computer, or take advantage of super-computing capabilities. The collection of techniques presented here not only investigate the use of hybrid algorithms to solve problems, but also how to better use conventional techniques to improve the operation of a real Quantum computers, enhancing their current limitations.

3.8.2. Overview

In Subsection 11.1, BSC and collaborators present a Reinforcement Learning approach to reduce the size of a Quantum circuit. The initial description of the Quantum circuit is translated to a description based on the ZX-calculus approach. Inside this formalism, a collection of rules guides complex transformations on the circuit. These operations can be selected using a Reinforcement Learning agent, trained with the goal to reduce the depth of the converted circuit. This approach allows the simplification of Quantum circuits, a scarce resource in current devices required to perform a Quantum computation.





4. Lifelong and on-line learning (Task 3.1)

Contributing partners: CEA, AUTH, UNITN, UNIFI

Standard deep learning methods assume that all training data are available at once. This hypothesis is often unrealistic since application-related data arrive in streams, and their characteristics shift over time. Lifelong learning and on-line learning are two closely related research topics whose purpose is to train models that constantly evolve as new data are ingested. This poses certain challenges in the learning process since balance between stability and plasticity needs to be guaranteed, two crucial properties that account for the performance obtained for past/new data at each stage of the lifelong or on-line learning stages. Advances in these fields are needed in AI4Media in order to keep pace with the dynamic nature of news and media content. New concepts and events occur continually in them and the underlying models used for their automatic analysis need to be updated continually to ensure an up-to-date processing.

4.1. Dynamic Conceptual Contrastive Learning for Generalized Category Discovery

Contributing partner: UNITN

4.1.1. Introduction and methodology

Learning recognition models (*e.g.*, image classification) from labeled data has been widely studied in deep learning [26–28]. In spite of their tremendous success, supervised learning techniques rely heavily on huge annotated data making them fairly unsuitable for open-world applications. Thus, the researchers recently have paid considerable effort on learning with label-imperfection data, such as semi-supervised learning [29,30], self-supervised learning [31,32], weakly-supervised learning [33,34], few-shot learning [35,36], open-set recognition [37] and learning with noisy labels [38], etc.

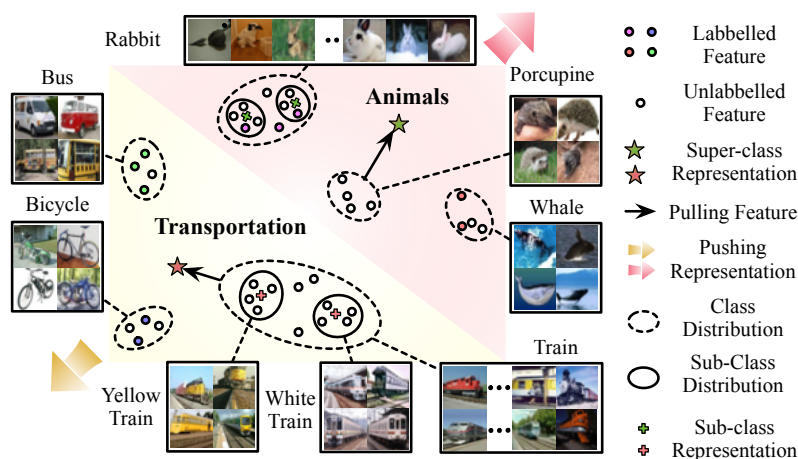


Figure 1. Diagram of the proposed Dynamic Conceptual Contrastive Learning (DCCL). Samples from the conceptions should be close to each other. For example, samples from the same classes (bus) at the class level, samples belonging to the transportation (bus and bicycle) at the super-class level, and samples from trains with different colors at the sub-class level. Our DCCL potentially learns the underlying conceptions in unlabeled data and produces more discriminative representations.



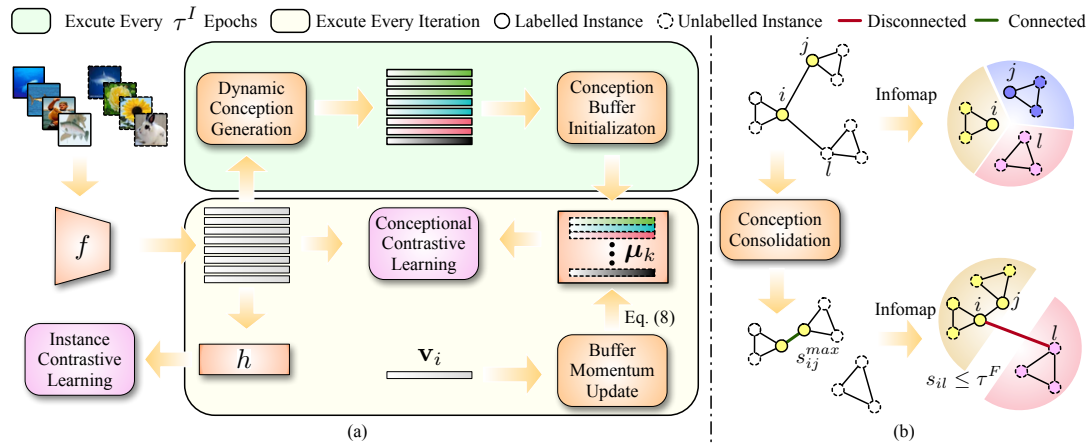


Figure 2. (a) Overview of our dynamic conceptual contrastive learning framework. We first extract features by ViT backbone network, then cluster the features to generate conceptual labels and initialize conception representations, and learn representations by joint instance-level and conception-level objectives. During the training process, the initialization of conception representations and dual-level representation learning are performed alternately, in which the conception buffer is updated every iteration to keep the consistency of the changing instance features and conceptual representations. (b) Illustration of the proposed conception consolidation. Without consolidating the relationships of conceptions by label information, Infomap tends to over-cluster data and thus provides the supervision that has a high risk to over-correct affinities between neighbor instances.

Recently, inspired by the fact that humans can easily and automatically learn new knowledge with the guidance of previously learned knowledge, novel category discovery (NCD) [39, 40] was introduced to automatically cluster unlabeled data of unseen categories with the help of knowledge from seen categories. However, the implementation of NCD is under a strong assumption that all the unlabeled instances belong to unseen categories, which is not practical in real-world applications. To address this limitation, Vaze *et al.* [7] extend NCD to the generalized category discovery (GCD) [7], where unlabeled images are from both novel and labeled categories.

GCD is a challenging open-world problem in that we need to 1) jointly distinguish the known and unknown classes and 2) discover the novel clusters without any annotations. To solve this problem, Vaze *et al.* [7] leverage the contrastive learning technique to learn a discriminative representation for unlabeled data and use k -means [41] to obtain final clustering results. In this method, the labeled data are fully exploited by supervised contrastive learning. However, self-supervised learning is applied to the unlabeled data, which enforces samples to be close to their augmentation counterparts while far away from others. As a consequence, the underlying relationships between samples of the same conceptions are largely overlooked and thus will lead to degraded representation learning. Intuitively, samples that belong to the same conceptions should be similar to each other in the feature space. The conceptions can be regarded as: classes, super-classes, sub-classes, etc. For example, as shown in Figure 1, samples of the same class should be similar to each other, *e.g.*, samples of the bus, samples of the bicycle. In addition, in the super-classes view, classes of the transportation, *e.g.*, Bus and Bicycle, should belong to the same concept. Hence, the samples of transportation should be closer than that of other concepts (*e.g.*, animals). Similarly, samples belonging to the same sub-classes (*e.g.*, red train) should be closer to that of other sub-classes (*e.g.*, white train). Hence, embracing such conceptions and their relationships can greatly benefit the representation learning for unlabeled data, especially for unseen classes.

Motivated by this, we propose a Dynamic Conceptual Contrastive Learning (DCCL) framework for GCD to effectively leverage the underlying relationships between unlabeled data for representation



Table 2. Results on generic image recognition datasets.

Method	CIFAR10			CIFAR100			ImageNet-100		
	All	Old	New	All	Old	New	All	Old	New
k-means	83.6	85.7	82.5	52.0	52.2	50.8	72.7	75.5	71.3
RankStats+	46.8	19.2	60.5	58.2	77.6	19.3	37.1	61.6	24.8
UNO+	68.6	98.3	53.8	69.5	80.6	47.2	70.3	95.0	57.9
GCD	91.5	97.9	88.2	73.0	76.2	66.5	74.1	89.8	66.3
DCCL	96.3	96.5	96.9	75.3	76.8	70.2	81.4	94.5	76.2

learning (see Figure 2). Specifically, our DCCL includes two steps: Dynamic Conception Generation (DCG) and Dual-level Contrastive Learning (DCL). In DCG, we dynamically generate conceptions based on the hyper-parameter-free clustering method equipped with the proposed semi-supervised conceptional consolidation. In DCL, we propose to optimize the model with conception-level and instance-level contrastive learning objectives, where we maintain a dynamic memory to ensure comparing with the up-to-date conceptions. The DCG and DCL are alternately performed until the model converges.

4.1.2. Experimental results

4.1.2.1. Data and Evaluation Metric We evaluate DCCL on three generic image classification datasets, namely CIFAR-10 [42], CIFAR-100 [42] and ImageNet-100 [7]. ImageNet-100 denotes randomly sub-sampling 100 classes from the ImageNet [43] dataset. We further evaluate DCCL on three more challenging fine-grained image classification datasets: CUB-200 [44], Stanford Cars [45], and Oxford-IIIT Pet [5]. The original training set of each fine-grained dataset is separated into labeled and unlabeled parts. We follow [7] and sample a subset of half the classes as “Old” categories. 50% of instances of each labeled class are drawn to form the labeled set, and all the remaining data constitute the unlabeled set. For evaluation, we measure the clustering accuracy by comparing the predicted label assignment with the ground truth, following the protocol in [7].

4.1.2.2. Comparison with State-of-the-Art To evaluate the performances of our DCCL, we conduct three group experiments by comparing our DCCL with three strong GCD baselines, including RankStats [46] and UNO [40] and the state-of-the-art GCD method [7].

Comparison on Generic Datasets. As shown in Table 2, our DCCL is compared with other competitors on the generic image recognition datasets. Overall, the results in Table 2 show that our DCCL consistently outperforms all others by a significant margin. Specifically, DCCL outperforms the GCD method [7] by 4.8% on CIFAR-10, 2.3% on CIFAR-100, and 7.3% on ImageNet-100 for ‘All’ classes, and by 8.7% on CIFAR-10, 3.7% on CIFAR-100, and 9.9% on ImageNet-100 for ‘Unseen’ classes. These results experimentally demonstrate that the generated dynamic conceptions provide effective supervision to learn better representations for unlabeled data. Moreover, UNO+ shows a strong accuracy on “Old” classes, but its accuracy when testing on “New” classes is relatively lower. This is because UNO+ trains the linear classifier on “Old” classes, thus resulting in an inevitable bias. On the contrary, our DCCL gets a relatively good balance on both the “Old” and “Unseen” classes, without being biased to the labeled data.

Comparison on Fine-Grained Datasets. In general, the differences between different classes in fine-grained datasets are subtle, which leads the fine-grained visual understanding to be more challenging for GCD. For verifying the effects of DCCL on fine-grained tasks, we compare our



Table 3. Results on fine-grained datasets.

method	CUB-200			Stanford-Cars			Oxford-Pet		
	All	Old	New	All	Old	New	All	Old	New
k-means	34.3	38.9	32.1	12.8	10.6	13.8	77.1	70.1	80.7
RankStats+	33.3	51.6	24.2	28.3	61.8	12.1	-	-	-
UNO+	35.1	49.0	28.1	35.5	70.5	18.6	-	-	-
GCD	51.3	56.6	48.7	39.0	57.6	29.9	80.2	85.1	77.6
DCCL	63.5	60.8	64.9	45.6	59.2	40.3	88.1	88.2	88.0

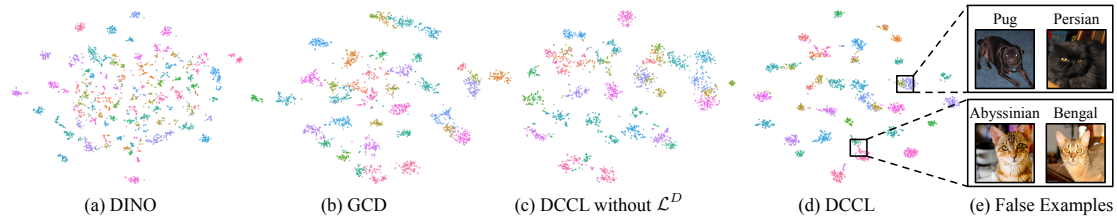


Figure 3. Visualization of features distributions of the unlabeled set of the Pet [5] dataset. (a)-(d) are the results generated from DINO [6], GCD [7], our DCCL without \mathcal{L}^D and full DCCL, in turn. (e) is a visualization of false samples that are easy to be incorrectly clustered.

method with others on fine-grained image recognition datasets. The results in Table 3 show that DCCL consistently outperforms all other methods for “All” and “New” classes. Specifically, on CUB-200 and SCars, DCCL achieves 12.2% and 6.6% improvement over the state-of-the-art for “All” classes. Especially for “New” classes, DCCL outperforms GCD by 16.2% on the CUB-200 dataset. These results demonstrate that our DCCL is efficient in capturing the conceptual information shared across different fine-grained classes, thereby generating precise and helpful supervision for representation learning.

Visualization of Feature Distributions. To qualitatively explore the clustered features on Pets dataset [5], we visualize the t-SNE embeddings projected from the features extracted by pre-trained ViT [6], GCD [7], the DCCL without the proposed dispersion loss and our full DCCL method. As shown in Fig. 3, our features are more discriminative than the features from the pre-trained ViT and GCD. By comparing Fig. 3(c) and Fig. 3(d), the proposed dispersion loss effectively pushes cluster centers away from each other. A large inter-cluster margin not only improves cluster boundaries for “Old” and “New” categories, but also compacts intra-cluster distribution.

4.1.3. Conclusions

In summary, the contributions of this work are as follows:

- We propose a novel dynamic conceptual contrastive learning (DCCL) framework to effectively leverage the underlying relationships between unlabeled samples for learning discriminative representation for GCD.
- We introduce a novel dynamic conception generation and update mechanism to ensure consistent conception learning, which encourages the model to produce more discriminative representation.



- Our DCCL approach consistently achieves superior performance over state-of-the-art GCD algorithms on both generic and fine-grained tasks.

4.1.4. Relevant publications

- N. Pu, Z. Zhong, and N. Sebe, Dynamic Conceptual Contrastive Learning for Generalized Category Discovery, IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2023 [47].
Zenodo record: <https://zenodo.org/record/8337043>

4.1.5. Relevant software/datasets/other outcomes

- The Pytorch implementation can be found in
<https://github.com/TPCD/DCCL>

4.1.6. Relevance to AI4Media use cases and media industry applications

We have analyzed how to cope with the generalized category discovery (GCD) from the perspective of mining underlying relationships between known and unknown categories. To implement this idea, we proposed a dynamic conceptual contrastive learning framework to alternately explore latent conceptual relationships and perform conceptual contrastive learning. This mechanism enables models to learn more discriminable representations. The approach could be directly relevant to use cases (a) 3A3 (archive exploration), specifically 3A3-11 Visual indexing and search and (b) 7A3 (Re)organisation of visual content by supporting the efficient training and organization of image and video collections. However, the approach can also be applied when other modalities are involved, e.g., 4C3 (audio analysis).

4.2. A reality check on pretraining for exemplar-free class-incremental learning

Contributing partner: CEA

4.2.1. Introduction and methodology

4.2.1.1. Introduction Continual learning refers to systems capable of learning new tasks over time [48,49]. In particular, Class-Incremental Learning (CIL) deals with classification problems where new classes are integrated step by step into the model [50–52]. The challenging setting of Exemplar-Free CIL (EFCIL) [8,53] imposes the additional restriction of not storing examples of previous classes throughout the learning steps. This paradigm is driven by the need to comply with memory constraints in low-resource environments [54,55] or with confidentiality requirements in privacy-sensitive applications [56]. It also aligns with the development of off-the-shelf pretrained models whose training data is not always publicly available [57]. Additionally, EFCIL algorithms are advantageous due to their relatively low update cost in terms of memory and execution time [51,55].

The transferability of the deep representation available at the beginning of the incremental learning process highly impacts the model’s accuracy [58]. Numerous works [53,59–62] trained models from scratch on the first subset of classes. Alternatively, using pretrained models as initial representations recently gained momentum [63,64]. Transformer-based models trained in a self-supervised way [6,57,65] receive growing interest, including in EFCIL [8,66,67]. Another emerging trend in EFCIL is the use of synthetic data generated by a pretrained diffusion model [68,69] to





improve model transferability. The works of [70–72] propose such approaches for CIL but do not address their possible interaction with diverse EFCIL algorithms.

4.2.1.2. Methodology Notations. We remind the common EFCIL paradigm, in which pretraining precedes continual learning [66, 73–76]. We consider a dataset $\mathcal{D} = D_1 \cup D_2 \cup \dots \cup D_T$ and a sequential learning process composed of T non-overlapping steps s_1, s_2, \dots, s_T . Each subset D_i comprises a set of classes C_i , such that each sample in D_i belongs to a class from C_i , and each class only appears in a single data subset. A step s_i consists of learning a model \mathcal{M}_i from the labeled samples from D_i , aiming to recognize all the classes from $C_1 \cup C_2 \dots \cup C_i$.

Pretraining. EFCIL works using pretraining [51, 66, 74, 75] assume that the initial model \mathcal{M}_1 is trained without computational or memory constraints, which may however apply during the incremental steps. Under this assumption, we consider three ways of initializing \mathcal{M}_1 . Either it inherits the weights of a network pretrained on an external dataset, or it is trained on the initial data subset D_1 (“init”) or on $D_1 \cup \tilde{D}_1$ where \tilde{D}_1 is a synthetic dataset (“init+gen”). We explain how to generate \tilde{D}_1 below.

Model update. At step s_i , the model \mathcal{M}_i recovers the parameters from the model \mathcal{M}_{i-1} and learns from the samples of D_i using an EFCIL-algorithm-specific procedure.

Enriching the initial data subset with synthetic classes

Generation of class labels. As in [70, 71], we use the labels from C_1 to prompt a large language model (LLM) and obtain a set \tilde{C}_1 containing \tilde{N}_1 new class names from the same topic as D_1 , along with a short visual description of each class. This preliminary step allows controlling the semantic content of the synthetic dataset.

Generation of images. Then, each class is populated with images by prompting a pretrained text-to-image model with its class name c and its visual description d . As reported in [77], associating class names with a description produces better image diversity and helps disambiguation. We prompt the model with n random seeds to obtain n synthetic images, where n is the maximum number of images per class in D_1 .

4.2.2. Comparison to State-of-the-art Methods

4.2.2.1. Experimental Settings

Datasets. We consider three datasets of a thousand classes each, sampled from Casia [78] (faces), Google Landmarks v2 [79] (human-made and natural landmarks), and iNaturalist 2018 [80] (natural species). In our experiments, we named these sampled datasets Casia, Landmarks, and Inat, respectively.

CIL scenarios. Datasets are randomly split into either 100 initial classes and nine sets of 100 classes (denoted b100t9) [81], or 500 initial classes and ten sets of 50 classes (b500t10) [82].

EFCIL algorithms. We experiment with four transfer-based EFCIL algorithms, namely the Nearest Class Mean classifier (NCM) [83, 84], DSLDA [66], FeTrIL [85] and FeCAM [8]. These algorithms offer competitive accuracy at a relatively low computational cost since they only update a classifier to learn new classes. Each method stores at least the class prototypes from the classes seen so far, defined as the mean of the embedding vectors from a given class.

Initial training. EFCIL algorithms benefit from feature extractors offering diverse and transferable representations. Recent works [58, 64] report accuracy gains when using models pretrained on large datasets compared to training from scratch on the initial subset of data. For a given data stream, we compare such pretrained models with models trained using vanilla supervised learning (SL) or with a more elaborate supervised learning scheme, namely tReX [86], with or without synthetic data augmentation.





Synthetic augmentation. We augment the initial dataset D_1 with up to 1,000 synthetic classes. Inat and Landmark are enriched using the LLaMa-v2-13b-chat model [87] to obtain class names and descriptions, and Stable-Diffusion-2-1-base [69] (SDv2.1) for image generation. We prompt SDv2.1 with the following pattern: “a photo of a *class, description*”, e.g. “a photo of *Vitis vinifera*, a climbing plant with purple grapes”. Since SDv2.1 is not designed for face generation, we use DCFace [88] to augment Casia.

Baselines. We consider the following pretrained models, available via PyTorch [89]: a supervised RN50 [90] trained on ImageNet-1k [91], a RN50 trained on LAION-400m [92] with CLIP [65, 93], a ViT-small [94] (ViT-S) trained on ImageNet-1k with AugReg [95], a ViT-S trained on ImageNet-21k [96] with AugReg, a ViT-S trained on LVD-142m using DINOv2 [57].

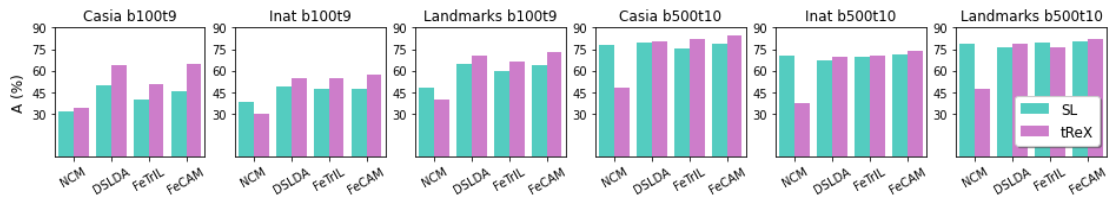


Figure 4. Comparison of SL and tReX training strategies for four EFCIL algorithms (NCM, DSLDA, FeTrIL, FeCAM). Feature extractors are trained on the initial data subset of each dataset (Casia, Inat, Landmarks), i.e. with either 100 (b100t9) or 500 classes (b500t10).

4.2.2.2. Results

How much does the supervised strategy matter in EFCIL? In Figure 4, we compare models trained on the initial classes using either SL or tReX strategies. Results show that tReX improves accuracy for most EFCIL scenarios and algorithms. More precisely, tReX tends to both increase the initial accuracy and decrease forgetting of EFCIL algorithms, indicating an improved transferability of the obtained representations. The tReX data augmentation strategy is particularly effective in diversifying a small initial subset in the b100t9 scenario. It produces better representations for the SVC, LDA, and Mahalanobis-based classifiers of FeTrIL, DSLDA, and FeCAM, respectively, but not for the cosine distance of NCM. This could be explained by a different distribution of the data around their class prototypes in the latent space.

Can supervised learning models trained on initial (augmented) classes perform better for EFCIL than models trained on larger datasets? Recent continual learning works [8, 83, 97] argue that strong pretrained models outperform models trained with initial data. We present more nuanced results in Figure 5. In four out of six cases, models trained on a mixture of initial data and synthetic data from the domain of interest are superior to models pretrained on larger datasets. We find that a larger dataset generally improves performance for models trained with initial data but does not scale alike for pretraining datasets presenting a significant domain shift. These highlight the prevalent role of the domain shift between the initial training set and the incremental tasks on EFCIL accuracy. For Casia, we observe that the baseline models are ineffective. This is coherent with the fact that none of their pretraining datasets specifically cover human faces. Our evaluation differs from those reported in [8, 83, 97] by an increased diversity of the test datasets in regard to the domains covered by pretrained models. Our results highlight the importance of using challenging datasets for EFCIL evaluation.

Is the initial accuracy a reliable predictor of the incremental accuracy? We present the relation between initial and incremental accuracy for all data-algorithm combinations in Figure 6. We compare linear regressions computed on the results of (i) EFCIL experiments with pretrained



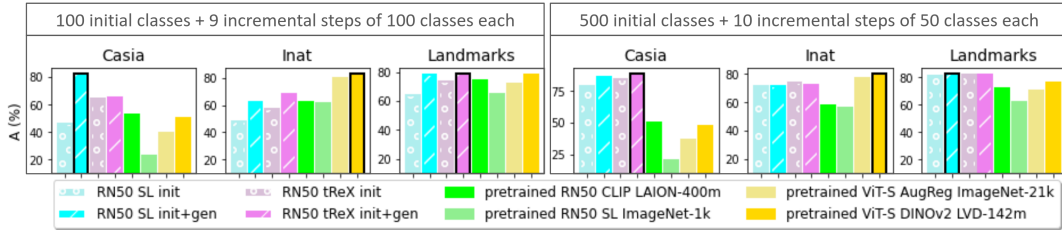


Figure 5. Average incremental accuracy of FeCAM [8] for various feature extractors: Res-net50 (RN50) model trained using vanilla supervised learning (SL) or tReX on the initial classes (init) and on the initial subset of classes augmented by 1000 classes generated using Stable Diffusion v2.1. (init+gen), supervised RN50 trained on ImageNet-1k, RN50 trained with CLIP on LAION-400m, ViT-S trained with DINOv2 on LVD-142m, supervised ViT-S trained on ImageNet-21k with AugReg. The best results are highlighted in black. Supervised learning on the initial data subset outperforms strong pretraining in four cases out of six.

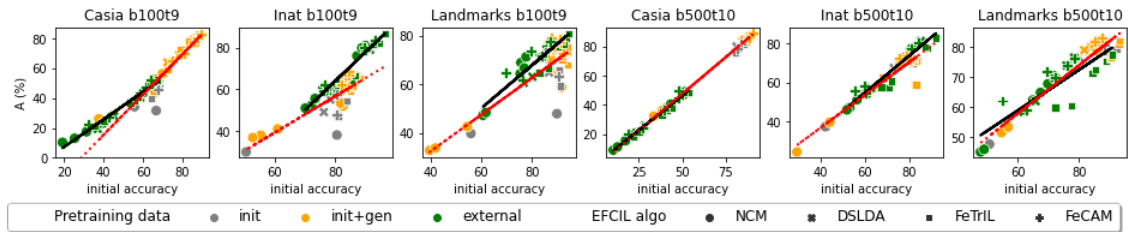


Figure 6. Average incremental accuracy as a function of the initial accuracy. Linear regressions fitted on the average incremental accuracies of four EFCIL algorithms obtained with models pretrained on large external datasets (full, black line) or the first subset of classes with or without synthetic augmentation (dashed, red line).

baselines and (ii) EFCIL experiments with feature extractors trained on the initial classes. We obtain correlation coefficients above 0.90 for pretrained models and above 0.65 for models trained with initial (augmented) data. For models trained with SL on initial data, the rank of the initial data matrix correlates slightly better with the average incremental accuracy than the initial accuracy, i.e. it informs better on model transferability. So, when choosing between several feature extractors and based on the initial data, we recommend taking into account both the initial accuracy and the rank of the initial feature matrix.

4.2.3. Conclusion

Through an extensive experimental study, we analyzed the impact of pretraining choices on the performance of EFCIL algorithms. Our findings indicate that models trained on large-scale datasets are not always the best choice for initializing incremental learning of a classification problem. In the case of visual domains that are well represented in pretraining data sets, pretrained models demonstrate high and stable accuracy, as expected. However, for domains less well covered by the pretraining data, we find that a supervised ResNet50 network trained on initial data competes with a ViT-small network trained on LVD-142m with DINOv2 and a ResNet50 network trained on LAION-400m with CLIP. We find that adding synthetic classes is usually helpful for training a model with a supervised objective, especially when the initial dataset is small. However, the accuracy gain depends on the visual and semantic coherence of the generated images. Based on the reported findings and contrary to common practice [8, 81, 83, 85], we advocate for using large datasets from a variety of domains in continual learning evaluation. This is particularly important when using models pretrained with large datasets whose content is likely to overlap with that of



the evaluation datasets.

4.2.4. Relevant publications

- E. Feillet, A. Popescu, C. Hudelot. A reality check on pretraining for exemplar-free class-incremental learning. Under review.

4.2.5. Relevance to AI4media use cases and media industry applications

The proposed analysis gives insights into the optimization of pre-training-based continual learning methods. These methods are particularly suited for analyzing visual content in the news since they can be adapted quickly. They can be coupled with the generalized category discovery contribution (Section 4.1) to obtain a full pipeline for handling novelty in news streams. The pipeline can support the following use cases: up-to-date tagging of visual content in the new (2B1), comprehensive indexing of archives for improved exploration (3A3), and reorganization of visual content (7A3).

4.3. Stationary Representations: Optimally Approximating Compatibility and Implications for Improved Model Replacements

Contributing partner: UNIFI

4.3.1. Introduction and methodology

By learning powerful internal feature representations from data, Deep Neural Networks (DNNs) [98–101] have made tremendous progress in some of the most challenging search tasks such as face recognition [102–106], person re-identification [107–109], image retrieval [110–112] and this significance also extends to a variety of other data modalities [113, 114]. Although all of the works mentioned above have focused on learning feature representations from *static* and, more recently, *dynamic* datasets [115–118], the now-standard practice is downloading and fine-tuning representations from models pre-trained elsewhere [87, 119]. These “third-party” pre-trained models often incorporate new data, utilize alternative architectures, adopt different loss functions or more in general provide novel methodologies. Whether applied individually or combined, these advancements aim to encapsulate the field’s rapid progress within a single unified model [120]. This greatly facilitates the exploitation of internally learned semantic representations, particularly as models, datasets, and computational infrastructure continue to expand in size, complexity, and cost [121, 122].

The challenge of fully exploiting such standard practice in retrieval/search systems has to deal with the underlying problem of *compatible learning* [123–125]. That is the desire to align the representation of different models trained with different data, initialization seeds, loss functions, or alternative architectures—either individually or in combination. In such applications, maintaining alignment is crucial to minimize the need for repeated reprocessing of gallery images for feature extraction each time a new pre-trained model becomes available [120]. Reprocessing is not only computationally intensive but may also be unsustainable for extensive gallery sets [121, 122, 126] or unfeasible if the original images are no longer accessible due to privacy concerns [127]. This holds across various typical galleries: social networks update millions of images every month, while in robotics and automotive domains, the update rate can be as rapid as hundreds of images every second. Similarly, in textual domains, books can be structured into chapters, paragraphs, and sentences, enabling the capture of semantic relationships between these segments. While a similar organizational principle can be structured for the web with LLMs [114, 128], the challenge lies in the impracticality



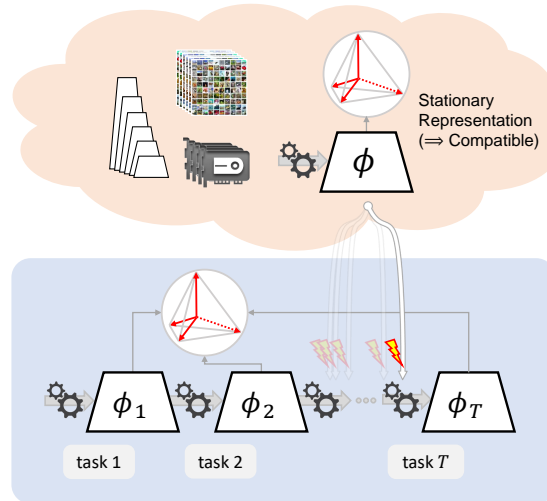


Figure 7. Improved Asynchronous Model Compatible Lifelong Learning Representation (IAM-CL²R pronounced “I am clear”). In the process of lifelong learning, a model is sequentially fine-tuned and asynchronously replaced with improved third-party models that are pre-trained externally. Stationary representations ensure seamless retrieval services and better performance, without the need to reprocess gallery images.

of reprocessing such extensive content with each advancement in representation models. Although recent research has shown the effectiveness of compatible representation learning [123–125, 129–137], there is still a lack of comprehensive theoretical understanding about compatibility.

This work introduces a theorem that demonstrates how the stationary representations proposed in [138, 139] optimally approximate compatibility according to the two inequality constraints of its formal definition as provided in [123]. This not only establishes a solid foundation for future works, but also presents implications that can be exploited in fine-tuning third-party models without the need of reprocessing gallery images. Specifically, we show that a continuously fine-tuned model can be asynchronously replaced by downloading a higher-performing, pre-trained model from an external source. Due to stationarity (and therefore optimal compatibility), such a replacement provides seamless retrieval services with improved performance, eliminating the need for image gallery reprocessing. We refer to this scenario as Improved Asynchronous Model Compatible Lifelong Learning Representation (IAM-CL²R pronounced “I am clear”). Fig. 7 illustrates the relationship between sequential fine-tuning and model replacement. Furthermore, as will be elaborated in the related work section, our foundation draws connections with the Neural Collapse phenomenon [140] and its associated theory.

Our second contribution is related to a specific challenge that arises: the tendency of the old and the new replaced models to align at their first-order statistics, an inherent property of stationary representation. Consequently, cross-entropy based prediction errors alone, when fine-tuning the representation, may not fully capture higher-order dependencies. To address this issue while preserving compatibility, we show that learning stationary representations using a convex combination of the cross-entropy loss and the infoNCE loss [141] is equivalent to training under one of the compatibility inequality constraints in [123]. This combined loss, termed Higher-Order Compatibility (HOC), distinguishes itself from the use of cross-entropy alone by capturing higher-order dependencies and optimally approximating compatibility.



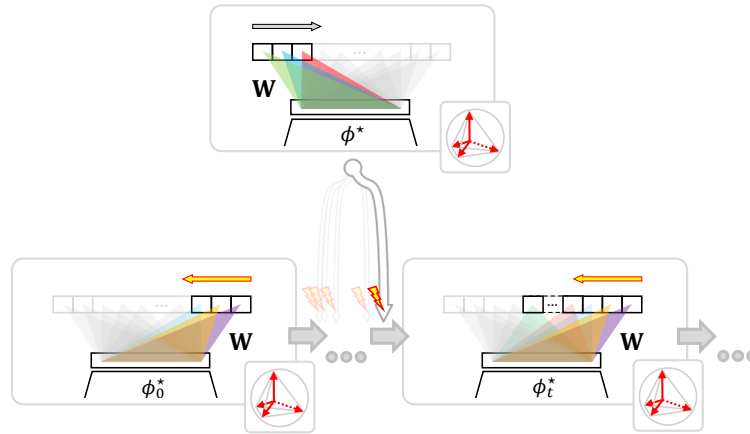


Figure 8. Implementation details of fine-tuning and model replacement in IAM-CL²R. In the third-party pre-trained model ϕ^* , class labels are assigned from left to right. Conversely, in the current fine-tuned model ϕ_t^* , class labels are assigned from right to left. This simple convention maintains distinct class labels for pre-training and fine-tuning, preventing any overlap between them. The d -Simplex fixed matrix \mathbf{W} can be seen as a common interface between the two learning processes.

4.3.2. Experimental results

4.3.2.1. Pre-trained Models. We pre-train our models in a supervised manner using the ImageNet32 [142]. Three distinct models are pre-trained on ImageNet32 with 100, 300, and 600 classes. The model trained with 100 classes is used to initialize the model before fine-tuning on the sequence of tasks. The other two models are used to simulate the practice of downloading and fine-tuning pre-trained models and serve as third-party models that will replace the current one undergoing fine-tuning.

4.3.2.2. Fine-tuning. We replicate the fact that dataset size for training third-party models is typically significantly larger than the dataset size used for fine-tuning [143]. According to this, pre-trained models are fine-tuned with a reduced version of CIFAR100 [144] denoted in this paper as CIFAR100R.

We considered two distinct task sequences consisting of 7 and 31 tasks each. We fine-tune the pre-trained model with an initial task comprising 10 classes. Subsequently, for the sequences of 7 and 31 tasks, the respective tasks contain 15 and 3 classes each. The fine-tuning process incorporates incoming task data, consisting of 300 images per class, and utilizes an episodic memory that stores 20 images from each class of previous tasks.

4.3.2.3. Model Replacement. In our experiments, we verify the impact of replacing the current fine-tuned model with two improved models pre-trained elsewhere. The two replacements occur while fine-tuning on CIFAR100R: at the third and fifth tasks in the shorter sequence, and at the eleventh and twenty-first tasks in the longer sequence. We also consider the challenging scenario of improved model replacement considering more sophisticated network architectures.

The d -Simplex fixed classifier is pre-allocated with a number of classes K , ensuring enough space to accommodate future classes for both pre-training and fine-tuning. Class assignments for pre-training are made from left to right, and for fine-tuning, from right to left. This straightforward convention is used to ensure that classes assigned for pre-training and fine-tuning remain distinct,



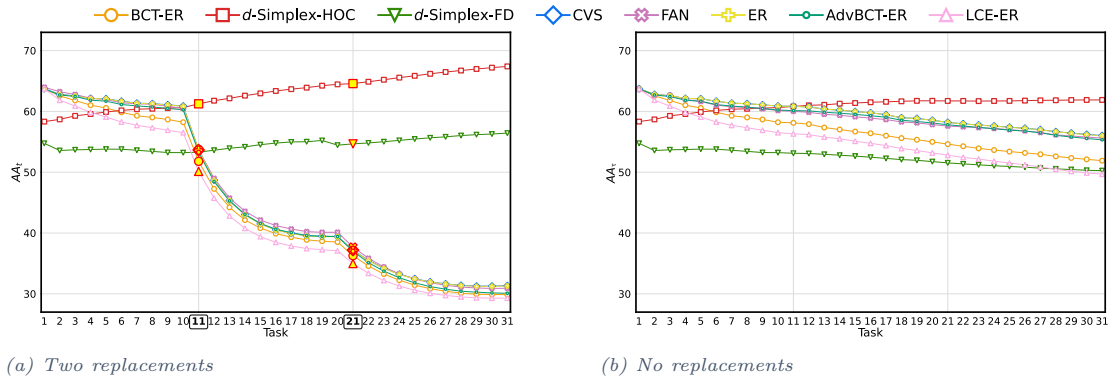


Figure 9. Average multi-model Accuracy (AA_t) evaluated across 31 tasks using CIFAR100R/10, showing: (a) model replacements at tasks 11 and 21 (indicated by yellow markers); (b) no model replacement.

without overlap, as illustrated in Fig. 8. Other non overlapping assignment methods could also be used.

4.3.2.4. Network Architectures. We use ResNet18 [26] as network architecture. In the scenario using more sophisticated network architectures, we initially replace ResNet18 with SENet18 [145], followed by a subsequent replacement with a RegNetY_400MF [146].

4.3.2.5. Hyper-parameters. The ResNet18, SENet18, and RegNetY_400MF models were pre-trained on ImageNet32 using the following hyper-parameters: 300 epochs, a batch size of 128, and an initial SGD optimizer learning rate of 0.1, which was adjusted using a Cosine Annealing schedule. For each task used for fine-tuning, the model is trained for 70 epochs with a batch size of 128, starting with a learning rate of 0.001 that was reduced by a factor of 10 after the 50th and 64th epochs. The d -Simplex was pre-allocated with $K = 1024$ classes (i.e., $d = K - 1$).

4.3.2.6. Performance Evaluation. The evaluation focuses on the open-set recognition task, in which separated datasets for training and evaluation are required. The standard 1:N search protocol, applicable to re-identification and similar tasks [123], is employed in the evaluation. To ensure strict separation between datasets, the CIFAR10 dataset is utilized for evaluation during fine-tuning with CIFAR100R. Specifically, the test set of CIFAR10, comprising 10,000 images, is used as the gallery set, while its training set of 50,000 images serves as the query set.

Following [123] and [125], we measure performance progression across the two sequences of tasks using two established metrics: Average Compatibility (AC) and Average multi-model Accuracy (referred shortly as to AA_t). The metric AC quantifies the extent of compatibility across all possible pairs of model combinations by providing a normalized count of times in which compatibility is achieved. Conversely, AA_t calculates the mean accuracy across all combinations of the previously learned models until task t , providing an overall measure of accuracy.

4.3.2.7. Comparison to State-of-the-art Methods

We performed a comparative analysis of d -Simplex-HOC against FAN [133], CVS [134], d -Simplex-FD [135], and the lifelong adapted versions of BCT [123] (BCT-ER), LCE [124] (LCE-ER), and AdvBCT [132] (AdvBCT-ER). The experiments also incorporate a baseline method, Experience Replay (ER), in which the model is fine-tuned using cross-entropy loss on data of the new task and

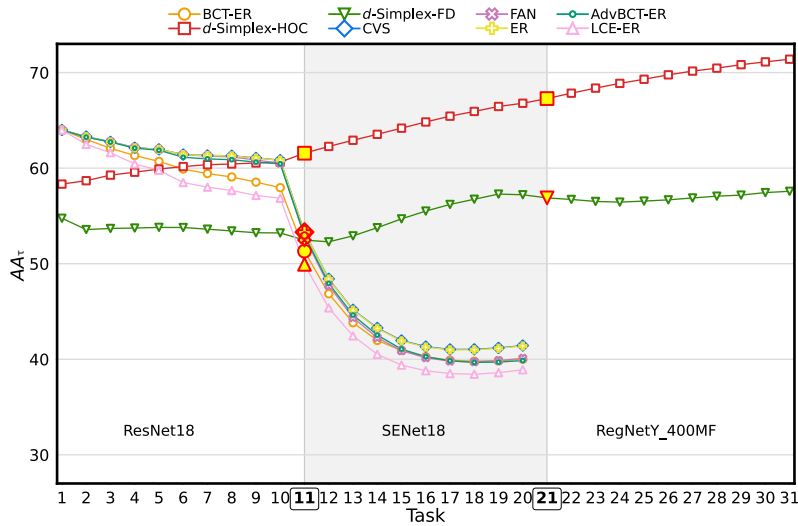


Figure 10. Plots of Average multi-model Accuracy (AA_t) for 31 tasks on CIFAR100R/10, showing the impact of model replacements with different network architectures at tasks 11 and 21.

an episodic memory. Ablation studies of IAM-CL²R with the d -Simplex-HOC are provided in our paper [147].

The comparison provides insights into the performance benefits that can be obtained by replacing models when representations are trained in a compatible manner. The d -Simplex-HOC effectively incorporates improvements from model replacements, showing increased performance compared to the case without model replacement, as indicated in Fig. 9b. The d -Simplex-FD demonstrates a similar capability, though to a reduced extent. The other methods have a clear performance decay after model replacements and end up with a worse performance than the case without replacement. This can be attributed to the fact that after replacement, fine-tuning is applied to a model obtained by retraining the network from scratch, leading to an entirely different representation.

To provide a full evaluation of compatibility, the AA_t of Fig. 9 is complemented with the Average Compatibility AC . We also report the Average multi-model Accuracy AA_7 and AA_{31} for methods compared at the end of the 7-th and 31-th task, respectively. It is observed that, in both instances, all models—with the exception of d -Simplex-HOC—fail to achieve significant compatibility performance.

Fig. 10 shows the performance of the evaluated methods when the original ResNet18 is replaced first by a SENet18 and then by a more expressive RegNetY_400MF. It is observed that the change of network architecture not only does not adversely affect compatibility in the d -Simplex-HOC but takes advantage of their more expressive representation power. In particular, results show that d -Simplex-HOC improves performance gradually with each model replacement. This is in contrast to d -Simplex-FD, which does not demonstrate the same trends leading to a plateau around the 20-th task. Given the different feature sizes before and after the second model replacement with the RegNetY_400MF architecture—512 and 384, respectively—all methods except d -Simplex-HOC and d -Simplex-FD require non-trivial extensions to adapt to the changed feature size. According to this, for these methods, evaluation cannot be reported.





4.3.3. Conclusion

The main contributions of this work are:

- We theoretically show that stationary representations learned according to d -Simplex fixed classifiers optimally approximate compatibility (proof details in the original paper [147]).
- Practical Implications: in the process of lifelong learning the model is sequentially fine-tuned and occasionally replaced with improved third-party models that are pre-trained externally (IAM-CL²R scenario). This allows seamless retrieval service, improved performance and no gallery reprocessing.
- We introduce the Higher-Order Compatibility (HOC) loss that captures higher-order dependencies and optimally approximates compatibility.

4.3.4. Relevant publications

- Niccolò Biondi, Federico Pernici, Simone Ricci, and Alberto Del Bimbo. “Stationary Representations: Optimally Approximating Compatibility and Implications for Improved Model Replacements.” In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 28793-28804. 2024. <https://zenodo.org/records/11622697>.

4.3.5. Relevant software and/or external resources

- The PyTorch implementation of our work “Stationary Representations: Optimally Approximating Compatibility and Implications for Improved Model Replacements” can be found in: <https://github.com/miccunifi/iamcl2r/>.

4.3.6. Relevance to AI4media use cases and media industry applications

We have addressed the challenge of compatible learning in the context of fine-tuning third-party pre-trained models, focusing on maintaining alignment between different model representations to avoid the costly process of reprocessing extensive galleries. To achieve this, we introduced the Improved Asynchronous Model Compatible Lifelong Learning Representation (IAM-CL²R), a framework that leverages stationary representations to seamlessly replace fine-tuned models with higher-performing pre-trained models without reprocessing. This approach is directly applicable to various use cases such as (a) automated image and video management, particularly 3A3-11 visual indexing and search, and (b) (Re)organisation of visual content 7A. Furthermore, this method can be extended to other applications, such as 4C3 audio analysis and 4C5 multi-modal analysis, demonstrating its versatility and efficiency across different domains.

4.4. Collaborative Knowledge Distillation via a Learning-by-Education Node Community

Contributing partner: AUTH

4.4.1. Introduction and methodology

Deep Neural Networks (DNNs) have advanced over the past decade partially by modeling abstractly various human brain structure functions and capabilities [148, 149]. One particularly alluring sociobiological aspect that can be integrated into DNNs concerns transactional knowledge exchanges





between humans. This is a fundamental dimension of human learning that has significantly shaped human communities. In particular, human learning emerges from collaboration and knowledge exchange among individuals, primarily through formal and informal education. Organized human societies have relied on class education and teacher-student learning for thousands of years, building entire education systems on this concept for knowledge transfer between generations.

Existing frameworks for teacher-student communities are commonly employed in Reinforcement Learning (RL) [150–153]. However, few methods have been proposed for multi-node supervised or semi-supervised teacher-student learning [154, 155]. Certain frameworks exploit the diverse knowledge of multiple DNNs to augment the community’s collective knowledge, a strategy called Collaborative Knowledge Distillation (CKD) [156–158].

The limitations of existing CKD frameworks are multiple and need proper solutions. Firstly, most if the majority of them do not support the on-line acquisition of knowledge regarding new *tasks* by pretrained DNNs. *Task*, in this sense, comprises DNN training and inference on a set of semantic classes, where training has been performed using appropriate data. Thus, each DNN node knowledge is statically limited to a specific collection of tasks defined before framework deployment (during original training). Task in this sense is a set of semantic classes on which training has been performed using appropriate data. In the only relevant framework that does use Continual Learning (CL) for dynamic acquisition of novel tasks, knowledge transfer is limited to distillation of a trained GAN by a VAE and can happen only once [154]. Continual Learning (CL) and Lifelong Learning refer to the same scientific field in modern literature. Secondly, the non-task-agnostic nature of these frameworks limits their potential. The DNN students in such frameworks acquire knowledge only when being aware of the task boundaries, i.e., when the switch of training data from an old task to a new one is fully defined and known [155–158]. A task boundary is defined as the last picture of the last task learned before the the first picture of the new task to be learned for image classification problems. Finally, their performance is significantly decreased when only a data subset is available during training, which is closer to a real-world scenario. Consequently, they cannot operate in a dynamic realistic environment, where raw unlabelled data is the only input.

To address these limitations, we introduce the *Learning-by-Education Node Community* (LENC) framework, i.e., a CKD environment consisting in a community of DNN nodes that can dynamically act either as teachers or students in different, autonomously initiated on-line knowledge DNN exchanges of a transactional nature. LENC addresses the issue of task-agnostic CL by equipping each participating DNN node with knowledge self-assessment capabilities. These features simulate an environment of human nodes, which are aware of their knowledge and seek expert advice to expand it. When a DNN node encounters a task, for which it has not been trained for, it can assume the role of a student DNN. By sending the new training data to other DNN nodes in the community, the student seeks guidance from those nodes, who have already acquired knowledge in the given task and can become teacher DNNs. This knowledge transfer is crucial as it allows the student DNN node to leverage the collective expertise of the DNN community, for accelerating its learning process and enhancing its performance in the specific task. Overall, such a knowledge exchange framework in a DNN node community fosters a collaborative learning environment. The flexibly decided teacher/student role, as well as the task-agnostic nature of the framework, simulate a community of human nodes, where an unknown data stimulus coming from the environment triggers the urge to attain relevant knowledge from other humans, e.g., from a specialized teacher. Overall, the LENC framework is the first CKD approach enabling task-agnostic CL, which is a key characteristic of human communities.

Specific algorithmic components of the LENC framework are: a) an *Out-Of-Distribution* (OOD) detection algorithm (e.g., the simple Likelihood Regret (LR) [159] can be adopted), for DNN node task-agnostic knowledge self-assessment, b) a knowledge transfer mechanism (e.g., standard neural distillation [160], more advanced distillation variants, or even simple parameter copy) and c) a





CL algorithm (e.g., EWC [161] can be adopted), for ensuring the retention of previously acquired knowledge in each node. This paper details a LENC node architecture that integrates all of the above components into an interactive LENC community participant, as well as the transactional peer-to-peer knowledge exchanges within this task-agnostic CKD community. Experimental evaluation on a proof-of-concept implementation demonstrates the LENC functionalities and benefits across multiple scenarios. The conducted experiments showcase the LENC framework’s ability to gradually maximize the average test accuracy of the community of interacting DNN nodes in image classification problems¹, as well as its ability to learn on-line from small batches of unlabelled data. In fact, it overcomes all competing existing methods in such settings.

The proposed LENC framework defines the protocol for LENC nodes to learn tasks from other peer LENC nodes that participate in a LENC community. Consequently, every deployed LENC node can assess on-line its knowledge of incoming external test inference data points. In case of ignorance, it can decide to transfer it to other LENC nodes via teacher-student interaction to a) identify potential teachers and b) learn from them. The integration of CL and CKD, combined with the ability of each node to self-assess its knowledge, emulates a human community where all nodes cooperate to broaden their knowledge on multiple tasks.

4.4.1.1. LENC Node Architecture Figure 11 illustrates the structure of a LENC node. Each LENC node contains a Feature Module (FM), namely a DNN model f , parameterized by \mathbf{w}_s . f is assumed to be shared across T tasks, $T \geq 0$, on which it has been trained using the appropriate training datasets $\mathcal{D}_\tau, \tau = 1, \dots, T$. The shared FM culminates in T individual Decision Heads (DHs) $\hat{f}_\tau, \tau = 1, \dots, T$, parameterized by \mathbf{w}_τ , so that the decision for each task is taken by the function $y_\tau = \hat{f}_\tau(f(\mathbf{x}; \mathbf{w}_s); \mathbf{w}_\tau), \tau = 1, \dots, T$ for an input test data point \mathbf{x} . This structure allows the deployed LENC node to support multiple tasks, which potentially have a different number of semantic classes, using a single DNN. Optionally, the known DH classification accuracy a_τ in the test set of \mathcal{D}_τ , as measured before deployment using any task-appropriate evaluation metric, can be stored along with \hat{f}_τ .

Each node also contains T Knowledge Self-Assessment (KSA) modules g_1, \dots, g_T , which assess if incoming test data points match the distribution of the corresponding original training datasets \mathcal{D}_τ . All nodes autonomously decide to act as either student or teacher DNNs, depending on the output of their KSA module. Furthermore, each LENC node contains an Interactions Manager (IM) and a set of node Interaction Rules (IRs) that specify its exchanges with other nodes to allow teacher-student interactions. The various LENC node modules are subsequently detailed.

If a LENC node KSA modules indicate that it does not have sufficient knowledge of current test inputs, this particular node can temporarily become a student and trigger a search for one or more teacher nodes in the LENC community. Once they are found, the student may learn from them using IRs. This process, when triggered by a specific incoming test data stream that proves to be unknown or not well-known, is called an *education cycle*.

The τ -th KSA module of a LENC node consists of an OOD detector $g_\tau(\mathbf{x}) : \mathcal{X}_\tau \rightarrow \{0, 1\}$ corresponding to the τ -th task the node has been trained on, where $\tau = 1, \dots, T$. It classifies a stream of incoming test data points $\mathbf{x} \in \mathcal{X}_\tau$ as either ID or OOD, thus assessing the relevant knowledge of the FM. Thus, the T node’s KSA modules are tailored to the T known training datasets \mathcal{D}_τ that the node has encountered before its deployment. In short, the KSA modules provide each node with task-agnostic, on-line knowledge self-assessment capabilities, allowing it to assess on its own the DH most relevant to the current test data point \mathbf{x} and also to identify the most knowledgeable DH. As a result, during FM inference on \mathbf{x} , the node automatically

¹The proof-of-concept experiments in this paper are limited to image classification, but there is no inherent reason to exclude other machine learning problems such as regression, object detection, semantic segmentation, etc.



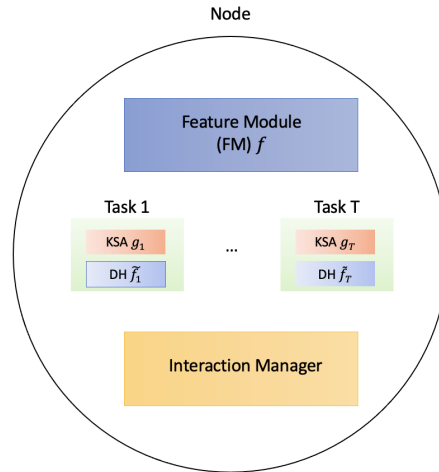


Figure 11. LENC node architecture.

detects and activates/utilizes the j -th DH $\tilde{f}_j, j = 1, \dots, T$ as the one most relevant to \mathbf{x} , since $j = \arg \min(g_1, \dots, g_T)$ is the index of the supported task with known training data most similar to the ones found in \mathcal{X} . This implies that, assuming the decision for the τ -th task is taken among c_τ classes, then the node will finally predict $y_j = \tilde{f}_j(f(\mathbf{x}; \mathbf{w}_s); \mathbf{w}_j)$.

Given an unlabelled data point \mathbf{x} incoming from the external world, each of the node's KSA modules may output one of three potential verdicts: i) the task is not known at all by the FM (non-expert), ii) the task is known, but the node's relevant knowledge is limited (non-expert), or iii) the task is well-known by the node's FM (expert). In the first and the second case, an education cycle will be triggered as a response, but in the limited existing knowledge scenario, the existing DH will be employed for receiving education, instead of appending a new DH. In the third case, no education is kick-started and the node only infers on the incoming data stream. The verdict depends on the KSA's internally computed OOD score for the current stream and on two relevant manually prespecified and task-specific thresholds: δ and ϵ . Thus, the first, second, or third case is activated if $g_j(\mathbf{x}) > \delta$, $\epsilon < g_j(\mathbf{x}) < \delta$, or $g_j(\mathbf{x}) < \epsilon$, respectively.

The LENC node Interaction Rules (IRs) are defined by the LENC framework and serve three basic LENC node interaction functions. The first one is that they specify the interaction between a deployed LENC node and the external environment, which can constantly fetch unlabeled test input data points for analysis in the form of a data stream $\mathcal{D}^s = \{\mathbf{x}_i\}$, where $i = 0, \dots, M^s$ and $M^s > 0$ is the total number of currently received data points. If the node's KSA modules respond that the FM does not know the current test data distribution well or at all, and is therefore a non-expert, an education cycle is triggered: DNN experts are automatically searched for within the LENC node community, in order to serve as teachers.

The second function specified by IRs is the transmission of the data stream \mathcal{D}^s to the other LENC community participants and the reception of their responses: $[q_1, \dots, q_{N-1}]$ for N nodes within the node community. The response q_n from the n -th node, $1 \leq n \leq N - 1$, is either (0), if none of that node's KSA modules is aware of the distribution of \mathcal{D}^s , or a non-zero numerical score that denotes how well the n -th node and its most suitable DH know the distribution of \mathcal{D}^s . Assuming that the KSA modules of the n -th node indicate that the latter's j -th DH/supported task is the most suitable to \mathcal{D}^s , different policies can be alternatively employed for computing q_n





and, therefore, for *teacher selection* at each knowledge transaction. In LENC, a policy where q_n is a scalar measure of the *disagreement* between the current student and the n -th node was selected. To this end, a function of the *churn* metric [162] can be used: the accuracy of student predictions given as input \mathcal{D}^s , using the corresponding predictions of the n -th node as pseudo-ground-truth.

The teacher with the best response ($t = \arg \max(q_1, \dots, q_{N-1})$) is then selected for transferring its knowledge of \mathcal{D}^s to the querying student node. In the case of the Disagreement Policy, this will lead to learning from the teacher from which the student diverges the most.

Third, the IRs are responsible for specifying the actual teacher-student knowledge exchanges. Various policies can be alternatively employed, resulting in messages of different content.

4.4.1.2. Learning a Novel Task Similarly to human societies, the external world constantly provides novel data stimuli for classification by one or more of the deployed individual LENC nodes. For each such test input data stream \mathcal{D}^s , the first question that needs to be answered is if the triggered LENC node is knowledgeable of it. Such a question is answered using this node's KSA modules. If the task is judged to be known and the LENC node is considered to be an expert, no action is taken by LENC and the LENC node proceeds to infer its own predictions for \mathcal{D}^s . In any other case, an education cycle is triggered: the node temporarily assumes a student role and sends \mathcal{D}^s to other active LENC nodes within the community. Each of the other $N - 1$ LENC nodes receives \mathcal{D}^s and forwards it through its own KSA modules. Then, it replies to the student with q_n , $1 \leq n \leq N - 1$, which specifies whether it knows the distribution of \mathcal{D}^s data or not. From this point on, in the context of this particular knowledge transaction, any node with non-zero q_n is considered a potential teacher. In the implemented LENC system, the student node automatically selects as actual teacher the node with the highest q_n score, although in principle knowledge transfer from multiple teachers can also be supported.

Assuming that the student node already knows $T^s \geq 0$ tasks, using T^s existing DHs $\tilde{f}_1^s, \dots, \tilde{f}_{T^s}^s$, it will now learn the new task driven by the dataset \mathcal{D}^s , using the selected teacher LENC node knowledge. If the student LENC node has limited prior knowledge of the task, the existing DH for the current task will be enhanced via knowledge transfer from the teacher. In contrast, if the student has no prior knowledge of the task, a new DH will be appended. The LENC framework offers a KD policy for teacher-student knowledge transfer, which can be combined with a CL method, so that the student retains its previously acquired knowledge and does not experience catastrophic forgetting, just as in human society. Although multiple *knowledge transfer policies* can be alternatively supported, this paper uses Knowledge Distillation (KD) from output activations [160].

The student receives from the selected teacher node the latter's soft-output activations $\tilde{\mathbf{a}}_j^t = \tilde{f}_j^t(f^t(\mathbf{x}^t; \mathbf{w}_s); \mathbf{w}_j)$. Note that, obviously, for classification problems the teacher's respective prediction is $\tilde{y}_j^t = \arg \max(\tilde{\mathbf{a}}_j^t)$. The student is subsequently trained using \mathcal{D}^s and a KD loss (e.g., the one from [160], for simple classification problems). CL is utilized if the student is not previously untrained. An example for classification is the following one:

$$\mathcal{L}_t = \begin{cases} \mathcal{L}_c + \beta KL(\tilde{\mathbf{a}}_j^t, \tilde{\mathbf{a}}^s), & T^s \geq 1 \\ \beta KL(\tilde{\mathbf{a}}_j^t, \tilde{\mathbf{a}}^s), & T^s = 0, \end{cases} \quad (1)$$

where KL denotes the Kullback-Leibler (KL) divergence, β is a distillation hyperparameter to control the relative influence of the distillation loss, and \mathcal{L}_c is the EWC CL regularizer.





4.4.2. Experimental results

Existing CKD literature has several limitations, as it leaves important issues unaddressed. Most CKD methods do not consider the potential help of potential teachers who have acquired their knowledge asynchronously with their peers (other nodes), or cases where the external environment provides only a few unlabelled data points to the node community. Such scenarios resemble more closely human learning in real communities. To evaluate the effectiveness of the LENC framework in similar setups, it was experimentally compared against existing CKD methods in scenarios involving on-line unlabelled CKD from a pretrained teacher. This section displays the LENC framework ability to reach state-of-the-art performance on on-line unlabeled CKD. However, the LENC framework utilizes a wide spectrum of knowledge transfer policies and its nodes learn tasks continually without forgetting in a task-agnostic manner. Experiments showcasing the LENC framework abilities are detailed in the related paper.

Two different sets of main experiments were performed: a) whole-image classification with untrained nodes and one expert, where no CL capabilities need to be activated, and b) whole-image classification where most nodes contain prior knowledge. In the first case, a traditional CKD setup is simulated (up to a degree), in order to facilitate comparisons against competing CKD methods. Thus there is only a single task ($T^s = 0$), with only a single node having been pretrained and able to serve as an expert teacher². In the second case, LENC is configured to run in a setup that demonstrates a fuller extent of its true capabilities. Most of the participating nodes are pretrained in different ways and LENC showcases its ability to handle on-line learning of multiple tasks on-the-fly, based on incoming unlabelled data points.

CL is employed only in the second experimental setup, while the KSA modules are utilized in both. However, in the CKD experiments the KSA modules are only used for identifying which node is a potential teacher, while in the CL experiments they additionally address task-agnostic CL by automatically identifying the task index. The baseline CL method of EWC [161] was selected for integration into the implemented LENC system, due to its combination of simplicity and good performance³. Similarly, LR [159] was selected for OOD detection within the KSA modules.

Following common CKD evaluation protocols [156, 157, 165–169], the LENC framework is evaluated on datasets CIFAR-10 (C10) and CIFAR-100 (C100), using nodes with the neural architectures ResNet [26], Wide-ResNet (WRN) [170] and VGG [171]. A pretrained ResNet-18 is employed as the only teacher, while two alternative sizes are utilized for the incoming data stream \mathcal{D}^s that originates in the external environment: 1,000 and 5,000 data points. The use of a pretrained expert excludes from the comparisons CKD methods for collaborative learning from scratch with neural branches of identical architecture [166–168]. The data points of a stream \mathcal{D}^s are randomly sampled from the teacher's actual training dataset, with 10 different \mathcal{D}^s sets constructed in this manner. Each student receives sequentially the 10 streams, with each one triggering an education cycle; although the node is no longer entirely untrained after the first cycle, it is not an expert either. To ensure collaboration among peers, the students involve experts in the teaching selection process every 2 education cycles. The competing CKD methods were adapted to distill the teacher's response, instead of training with ground-truth, in order to enable fair comparisons with LENC. Although the competing SwitOKD method [169] uses ground-truth labels to calculate the teacher's influence, it is also included in the evaluation.

The LENC community included two homogeneous students (2 ResNet-18 models) and two heterogeneous students (WRN-16-4 and VGG11). After hyperparameter search, the batch size

²However, this is not a priori known to the students. It is automatically discovered by the LENC framework, in contrast to existing CKD methods.

³These qualities of EWC underlie its continuing use as a building block by newer CL approaches [163, 164].





was set to 128 and SGD was adopted as an optimizer, with an initial learning rate of $1e-3$ and momentum of 0.9. The number of epochs for knowledge transfer was set to 100. The experiments were conducted on four NVIDIA GeForce GTX 1080 Ti GPUs.

Table 4. Comparisons of LENC with competing CKD methods, for incoming data streams \mathcal{D}^s of sizes 1,000 and 5,000. The average test accuracy (%) and deviation of the student nodes is reported.

Dataset	Students	Stream Size	DML	KDCL	SwitOKD	LENC (proposed)
C10	ResNet-18 & ResNet-18	1,000	52.20 ± 0.52	62.23 ± 0.15	56.15 ± 0.73	76.93 ± 0.71
	WRN-16-4 & VGG11		51.17 ± 0.71	62.09 ± 0.21	57.85 ± 0.80	70.16 ± 0.82
C10	ResNet-18 & ResNet-18	5,000	77.85 ± 0.31	85.76 ± 0.07	79.08 ± 0.70	86.31 ± 0.32
	WRN-16-4 & VGG11		75.56 ± 0.82	84.47 ± 0.08	78.79 ± 0.68	87.12 ± 0.24
C100	ResNet-18 & ResNet-18	1,000	9.77 ± 0.25	25.16 ± 0.12	13.71 ± 0.57	34.96 ± 0.47
	WRN-16-4 & VGG11		6.12 ± 0.38	27.59 ± 0.19	14.72 ± 0.61	29.75 ± 0.49
C100	ResNet-18 & ResNet-18	5,000	31.53 ± 0.31	58.70 ± 0.09	35.31 ± 0.29	65.02 ± 0.13
	WRN-16-4 & VGG11		8.30 ± 0.16	56.94 ± 0.12	37.27 ± 0.45	58.18 ± 0.17

Average community accuracy in the respective test set for the last education cycle, over all students and 10 independent runs, is reported in Table 4 along with the standard deviation over the different runs. As it can be seen, the LENC framework outperforms existing CKD methods when digesting unlabelled incoming streams, under the assumption that the sole expert indeed knows data similar to the incoming ones. The main reason for LENC's higher performance is the employed teacher selection policy: given the lack of ground-truth annotation, each student node picks the teacher to which it disagrees the most, to leverage diverse knowledge within the community. Instead, the adapted competing CKD methods also consider the non-expert responses of other nodes.

4.4.3. Conclusion

The main contributions of this work are:

- It introduces an on-line CKD technique that reaches state-of-the-art performance.
- It introduces a CKD framework of interconnected nodes that can continuously learn without forgetting, using peer-to-peer connections.
- The nodes can run autonomously after deployment as they can assess their knowledge and learn multiple tasks without human intervention.

4.4.4. Relevant publications

- A. Kaimakamidis, I. Mademlis, I. Pitas, Collaborative Knowledge Distillation via a Learning-by-Education Node Community, Under review.

4.4.5. Relevance to AI4media use cases and media industry applications

This work introduces the Learning-by-Education Node Community (LENC) framework, a CKD environment. It comprises a community of DNN nodes capable of dynamically switching roles between teachers and students. These roles are engaged in autonomously initiated online knowledge exchanges that are transactional in nature. Multiple media organizations that participate in a collaborative network of AI tools can benefit from the proposed method by utilizing or even





purchasing DNN knowledge within a Teacher-Student scenario. This setting provides a novel and efficient solution to data-sharing issues while at the same time enhancing memory efficiency. It finds relevance in several AI4Media use cases where advanced deep learning techniques are utilized, such as UC3 “AI in Vision - High quality Video Production and Content Automation”. It offers a novel method for knowledge sharing among DNNs, making it a valuable asset in addressing the challenges of content organization, content enhancement, and media content analysis.

4.5. Efficient DNN knowledge assessment in a multi-agent Teacher-Student DNN environment

Contributing partner: AUTH

4.5.1. Introduction and methodology

Typically, the knowledge transfer process in multi-agent systems involves defining one or more Teacher ML agents, as well as Student ML agents. In most cases, ML agents are essentially DNN models. The selection of Teacher DNN agent(s) is typically based on the number of their trainable parameters [156,172–174], as it is considered to be directly associated with the knowledge contained within a trained DNN model [175]. However, the selection of Teacher DNN agent(s) within a multi-agent system should consider more than one factor, such as the specific data domain, the ML task to be performed, as well as the overarching goals of the whole environment. If Student DNN agents decide to learn from an unsuitable model, system errors will persist and potentially magnify and their performance will be poor and possibly deteriorate. Furthermore, in scenarios where multiple DNN agents possess domain-specific knowledge for a task, selecting the appropriate Teacher DNN(s) becomes a more challenging problem. Therefore, it becomes imperative to establish a robust and efficient mechanism for assessing DNN knowledge in multi-agent environments. To achieve this, defining and quantifying the knowledge of DNN agents is essential.

In this work, Teacher-Student DNN environments that connect ML agents are examined, where Student DNN agents derive and enhance their capabilities by learning from Teacher DNNs. An efficient self-assessment mechanism, specifically tailored for multi-DNN agent environments, capable of evaluating the agent knowledge is proposed. The self-assessment mechanism consists of two components: an innovative DNN agent architecture capable of adapting to diverse ML tasks and an efficient DNN knowledge assessment method. The proposed DNN agent architecture possesses the ability to conduct knowledge self-assessment and facilitates knowledge exchange between Teacher agents, who possess domain-specific knowledge, and Student agents, who can select the most knowledgeable Teacher agent(s) in a given domain, ensuring the reliability of their inferences. It enables both single and multi-DNN inferences for diverse ML tasks, while also fostering the knowledge exchange between Teacher agents, possessing task-specific expertise, and Student agents, skilled in choosing adept Teachers for their learning. The overarching goal is to reliably pinpoint the most domain-knowledgeable Teacher DNN agent(s) using the minimal amount of testing data, towards achieving optimal performance. Leveraging the proposed DNN knowledge assessment method, we explore the critical question of determining the minimum amount of data necessary for the network environment to achieve optimal performance and provide reliable agent inferences.

In our study, DNN knowledge is straightforwardly defined and assessed post-training, as the success rate of the DNN predictions on a pre-determined test set. This approach provides a clear and reasonable solution to the knowledge assessment problem in the classification tasks we aim to examine. To determine the required sample size for reliable DNN agent inferences, individual testing of the DNN agent modules is conducted across various testing dataset cardinalities ranging from one data sample to many (*e.g* two thousand data samples N in our simulation experiments).





The mean value \bar{e} and the variance σ_e^2 of the evaluation metric values $e_i, i = 1, \dots, N$ for the total samples of testing evaluation values are obtained. For each one of these cardinalities in the range $[1, \dots, 2000]$, the networks are tested repeatedly multiple times so that the mean value and the variation of the evaluation metric can safely be estimated. The selection of data cardinalities was determined through extensive experimentation, revealing that further expansion of the dataset is unnecessary.

To verify the normal distribution of the evaluation metrics, essential for applying statistical inference methods accurately, the Shapiro-Wilk normality test is employed [176] on a random sample of 500 values. A confidence interval is a statistical range derived from a sample of data that is utilized to estimate a population parameter. The interval is accompanied by a confidence level, representing the degree of confidence in the interval containing the true population parameter. The 95% confidence interval for the population mean \bar{X} can be expressed as:

$$95\% \text{ confidence interval} = \bar{X} \pm 1.96\sigma/\sqrt{n}. \quad (2)$$

To determine the sufficient number of testing samples required for reliable agent inferences, once it is confirmed that the evaluation metric values conform to a Gaussian curve, the sample mean \bar{e} , sample standard deviation σ_e^2 , and confidence intervals for the population mean are calculated. The minimum number of testing samples needed is determined based on the condition that the evaluation metric corresponding to that number equals the total sample size mean value \bar{e} , with a 95% confidence interval. In particular, the quantity of testing samples needed corresponds to the number of samples necessary for the network to attain an evaluation metric equivalent to:

$$\bar{e} + 1.96\sigma_e/\sqrt{N}. \quad (3)$$

To establish an efficient knowledge assessment method, a novel agent architecture has been devised. It incorporates the *Agent Knowledge Self-Assessment Module* (AKSAM) that can conduct self-assessment for each agent, offering reliable decisions about the agent's knowledge status at any given time for each task when encountering new and relevant data. By implementing this approach, DNN agents can obtain information about the training domain of any potential Teacher agent. The OOD detection algorithm utilized is based on a Variational Auto-Encoder (VAE) using an OOD detection score, namely the Likelihood Regret (LR) [159]. The suggested agent structure integrates the *Decision Module* (DM), typically a classification DNN, which is tasked with handling new data and delivering decisions about the given task. In the scenarios explored within this paper, the DM typically is a DNN. This agent module can learn and execute inferences or engage in multi-agent inferences, making it adaptable to a diverse range of classification problems. DM results can be forwarded to other agents and the DNN agent can assimilate or disseminate knowledge.

Let a multi-agent environment comprise N agents, denoted as a_1, a_2, \dots, a_N , where each agent possesses the ability to self-assess its knowledge regarding a specific data domain, denoted as k_1, k_2, \dots, k_N . Then, the knowledge k of an agent a can be compared with that of others within the environment, e.g., $k_1 > k_2$. Consequently, the capabilities of the multi-agent environment are harnessed, allowing each agent to make decisions regarding its capacity to handle new data and proceed with the inference process. Additionally, agents can determine whether cooperation with other agents is beneficial, based on the knowledge they possess for the given task, leading to multi-agent inference. For example, if for a specific task $k_1 > k_2 > \dots > k_N$, we can assume that the agent a_1 can independently proceed with the inference. Similarly, if $k_1 > k_3, \dots, > k_N$ and $k_2 > k_3, \dots, > k_N$, we assume that the agents a_1 and a_2 can collaboratively engage in multi-agent inference. Conversely, an agent may recognize its limitations in handling certain data and may seek knowledge from other agents through Knowledge Distillation (KD) algorithms [160]. This comparative assessment, presented in Figure 12 empowers the agents to make informed decisions



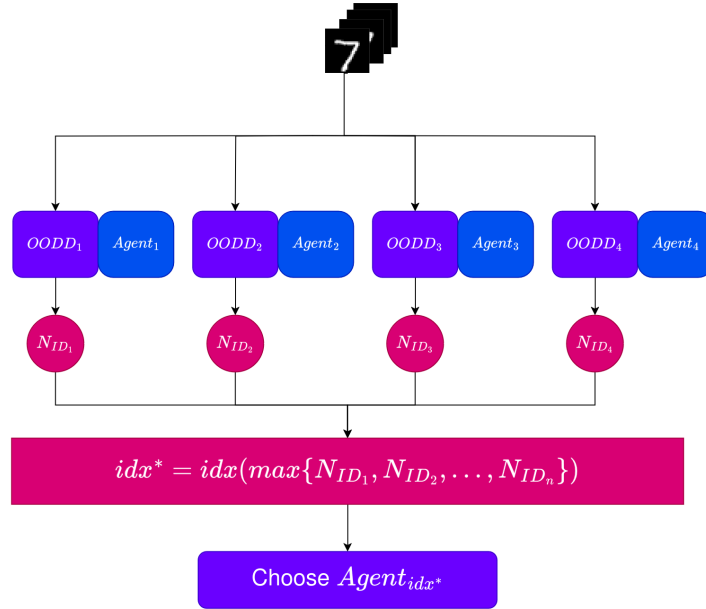


Figure 12. Integration of an Out-of-Distribution (OOD) Detection Module in each DNN agent to optimize evaluation in a multi-agent system. The selection of the Teacher agent is based on task-relevant In-Distribution (ID) data. The DNN agent possessing the most ID data is selected to function as a Teacher agent. We notate as $idx(\cdot)$ the function that returns the index of its argument and idx^* its output.

about their ability to process new data efficiently. The identification of the optimal Teacher agent involves assessing and comparing the knowledge of all agents, considering their training domains. For instance, if $k_1 < k_2$, $k_2 >$, ..., $> k_N$, the Teacher agent would be a_2 and thus, a_1 or any other agent can distill knowledge from it. In scenarios where the collaboration of multiple agents is required, such as $k_1 < k_2$ and $k_3 >$, ..., $> k_N$, multiple Teacher agents can be defined (a_2 and a_3) and thus, other agents can distill knowledge from both of them.

4.5.2. Experimental results

To determine the appropriate number of samples required for reliable agent conclusions, AKSAM, specifically the LR VAE [159], is tested using different numbers of test samples. The performance of the OOD detector is evaluated using the Area Under the Curve-Receiver Operating Characteristics (AUCROC) score, which provides a comprehensive assessment of performance across various score thresholds [177]. Each experiment utilizes a pair of datasets: the first dataset, D_{in} , contains ID data samples, processed by our OOD detector, while the second, D_{out} , consists of OOD data samples. The key findings of our study are presented in Table 5, which shows the experimental results for multiple datasets and various testing data sample cardinalities in the range $[1, \dots, 2000]$.

Figure 13 presents a plot that examines the correlation between AUCROC scores and the number of samples used by the OOD detector during the inference procedure. The plot demonstrates that as the sample size increases, the variability in AUCROC scores decreases, resulting in a more stable range of values. Importantly, the AUCROC scores tend to stabilize beyond a certain number of testing samples, suggesting that further increases in sample size do not significantly enhance performance. This observation is crucial for identifying the minimum number of samples required to ensure reliable conclusions from the agents.





Table 5. Out-of-Distribution (OOD) detection experimental results. The metric used to evaluate the performance of the OOD detector is the AUCROC. The D_{in} is the In-Distribution dataset, while D_{out} is the Out-of-Distribution dataset. The results presented are the mean values from ten independent experiments.

Dataset and Test Samples	Number of Testing Samples	AUCROC	Standard Deviation
D_{in} : F-MNIST [9]- D_{out} : MNIST [10]	15	0.9533	0.0137
	50	0.9856	0.0294
	500	0.9237	0.0086
	1000	0.9556	0.0064
	1500	0.9680	0.0037
	2000	0.9582	0.00379
D_{in} : SVHN [178]- D_{out} : Cifar-10 [179]	15	0.7424	0.04696
	50	0.7139	0.0654
	500	0.7285	0.0031
	1000	0.7317	0.0080
	1500	0.7303	0.0078
	2000	0.7312	0.0042
D_{in} : Cifar-10 [179]- D_{out} : SVHN [178]	15	0.7984	0.02570
	50	0.7794	0.0333
	500	0.8294	0.0119
	1000	0.8359	0.0048
	1500	0.8364	0.0049
	2000	0.8295	0.00448
D_{in} :Cifar-100 [179]- D_{out} : SVHN [178]	15	0.7984	0.02570
	15	0.7076	0.0861
	500	0.8021	0.0147
	1000	0.8043	0.0074
	1500	0.8029	0.0030
	2000	0.8070	0.0051

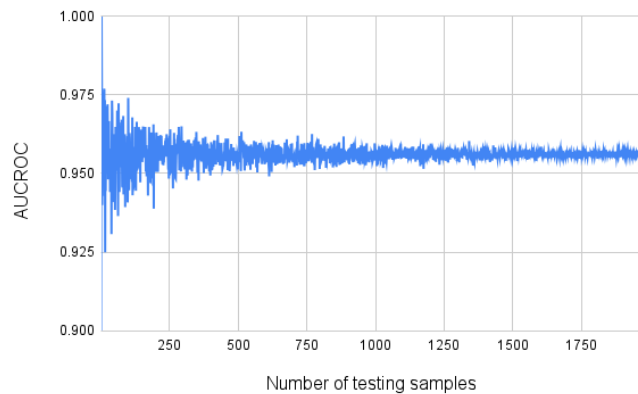


Figure 13. Relationship between AUCROC scores and sample size for the dataset Fashion MNIST [9] as D_{in} and the MNIST [10] as D_{out} .





In Table 6 we present the estimations derived based on the proposed methodology. The values indicate the minimum sample size required by the OOD detectors for multiple D_{in} datasets. The results indicate that the OOD detectors can be considered reliable when providing inferences for 1% of the data cardinality they are trained on. This observation, consistent with the data presented in Figure 13, emphasizes that the OOD detector can deliver accurate evaluations using only a small percentage of the data relative to what it was trained on, affirming its reliability and utility. Consequently, OOD detection is an effective and straightforward option for evaluating agents, tracking their domain expertise, and selecting the appropriate Teacher agent within a multi-agent environment.

Table 6. Minimum number of data required to ensure the reliability of the OOD detection module for each training dataset, D_{in} , and the percentage in relation to the training dataset size.

Dataset	Samples needed	Percentage of samples needed
F-MNIST [9]	586	0.98%
SVHN [178]	734	1.04%
Cifar-10 [179]	486	0.97%
Cifar-100 [179]	294	0.59%

To ascertain the optimal number of test samples necessary for trustworthy agent conclusions, the DM is subjected to testing using varying unknown testing sample sizes following the same experimental procedure.

4.5.3. Conclusion

Operating within a multi-DNN agent environment, where all agents are potential Teachers, our knowledge assessment method evaluates the agent knowledge and determines the minimum amount of data essential for reliable agent inferences. The proposed novel agent architecture can adapt to diverse task objectives, conduct knowledge self-assessment, and make single or collaborative inferences. Within the network environment, knowledge transfer is also possible to occur between Teacher agents possessing domain-specific knowledge and Student agents, who can select competent Teachers from whom to learn. Our experimental results indicate that a small subset of the total data is sufficient for the DNN agent to make reliable inferences.

4.5.4. Relevant publications

- I. Valsamara, C. Papaioannidis, I. Pitas, Domain Expertise Assessment for Multi-DNN Agent Systems, IEEE International Workshop on Distributed Intelligent Systems (DistInSys 2024), [180], Zenodo link: <https://zenodo.org/records/13384163>
- I. Valsamara, C. Papaioannidis, I. Pitas, Efficient DNN knowledge assessment in a multi-agent Teacher-Student DNN environment, Under review.

4.5.5. Relevance to AI4Media use cases and media industry applications

In this study, we introduce a novel self-assessment mechanism composed of two key components: an efficient knowledge assessment method and an innovative agent structure specifically tailored for multi-agent environments. It finds relevance in several AI4Media use cases where advanced deep





learning techniques play pivotal roles, such as UC3 “AI in Vision -High quality Video Production and Content Automation” and UC7 “AI for Content Organisation and Moderation”. It offers a novel method of DNN knowledge assessment, making it a valuable asset in addressing the challenges of content organization, content enhancement, and media content analysis. It can be also incorporated in all AI4Media use cases where the knowledge of DNNs needs to be evaluated.





5. Manifold learning and disentangled feature representation (Task 3.2)

Contributing partners: QMUL, UNITN

In recent years, manifold and disentangled feature representation learning have risen as a prominent research area addressing the problem of finding meaningful representation schemes for both the generative and the discriminative learning paradigms. Studying the structure of latent spaces of generative methods (such as GANs) by discovering semantic paths that govern the generation process, and therefore generating in a controllable manner synthetic data that can reduce dataset biases [14]. Similarly, finding directions in the latent space can help model modes of variation and disentangle different types of transformations [15, 16]. Advances in both generative and discriminative regimes are particularly useful in media generation and visual content analysis.

5.1. Flow Factorized Representation Learning

Contributing partner: UNITN

5.1.1. Introduction and methodology

Developing models which learn useful representations of data has become an increasingly important focus in the machine learning community [99, 181]. However, a precise definition of what makes an ideal representation is still debated. One line of work has focused on ‘disentanglement’ of the underlying ground truth generative factors [99, 182, 183]. In general, the definition of ‘disentanglement’ often refers to learning and controlling statistically independent factors of variation [99, 184]. Over the years, many disentanglement methods have been proposed, including axis-aligned single-dimensional manipulation [182, 183], linear multi-dimensional traversals [185–188], and, more recently, dynamic non-linear vector-based traversals [189, 190]. Although these methods have been met with significant success (and even linked to single-neuron brain activity [191, 192]), there are known theoretical limitations which make them ill-specified, including the presence of topological defects [193]. This has limited their deployment beyond toy settings.

Another line of work has focused on developing representations which respect symmetries of the underlying data in their output space [184, 194]. Specifically, equivariant representations are those for which the output transforms in a known predictable way for a given input transformation. They can be seen to share many similarities with disentangled representations since an object undergoing a transformation which preserves its identity can be called a symmetry transformation [184]. Compared with disentanglement methods, equivariant networks are much more strictly defined, allowing for significantly greater control and theoretical guarantees with respect to the learned transformation [195–199]. However, this restriction also limits the types of transformations to which they may be applied. For example, currently only group transformations are supported, limiting real-world applicability. To avoid this caveat, some recent attempts propose to learn general but approximate equivariance from disentangled representations [200–202].

In this research, we considered an alternative viewpoint at the intersection of these two fields which we call Flow Factorized Representation Learning. Fig. 14 depicts the high-level illustration of our method. Given k different transformations $p_k(\mathbf{x}_t|\mathbf{x}_0)$ in the input space, we have the corresponding latent probabilistic path $\int_{\mathbf{z}_0, \mathbf{z}_t} q(\mathbf{z}_0|\mathbf{x}_0)q_k(\mathbf{z}_t|\mathbf{z}_0)p(\mathbf{x}_t|\mathbf{z}_t)$ for each of the transformations. Each latent flow path $q_k(\mathbf{z}_t|\mathbf{z}_0)$ is generated by the gradient field of some learned potentials ∇u^k following fluid mechanical dynamic Optimal Transport (OT) [203]. Our framework allows for novel



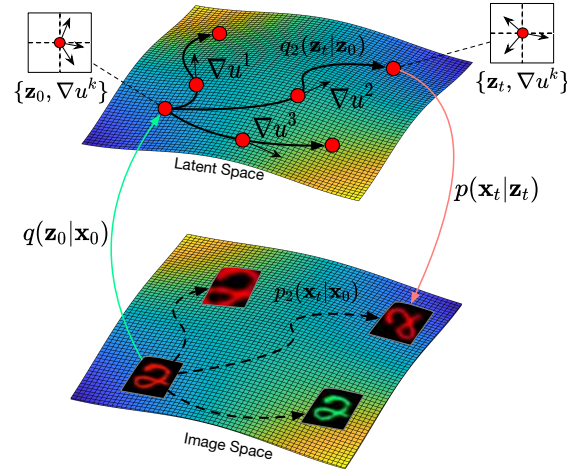


Figure 14. Illustration of our flow factorized representation learning: at each point in the latent space we have a distinct set of tangent directions ∇u^k which define different transformations we would like to model in the image space. For each path, the latent sample evolves to the target on the potential landscape following dynamic optimal transport.

understandings of both *disentanglement* and *equivariance*. The definition of disentanglement refers to the distinct set of tangent directions ∇u^k that follow the OT paths to generate latent flows for modeling different factors of variation. The concept of equivariance in our case means that the two probabilistic paths, *i.e.*, $p_k(\mathbf{x}_t|\mathbf{x}_0)$ in the image space and $\int_{\mathbf{z}_0, \mathbf{z}_t} q(\mathbf{z}_0|\mathbf{x}_0)q_k(\mathbf{z}_t|\mathbf{z}_0)p(\mathbf{x}_t|\mathbf{z}_t)$ in the latent space, would eventually result in the same distribution of transformed data.

We build a formal generative model of sequences and integrate the above latent probability evolution as condition updates of the factorized sequence distribution. Based on the continuity equation, we derive a proper flow of probability density for the time evolution of both the prior and posterior. To perform inference, we approximate the true posterior of latent variables and train the parameters as a Variational Autoencoder (VAE) [204]. When the transformation type k is not observed (*i.e.*, available as a label), we treat k as another latent variable and incorporate its posterior into our framework by learning it from sequences. Extensive experiments and thorough analyses have been conducted to show the effectiveness of our method. For example, we demonstrate empirically that our representations are usefully factorized, allowing flexible composability and generalization to new datasets. Furthermore, we show that our methods are also approximately equivariant by demonstrating that they commute with input transformations through the learned latent flows. Ultimately, we see these factors combine to yield the highest likelihood on the test set in each setting.

5.1.2. Experiments

Datasets. We evaluate our method on two widely-used datasets in generative modeling, namely MNIST [205] and Shapes3D [1] (we present here results only for the latter). For Shapes3D [1], we use the self-contained four transformations that consist of Floor Hue, Wall Hue, Object Hue, and Scale.

Baselines. We mainly compare our method with SlowVAE [200] and Topographic VAE (TVAE) [201]. These two baselines could both achieve approximate equivariance. Specifically, TVAe introduces some learned latent operators, while SlowVAE enforces the Laplacian prior $p(\mathbf{z}_t|\mathbf{z}_{t-1}) = \prod \alpha\lambda/2\Gamma(1/\alpha) \exp(-\lambda|z_{t,i} - z_{t-1,i}|^\alpha)$ to sequential pairs. Within the disentanglement



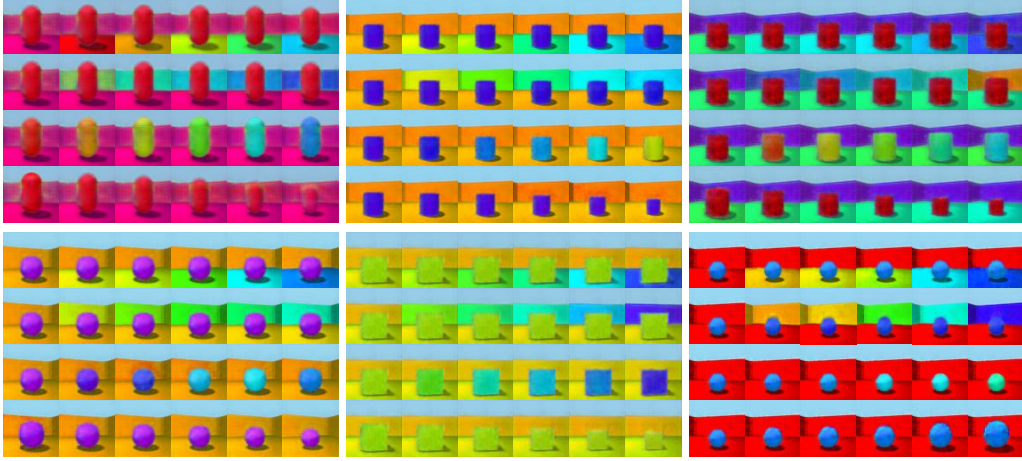


Figure 15. Exemplary latent flow results on Shapes3D [1]. The transformations from top to bottom are Floor Hue, Wall Hue, Object Hue, and Scale, respectively. The images of the top row are from the supervised experiment, while the bottom row is based on the weakly-supervised experiment.

literature, our method is compared with the supervised PoFlow [202], which adopts a wave-like potential flow for sample evolution, and the unsupervised β -VAE [182] and FactorVAE [206], which encourage independence between single latent dimensions. Finally, the vanilla VAE is used as a controlled baseline.

Metrics. We use the approximate equivariance error \mathcal{E}_k and the log-likelihood of transformed data $\log p(\mathbf{x}_t)$ as the evaluation protocols. The equivariance error is defined as $\mathcal{E}_k = \sum_{t=1}^T |\mathbf{x}_t - \text{Decode}(\mathbf{z}_t)|$ where $\mathbf{z}_t = \mathbf{z}_0 + \sum_{t=1}^T \nabla_{\mathbf{z}} u^k$. For TVAE, the latent operator is changed to $\text{Roll}(\mathbf{z}_0, t)$. For unsupervised disentanglement baselines [182, 206] and SlowVAE [200], we carefully select the latent dimension and tune the interpolation range to attain the traversal direction and range that correspond to the smallest equivariance error. Since the vanilla VAE does not have the corresponding learned transformation in the latent space, we simply set $\nabla_{\mathbf{z}} u^k = 0$ and take it as a lower-bound baseline. For all the methods, the results are reported based on 5 runs. Note that the above equivariance error is defined in the output space. Another reasonable evaluation metric is instead measuring the error in the latent space as $\mathcal{E}_k = \sum_{t=1}^T |\text{Encode}(\mathbf{x}_t) - \mathbf{z}_t|$. We see the first evaluation method is more comprehensive as it further involves the decoder in the evaluation.

Methods	Supervision?	Equivariance Error (\downarrow)				Log-likelihood (\uparrow)
		Floor Hue	Wall Hue	Object Hue	Scale	
VAE [204]	No (✗)	6924.63±8.92	7746.37±8.77	4383.54±9.26	2609.59±7.41	-11784.69±4.87
β -VAE [182]	No (✗)	2243.95±12.48	2279.23±13.97	2188.73±12.61	2037.94±11.72	-11924.83±5.64
FactorVAE [206]	No (✗)	1985.75±13.26	1876.41±11.93	1902.83±12.27	1657.32±11.05	-11802.17±5.69
SlowVAE [200]	Weak (✓)	1247.36±12.49	1314.86±11.41	1102.28±12.17	1058.74±10.96	-11674.89±5.74
TVAE [201]	Yes (✓)	1225.47±9.82	1246.32±9.54	1261.79±9.86	1142.01±9.37	-11475.48±5.18
PoFlow [202]	Yes (✓)	885.46±10.37	916.71±10.49	912.48±9.86	924.39±10.05	-11335.84±4.95
Ours	Yes (✓)	613.29±8.93	653.45±9.48	605.79±8.63	599.71±9.34	-11215.42±5.71
Ours	Weak (✓)	690.84±9.57	717.74±10.65	681.59±9.02	653.58±9.57	-11279.61±5.89

Table 7. Equivariance error \mathcal{E}_k and log-likelihood $\log p(\mathbf{x}_t)$ on Shapes3D [1].

Qualitative results. Fig. 15 displays decoded images of the latent evolution on Shapes3D [1]. Our latent flow can perform the target transformation precisely during evolution while leaving other traits of the image unaffected. In particular, for the weakly-supervised setting, the decoded





images (*i.e.*, the bottom row of Fig. 15) can still reproduce the given transformations well and it is even hard to visually tell them apart from the generated images under the supervised setting. This demonstrates the effectiveness of the weakly-supervised setting of our method, and implies that qualitatively our latent flow is able to learn the sequence transformations well under both supervised and weakly-supervised settings.

Quantitative results. Table 7 compares the equivariance error and the log-likelihood on Shapes3D [1]. Our method learns the latent flows which model the transformations precisely, achieving the best performance across datasets under different supervision settings. Specifically, our method outperforms the previous best baseline by 291.70 in the average equivariance error and by 120.42 in the log-likelihood. In the weakly-supervised setting, our method also achieves very competitive performance, falling behind that of the supervised setting in the average equivariance error slightly by 67.88 on Shapes3D.

5.1.3. Conclusions and Limitations

In this research, we introduced Flow Factorized Representation Learning which defines a set of latent flow paths that correspond to sequences of different input transformations. The latent evolution is generated by the gradient flow of learned potentials following dynamic optimal transport. Our setup re-interprets the concepts of both *disentanglement* and *equivariance*. Extensive experiments demonstrate that our model achieves higher likelihoods on standard representation learning benchmarks while simultaneously achieving smaller equivariance error. Furthermore, we show that the learned latent transformations generalize well, allowing for flexible composition and extrapolation to new data.

Regarding the limitations, for flexibility and efficiency, we use Physical Informed Neural Network (PINN) constraints [207] to model the Hamilton–Jacobi (HJ) equation. However, such partial differential equation (PDE) constraints are approximate and not strictly enforced. Other PDE modeling approaches include accurate neural PDE solvers [208–210] or other improved PINN variants such as competitive PINNs [211] and robust PINNs [212]. Also, when inferring with observed k , we change the posterior from $q(\bar{z}|\bar{x}, k)$ to $q(\bar{z}|\mathbf{x}_0, k)$ because we assume k contains sufficient information of the whole sequence. To keep the posterior definition of $q(\bar{z}|\bar{x}, k)$, we need to make $q(\mathbf{z}_t)$ also a function of \mathbf{x}_t . This can be achieved either by changing the potential to $u(\mathbf{z}_{t-1}, \mathbf{x}_t, t-1)$ or modifying the external driving force to $f(\mathbf{z}_{t-1}, \mathbf{x}_t, t-1)$. Nonetheless, we see these modifications would make the model less flexible than our current formulations as the element \mathbf{x}_t might be needed during inference.

5.1.4. Relevant publications

- Y. Song, A. Keller, N. Sebe, and M. Welling, Latent Traversals in Generative Models as Potential Flows, International Conference on Machine Learning (ICML), July 2023 [190]. Zenodo record: <https://zenodo.org/record/8335476>
- Y. Song, A. Keller, N. Sebe, and M. Welling, Flow Factorized Representation Learning, Neural Information Processing Systems (NeurIPS), December 2023 [16]. Zenodo record: <https://zenodo.org/records/11303551>

5.1.5. Relevant software/datasets/other outcomes

The Pytorch implementation can be found in

- <https://github.com/KingJamesSong/latent-flow>



- <https://github.com/KingJamesSong/PDETraversal>

5.1.6. Relevance to AI4Media use cases and media industry applications

Our algorithm provides a solution for defining a set of latent flow paths that correspond to sequences of different input transformations using the gradient flow of learned potentials following dynamic optimal transport. As such, it may exhibit wide applicability in various media industry applications, such as image editing and generation. Specifically, since generative learning is fundamental in creative industries, our method can be used to control content generation that can subsequently be incorporated in media industry for the generation of controllable media content.

5.2. Parts of Speech–Grounded Subspaces in Vision-Language Models

Contributing partner: QMUL

5.2.1. Introduction and methodology

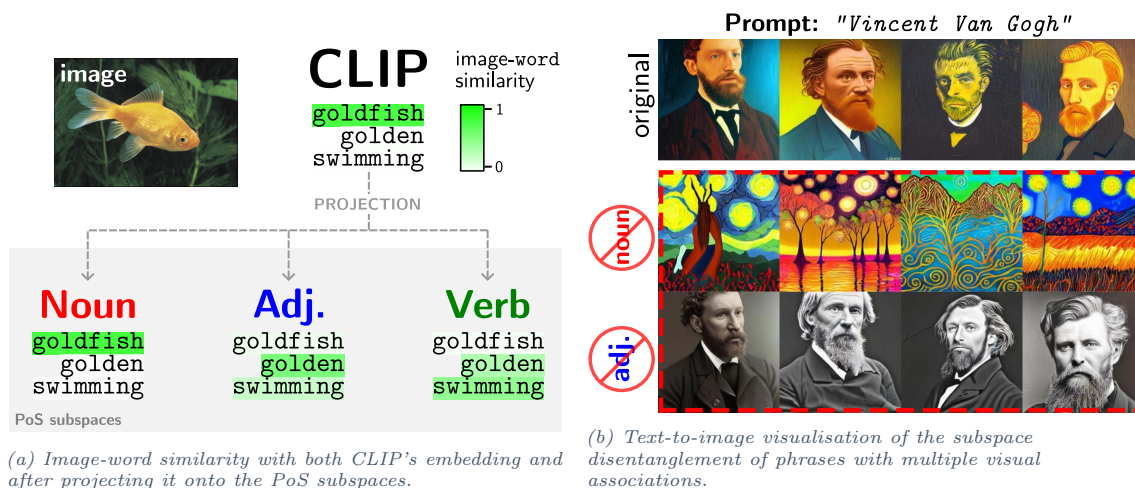


Figure 16. CLIP represents multiple visual modes of variation in an embedding (e.g. the ‘object’ and its ‘appearance’). The learnt PoS subspaces more reliably separate the constituent visual components.

Many recent advances in machine learning have been driven by vision-language (VL) models’ ability to learn powerful, generalisable image representations from natural language supervision [93, 213, 214]. The image features from VL models well-capture representations of many visual attributes as evidenced by the broad applicability they have found for use in downstream tasks. The image or text encoders of CLIP in particular [93] have been used for controllable image synthesis [215–217], image captioning [218, 219], and multiple other discriminative tasks [220–222]. However, modeling the many different visual modalities in a single vector representation is not without its drawbacks – recent work shows that CLIP’s visual representations are often *entangled*. For example, [223] find that specific neurons fire in response to both images containing a visual concept and *images of text* relating to the same concept. This leaves CLIP open to vulnerabilities in the form of ‘Typographic attacks’ – writing another class name as text on the image can often cause CLIP to predict this class with a higher probability than that of the original image’s true category. Other recent works show that CLIP’s visual representations encode task-specific attributes (such as





of the object or action depicted in an image) in an unpredictable manner, and often the embedding is biased in the prominence with which it encodes different modalities [224]. As a visual example, we find that CLIP encodings of text prompts containing ‘visually polysemous’ [225] phrases of artists’ names lead CLIP-based text-to-image models [226] to synthesize an unpredictable combination of *both* images of the artists and of artworks in their signature styles (as shown in 16b). Multiple visual associations of the text prompt, including both the appearance of the artist themselves and the style of their artwork, are entangled in the same CLIP embedding. For VL representations to make for useful image features, it’s vital that the particular modalities of interest are indeed well-represented in the embedding. One popular means to this end is fine-tuning the representations for specific downstream tasks [220]. However, this not only requires additional computation but makes the restrictive assumption of the existence of labelled data for each task.

In this work, we address the problem of better disentangling the modes of visual variation in CLIP’s shared vision-language space. In particular, we ask the question: do there exist subspaces in CLIP’s joint VL space that capture the representations of the ‘content’ of an image or text, that are invariant to its ‘appearance’? To take the first steps towards achieving this we leverage the association between *parts of speech* in natural language and specific modes of visual variation: our learnt noun subspace isolates representations of the ‘object’ of an image or text prompt (e.g. an animal in an image, or the noun described in a sentence), and the adjective space its appearance (e.g. whether an object is shiny, or a scene is snowy).

We seek a lower-dimensional subspace on which to project either a text or image CLIP representation $\mathbf{z} \in \mathbb{R}^d$ to predictably isolate the desired visual mode of variation. We achieve this through natural language supervision in the form of words from the different parts of speech. Let the elements of a set \mathcal{C} index into the relevant word class of interest (i.e. $\mathcal{C} = \{N, A, V, R\}$ for nouns, adjectives, verbs, and adverbs respectively), and $\mathbf{X}_i \in \mathbb{R}^{d \times n}, \forall i \in \mathcal{C}$ contain in their columns the CLIP encodings of n words belonging to word class i [227]. Then, for a word class $i \in \mathcal{C}$ of interest, we seek a k -dimensional subspace of \mathbb{R}^d spanned by the columns of a learnt $\mathbf{W}_i \in \mathbb{R}^{d \times k}$ in which the CLIP representations of the words in the class of interest i have a large norm and the remaining categories’ representations in $\mathcal{C} \setminus \{i\}$ are close to the zero vector. Intuitively, a hyperplane with this property models factors of variation that are uniquely present in representations of text that belong to a particular part of speech. We quantify this by formulating the following objective function:

$$\mathbf{W}_i = \arg \max_{\mathbf{W}_i^\top \mathbf{W}_i = \mathbf{I}_k} \left\{ (1 - \lambda) \|\mathbf{W}_i^\top \mathbf{X}_i\|_F^2 - \sum_{j \in \mathcal{C} \setminus \{i\}} \lambda \|\mathbf{W}_i^\top \mathbf{X}_j\|_F^2 \right\}, \quad (4)$$

where $\lambda \in [0, 1]$ is a hyperparameter that controls the importance of killing the variation in the non-target categories relative to preserving the variation in the target class. We solve for the subspaces in closed-form, proposing a further geometry-respecting extension of the subspaces.

5.2.2. Experimental results

One societal concern with free-form text-to-image models (TTIMs) is their ability to produce imitation artworks copying the style of artists. Here, we show how the learnt adjective subspace can be used as a step towards mitigating this. To achieve this, one simply modifies the TTIM forward pass to first project the CLIP text representations onto the orthogonal complement of the adjective subspace with $\Pi_A^\perp(\mathbf{z}_T)$ before feeding it into the image generator. We see from the results in Figure 17 that this modification indeed prevents the imitation of the visual styles of a range of artists (even with multiple forms of sentence structure in the prompt), whilst still enabling a diverse set of images to be generated nonetheless.

Visual theme subspaces Whilst successfully preventing the visual imitation of many famous artists, applying the adjective subspace projection to *every* text prompt’s CLIP representation



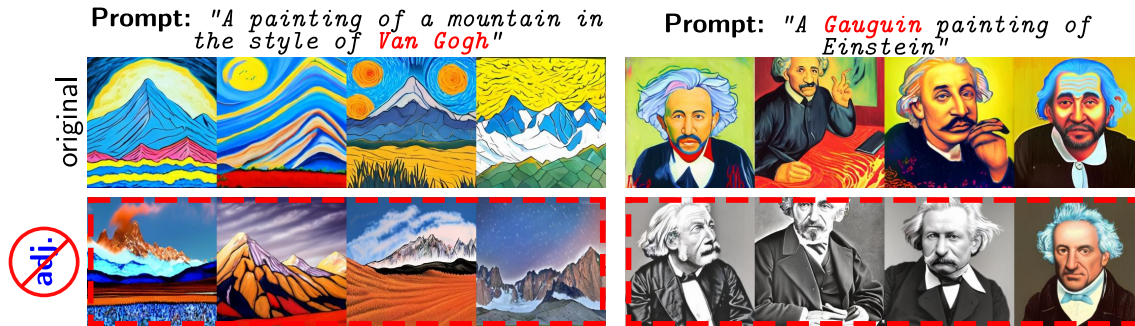


Figure 17. Killing the CLIP representations' component in the adjective subspace provides a way to block the imitation of artists' styles in CLIP-based text-to-image models.

can restrict the ability to use adjectives to specify visual appearance. We find a further effective strategy is to build additional subspaces for more specific visual appearances (e.g. artists' painting styles). Concretely, we embed 830 artists' names and surnames in a new matrix \mathbf{X}_i , and solve Eq. 4 using all PoS classes in the negative summation to prevent the destruction of existing concepts. In contrast to the adjective space, projection onto the orthogonal complement of this 'artist subspace' *preserves* adjective-based visual descriptions whilst also successfully preventing style imitation (Figure 18a). Crucially, we highlight that for the example shown in Figure 18a, **the artist's name Qi Baishi is not present in the 'training' list of example artists**, suggesting the subspace has learnt a more general notion of an artist rather than simply the variation for only those artists whose names are provided as supervision.



(a) A custom subspace for 'artistic style'.

(b) A custom subspace for gory/bloody visual themes.

Figure 18. Projecting onto the orthogonal complement of custom visual theme subspaces erases specific appearances from CLIP-based text-to-image models' images.

5.2.3. Conclusions

In this work, we have proposed a method for learning geometry-aware subspaces in CLIP's vision-language space that disentangle the modes of visual variation in representations of both images and text. To achieve this, we used the semantic relationship between parts of speech in natural language and specific visual modes of variation. We demonstrated the disentanglement qualitatively with a text-to-image model, showcasing the model's ability to remove visual imitation of artists'





styles from synthetic images. A common drawback of subspace learning approaches is choosing a good number of dimensions k . Our method inherits this limitation, and one must choose the appropriate value for the specific task. Despite this, the closed-form eigensolution means only a single fast computation is needed, and any desired number of eigenvectors can be used at test-time. Although the recovered subspaces show wide applicability in downstream tasks, they are not able to perfectly separate the modes of variation for every possible image and text prompt.

5.2.4. Relevant publications

Oldfield, J., Tzelepis, C., Panagakis, Y., Nicolaou, M., & Patras, I. (2023). Parts of Speech–Grounded Subspaces in Vision-Language Models. *Advances in Neural Information Processing Systems*, 36, 2700–2724. <https://zenodo.org/records/13619498>

5.2.5. Relevant software/datasets/other outcomes

Code is available at <https://github.com/james-oldfield/PoS-subspaces>.

5.2.6. Relevance to AI4Media use cases and media industry applications

The work provides a means of more controllable image generation and facilitates content generation in all relevant media industry applications.

5.3. Improving Fairness using Vision-Language Driven Image Augmentation

Contributing partners: UNITN, QMUL

5.3.1. Introduction and methodology

Today’s society is careful about ethical topics and with the raising of publicly available AI tools [228–230] concerns about their fairness are also growing. In a supervised learning setting, the importance of the training data is well-known since the behavior of the model at inference time is highly correlated to the seen data. Modern models can effectively learn and highly perform multiple downstream tasks generalizing to unseen data. Besides the effectiveness of the pipeline, training data also brings unwanted side effects. It has been proven that vision datasets contain biases [231], thus the models learn the correlations present in the data which may be malignant [232–235]. These concerns become a particularly sensitive subject when it comes to the facial domain. Modern machine learning models are dominant at a wide range of applications, such as face/emotion recognition and mask detection [230, 236, 237]. In this context, studying the behavior of deep learning models is crucial to avoid unwanted situations at inference time [238]. For example, the model’s performance may drop when presented with a particular protected characteristic (e.g., very young/old or dark-skinned faces). The above issues motivate us to study the behavior of a deep learning model with respect to facial protected characteristics which are sensitive to society and can raise ethical concerns.

Recently, generative models, such as Generative Adversarial Networks (GANs) [239], have shown remarkable performance in a multitude of tasks through discovering controllable generative paths in their latent or feature spaces [240–244]. Thus, GAN-based methods have been employed as a data augmentation technique to generate fairer data [245–247], to generate counterfactuals [248, 249] or to generate counterparts by editing sensitive attributes [250]. The above works train generative models from scratch which may be impractical, especially in low data regimes. Additionally, the



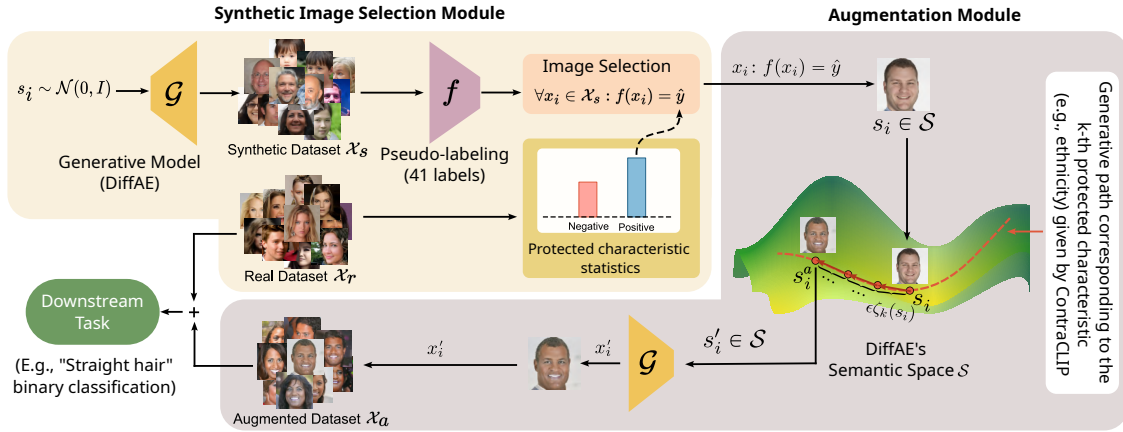


Figure 19. Overview of the proposed Vision-Language Bias Control (VLBC) method for controlling the bias in facial image datasets. Given a real training set \mathcal{X}_r and a downstream task, we find the under-represented protected characteristic (e.g., black in skin colour) by computing the sample statistics. Based on this, we select which images from a synthetic dataset \mathcal{X}_s (generated using the DiffAE [11] generator dataset \mathcal{X}_s and pseudo-labelled by a pre-trained network on 41 attributes f [12, 13]) to use for augmentation. Then, the selected images are manipulated by our augmentation module (ContraCLIP+DiffAE), pre-trained on text prompts defining the desired protected characteristic. In this example, we manipulate/augment the selected images based on skin colour. Note that the original labels of the augmented images (i.e., corresponding to the attribute class at hand) do not change. Finally, the augmented dataset \mathcal{X}_a is used along with the original real dataset \mathcal{X}_r for training downstream tasks.

pre-trained generative models are expected to reflect the bias that is inherent to the datasets where they have been trained on [15, 245, 250], challenging those methods that use them for bias mitigation.

In this work, we address the above limitations by proposing a novel approach that leverages a *pre-trained* diffusion model [11, 251] to edit sensitive attributes in facial images, in order to improve the fairness of existing (biased) datasets and, consequently, the fairness of a discriminative model trained on such datasets. By contrast to previous works that train generative models (e.g., GANs [239]) from scratch [245, 248, 250], we incorporate the power of a fixed pre-trained diffusion model to change sensitive facial attributes from a pool of generated images. The manipulated faces (with respect to the desired sensitive attributes) are used to make the original dataset fairer and mitigate the bias present on a downstream model trained on the original dataset. Our setting consists of a binary downstream classification attribute and a binary sensitive attribute towards which the model may exhibit bias. Throughout this work, we will be referring to the downstream classification attribute as *attribute* and to the sensitive attribute as *protected characteristic*.

Here, we present our method for controlling the bias of a given dataset via augmenting images by editing specific protected characteristics (e.g., *skin colour*) using a diffusion-based generative model (DiffAE [11]) and an augmentation module that learns to generate images driven by prompts in natural language (ContraCLIP [252]). An overview of the proposed framework, which we call Vision-Language Bias Control (VLBC), is shown in Fig. 19. Concretely, given a real training dataset \mathcal{X}_r of facial images, annotated for several attribute classes (e.g., CelebA-HQ’s [12] attributes, such as *chubby*, *long hair*, etc.), we first calculate, for each class, the number of positive and negative samples with respect to a protected characteristic (e.g., *skin colour*). By doing so, we identify whether the protected characteristic at hand is under-represented in the given dataset. Then, after having identified the bias towards a specific protected characteristic, we may control it (i.e., mitigate or increase it) by i) selecting fake images from a large dataset of synthetic facial images generated by DiffAE [11], \mathcal{X}_s , that have been pseudo-labelled by a pre-trained network on 41 attributes f [12, 13] (*Synthetic Image Selection*) and ii) manipulating accordingly using the proposed *Augmentation*



Module. The augmented dataset, \mathcal{X}_a , is then used along with the original (biased) dataset \mathcal{X}_r towards training fairer downstream classifiers. We note that we do not merely sample synthetic images from \mathcal{X}_s since we do not possess any control over the attributes of the generated images and there is no guarantee that generated images will be numerically adequate to compensate for the under-represented classes of protected characteristics. This is simply because synthetic images follow the dataset distribution where the generative models have been trained, thus, they still suffer from biases that are present in those datasets.

5.3.2. Experimental results

In this section, we present the experimental evaluation of the proposed framework for controlling the bias in facial datasets with respect to the protected characteristic of *age*, towards the downstream task of binary attribute classification. We note that we train the classification models only on the classification attributes, not the protected characteristic. We provide results on mitigating the bias (VLBC-), and we compare with the state-of-the-art (SOTA) works of [2–4]. We first train a baseline model on the original training set, quantifying its initial bias without applying any fairness-related technique. Then we investigate how the same model behaves when fine-tuned on the augmented dataset created to mitigate the bias. Moreover, while mitigating the bias, we introduce a second baseline, referred to as *baseline-sampling*, which injects synthetic images of the desired protected characteristic (e.g., black people) in the training set without applying the proposed augmentations. This baseline provides a simple yet effective way of determining whether and when our augmentation scheme is necessary, providing additional insight into the usefulness of synthetic images. We evaluate the proposed framework on **CelebA-HQ** [12], a diverse dataset in terms of *skin colour*, *gender*, and *age*, which contains 30,000 images annotated with 40 attributes.

5.3.2.1. Evaluation metrics Under the binary classification setting, a natural way for describing fairness is to have a model performing *equally* regardless of the protected characteristic. For instance, predicting *big lips* should perform independently of characteristics such as *age* or *skin color*. Following this intuition, we calculate the accuracy of the downstream task conditioned on the protected characteristic [245]. E.g., in the case of the protected characteristic of *age*, we split the attribute classification accuracy into “young” and “old”. We then calculate the difference between the two accuracies to capture the model’s fairness. Ideally, a model will exhibit equal behavior on a zero-valued difference in accuracy. We also note that the sign of the difference in accuracy is indicative of the “direction” of the bias – i.e., a negative difference value would indicate a bias towards elder people, and vice versa for a positive value. We report the overall accuracy, the f1-score, and the difference in accuracy (Acc Diff). Additionally, we calculate the mean (Δ_A) and max (Δ_M) disparity of opportunity similarly to [4].

5.3.2.2. Bias mitigation with VLBC- We show the results in Tab. 8, where we compare the baselines with the following SOTA works: Wang et al. [2] (weighting), Learning from Failure [3] (LfF) and Fairness with the Partially annotated Group labels [4] (CGL-FairHSIC). CGL-FairHSIC proposes to improve fairness by incorporating a partially annotated dataset, thus we apply it to the synthetic dataset. Please note that we denote with a “-” the (Δ_A) and (Δ_M) metrics when the model collapses (e.g, f1-score of LfF [3] and CGL-FairHSIC [4] in some cases). The results show how the proposed framework (VLBC-) is always capable of mitigating bias with respect to the baseline model on all attributes and metrics exhibiting *consistency* over multiple settings. The comparison with SOTA methods highlights how other works are robust in some settings but fail in others. Specifically, Wang et al. [2] (weighting) deteriorates the model’s fairness in *wearing necktie*, *arched eyebrows*, and *chubby* attributes when *age* is the protected characteristic. Moreover,





Task	Method	Accuracy \uparrow	f1-score \uparrow	Acc Diff	Fairness	
					$\Delta_A \downarrow$	$\Delta_M \downarrow$
Wearing Necktie	Baseline	94.37 \pm 0.11	78.19 \pm 0.24	10.65 \pm 0.31	6.02 \pm 0.27	6.59 \pm 0.54
	Baseline-sampling	94.29 \pm 0.11	77.46 \pm 0.59	10.97 \pm 0.26	4.59 \pm 1.57	6.19 \pm 2.13
	Weighting [2]	94.46 \pm 0.04	75.14 \pm 0.09	11.01 \pm 0.06	5.88 \pm 1.14	9.99 \pm 2.43
	LfF [3]	94.84\pm0.13	81.11\pm0.46	10.30 \pm 0.30	4.11\pm0.60	6.35 \pm 0.06
	CGL-FairHSIC [4]	92.83 \pm 0.00	48.14 \pm 0.00	15.88 \pm 0.00	–	–
	VLBC- (ours)	94.69 \pm 0.06	78.48 \pm 0.30	10.07\pm0.13	5.18 \pm 0.35	5.83 \pm 0.75
Chubby	Baseline	93.24 \pm 0.04	72.56 \pm 0.88	15.85 \pm 0.65	8.24 \pm 1.42	9.65 \pm 1.92
	Baseline-sampling	93.16 \pm 0.15	71.6 \pm 0.39	15.51\pm0.48	3.58\pm1.1	5.22\pm0.53
	Weighting [2]	93.33 \pm 0.09	66.33 \pm 0.72	16.75 \pm 0.21	9.64 \pm 0.57	18.15 \pm 1.38
	LfF [3]	92.60 \pm 0.51	76.74\pm0.74	15.76 \pm 0.94	18.28 \pm 2.65	21.77 \pm 3.61
	CGL-FairHSIC [4]	92.49 \pm 0.01	48.93 \pm 0.82	19.41 \pm 0.10	–	–
	VLBC- (ours)	93.44\pm0.02	71.96 \pm 0.45	15.69 \pm 0.22	5.39 \pm 0.37	5.79 \pm 0.28
Arched Eyebrows	Baseline	80.15 \pm 0.3	78.91 \pm 0.34	-9.00 \pm 0.20	9.22 \pm 0.12	12.41 \pm 0.36
	Baseline-sampling	80.18 \pm 0.22	79.15 \pm 0.22	-8.25 \pm 0.17	9.73 \pm 0.22	12.75 \pm 0.22
	Weighting [2]	79.57 \pm 0.13	78.29 \pm 0.14	-9.18 \pm 0.29	6.25\pm0.26	9.01\pm0.45
	LfF [3]	25.27 \pm 0.21	25.06 \pm 0.21	9.36 \pm 0.50	–	–
	CGL-FairHSIC [4]	84.17\pm0.46	83.28\pm0.46	-7.63\pm0.48	6.45 \pm 1.38	10.35 \pm 1.28
	VLBC- (ours)	80.44 \pm 0.06	79.27 \pm 0.04	-8.62 \pm 0.19	10.56 \pm 0.20	13.00 \pm 0.53
Double Chin	Baseline	80.56 \pm 0.14	79.39 \pm 0.14	-8.47 \pm 0.28	10.39 \pm 0.04	12.75 \pm 0.55
	Baseline	94.17 \pm 0.13	74.47 \pm 0.30	15.15 \pm 0.6	18.74 \pm 1.38	27.86 \pm 2.94
	Baseline-sampling	94.36 \pm 0.23	74.21 \pm 0.58	13.71\pm0.85	10.87 \pm 2.25	15.4 \pm 4.51
	Weighting [2]	94.66\pm0.09	69.09 \pm 0.78	14.46 \pm 0.25	1.69\pm0.34	2.00\pm0.24
	LfF [3]	94.16 \pm 0.08	78.39\pm0.29	14.27 \pm 0.13	21.94 \pm 1.01	30.54 \pm 1.89
	CGL-FairHSIC [4]	93.74 \pm 0.00	48.39 \pm 0.00	18.67 \pm 0.00	–	–
VLBC- (ours)	94.48 \pm 0.04	74.35 \pm 0.23	14.72 \pm 0.21	14.20 \pm 0.07	20.69 \pm 0.18	
VLBC- \f (ours)	94.46 \pm 0.04	74.19 \pm 0.33	14.57 \pm 0.15	15.01 \pm 0.25	22.44 \pm 0.48	

Table 8. Results of the classification tasks with *age* as protected characteristic. We train the model with the original training data (baseline), with the original data plus synthetic (baseline-sampling), and with the proposed VLBC-. We compare with weighting [2], LfF [3], and CGL-FairHSIC [4].

the “baseline-sampling” approach mitigates the bias only when enough images are available for balancing the training set – this is clear in the case of *age* (see Tab. 8). In this setting, the diffusion model has a low bias against *age*, thus it generates enough images for both young and old people to balance the original training set.

5.3.3. Conclusions

In this work, we presented a novel framework for controlling the bias in facial datasets leveraging a *pre-trained* and fixed diffusion model. We built on ContraCLIP [252] in order to find meaningful natural language driven generative paths in the semantic space of DiffAE [11], which we then applied to augment a given dataset with respect to under-represented protected characteristics (e.g., black people), making it fairer for downstream tasks. The proposed bias mitigation method (VLBC-) is able to counteract the bias learnt from a downstream model, while preserving accuracy and showing competitive results against existing SOTA works. Additionally, VLBC- exhibits consistency across multiple settings, a trait missing in concurrent.

5.3.4. Relevant publications

D’Incà, Moreno, Christos Tzelepis, Ioannis Patras, and Nicu Sebe. "Improving Fairness using Vision-Language Driven Image Augmentation." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 4695-4704. 2024. Zenodo record: <https://zenodo.org/records/10528955>

5.3.5. Relevant software/datasets/other outcomes

Code is available at <https://github.com/Moreno98/Vision-Language-Bias-Control>.





5.3.6. Relevance to AI4Media use cases and media industry applications

The work provides a means of fairer image generation. In this sense, the proposed method can facilitate content generation (which is the focus of various media-related applications) by mitigating bias against certain protected characteristics.





6. Transfer learning (Task 3.3)

Contributing partners: UNITN

Transfer Learning is an emerging field among Deep Learning practitioners that seeks to reuse and exploit previously generated models for different purposes. Considering the huge amount of data, human effort and computational power needed to train these models, being able to reuse them is of paramount importance. Beyond practical reasons, Transfer Learning poses a scientific challenge of relevance, as it forces researchers to question the internal knowledge representation of deep models. Indeed, to understand how to reuse deep representations, one must first understand how are these representations learned, and how are they internally structured. Advances in this field have potential relevance for key aspects of Deep Learning, such as explainability and interpretability, efficiency and footprint reduction, and real world deployment of AI powered systems.

6.1. Dynamically Instance-Guided Adaptation

Contributing partner: UNITN

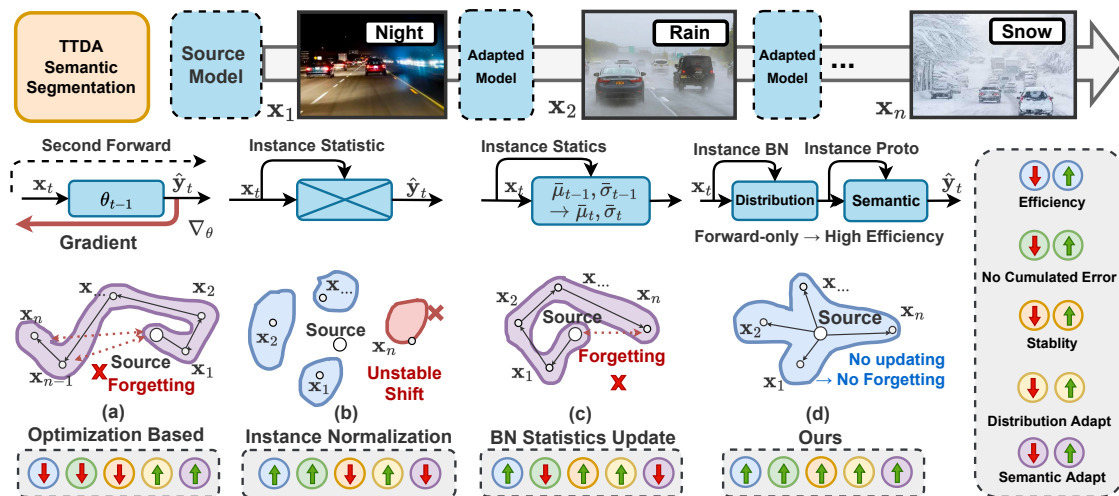


Figure 20. **Top:** Illustration of test-time domain adaptive semantic segmentation (TTDA-Seg). **Bottom:** Comparison with different TTDA methods. The proposed DIGA is a holistic method that has the properties of effectiveness (distribution&semantic adaptation and avoid unstable training&error accumulation) and efficiency (backward-free).

6.1.1. Introduction and methodology

Semantic segmentation (Seg) [253–257] is a fundamental task in computer vision, being an important step in the visual-based robot, autonomous driving and etc. Modern deep-learning techniques have achieved impressive success in segmentation. However, one serious drawback of them is that the segmentation models trained on one dataset (source domain) may undergo catastrophic performance degradation when applied to another dataset sampled from a different distribution. This phenomenon will be even more serious under complex and ever-changing contexts, e.g., autonomous driving.



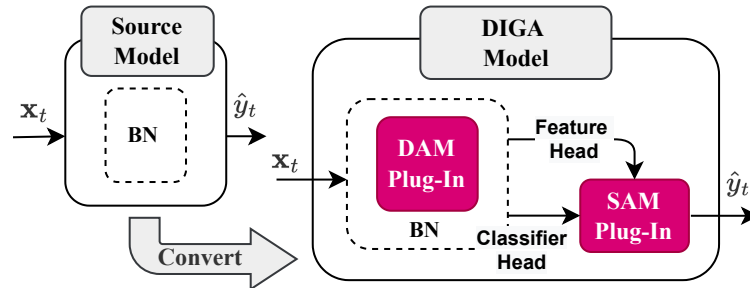


Figure 21. Illustration of the implementation of our DIGA. Given a source model, our DIGA can be readily equipped with only access to the BN layers, classifier head and feature head.

To solve this well-known problem caused by domain shifts, researchers have devoted great effort to domain generalization (DG) [258–263] and domain adaptation (DA) [264, 264–266]. Specifically, DG aims to learn generalized models with only labeled source data. Traditional DA attempts to adapt the model on the target domain by using both labeled source data and unlabeled target data. However, both learning paradigms have their own disadvantages. The performance of DG is limited especially when evaluated on a domain with a large gap from the source since it does not leverage target data [258]. DA assumes that the unlabeled target data are available in advance and can be chronically exploited to improve target performance. This assumption, however, can not always be satisfied in real-world applications. For example, when driving in a new city, the data are incoming sequentially and we expect the system to dynamically adapt to the ever-changing scenario.

To meet the real-world applications, [267] introduces the test-time domain adaptation (TTDA), which aims at adapting the model during the testing phase in an online fashion (see Fig. 20 Top). Generally, the existing methods can be divided into two categories: backward-based methods [267–271] and backward-free methods [272–275]. The former category (see Fig. 20 (a)) focuses on optimizing the parameters of models with self-supervision losses, such as entropy loss [267, 270]. In this way, both distribution adaptation and semantic adaptation can be achieved, which however has the following drawbacks. **(1) Low-Efficiency:** Due to the requirement of back-propagation, the computation cost will be multiplied, leading to low efficiency. **(2) Unstable Optimization & Error Accumulation:** Since the gradient is calculated with single sample by weak supervision, the randomness could be high thus leading to unstable optimization. Although this problem can be somehow mitigated by increasing the testing batch size, it still cannot be solved well. In such cases, the accumulated errors may lead the model to forget the original well-learned knowledge and thus cause performance degradation.

The second category aims to adapt the model in the distribution level by updating statistics in batch normalization (BN) [274] layers, which is very efficient as it is directly implemented in forward propagation with a light computation cost. Instance normalization [272] (see Fig. 20 (b)) directly replaces the source statistics with those from each instance, which is sensitive to the target variations due to discarding the basic source knowledge and thus is unstable. Mirza et al [274] (see Fig. 20 (c)) study the impacts of updating the historical statistics by instance statistics with fixed momentum or dynamically fluctuating momentum. However, these methods also suffer from the error accumulation issue caused by abnormal target distributions as well as the neglect of semantic adaptation, both of which will result in inferior adaptation performance.

To this end, we propose a holistic approach (see Fig. 20 (d)), called Dynamically Instance-Guided Adaptation (DIGA), for TTDA-Seg, which takes into account both effectiveness and efficiency. The main idea of DIGA is leveraging each instance to dynamically perform its own adaptation in a non-



Table 9. Comparison with state-of-the-art methods in terms of mIoU. The best score for each column is **highlighted**. CS: CityScapes, BDD: BDD100K, MA: Mapillary, IDD: IDD, CC: Cross-City. *: Use an extra augmented sample during adaptation. Avg.: Mean of mIoUs over five target domains.

Method	GTA5→						Synthia→					
	CS	BDD	MA	IDD	CC	Avg.	CS	BDD	MA	IDD	CC	Avg.
Source [282]	35.87	29.89	38.67	38.05	30.03	34.50	30.87	21.01	31.12	26.23	31.96	28.24
Backward-based Methods												
TENT [267]	37.30	31.53	38.29	38.96	30.59	35.33	34.89	16.99	33.46	26.23	31.68	28.65
EATA [270]	37.08	30.67	39.35	38.75	30.24	35.22	31.31	20.52	31.59	26.46	31.91	28.36
Backward-free Methods												
IN [272]	34.25	29.64	35.01	29.8	23.87	30.51	29.53	19.33	21.92	22.08	28.24	24.22
Momentum [273]	38.12	32.42	40.79	38.74	30.2	36.05	32.84	22.51	31.12	27.24	32.23	29.45
DUA [274]	37.79	31.76	40.26	34.75	26.32	34.18	32.17	21.56	27.42	24.06	29.87	27.02
SITA* [275]	40.64	32.94	37.80	35.66	28.19	35.26	34.63	22.51	26.60	24.64	28.18	27.79
DIGA (Ours)	45.81	35.78	44.25	42.73	33.72	40.46	41.85	29.09	36.54	38.36	36.78	36.52

parametric manner, which is efficient and can largely avoid the error accumulation issue. In addition, our DIGA is implemented in a considerate manner by injecting with distribution adaptation module (DAM) and semantic adaptation module (SAM). Specifically, in DAM, we compute the weighed sum of the source and current statistics in BN layers to adapt target distribution, which enables the model to obtain a more robust representation. In SAM, we build a dynamic non-parametric classifier by mixing the historical prototypes with instance-level prototypes, enabling us to adjust the semantic prediction. In addition, the non-parametric classifier can be associated with the parametric one, which can further benefit the adaptation results. Finally, our method is easy to implement and is model-agnostic, which can be readily injected into existing models (see Fig.21).

6.1.2. Experiments

Datasets. Following the previous works [253,257,275], we evaluate our method on sim2real scenarios. Specifically, for the source model, we pretrain it with two different source domains: GTA5 [276] and Synthia [276]. GTA5 provides 24,971 images from video games with 19 semantic classes. Synthia includes 12,000 simulated images with 16 semantic classes. Performance is evaluated on five target domains: Cityscapes [277], BDD-100K [278], Mapillary [279], IDD [280], Cross-City [281]. We test the results on the validation sets, where the number of samples is {500, 1,000, 2,000, 100, and 400} for {Cityscapes, BDD-100K, Mapillary, IDD, Cross-City}, respectively.

Evaluation. The mean intersection-over-union (mIoU) is used as the evaluation metric. As in [253,257], for source models pretrained on GTA5, we report the mIoU of 19 shared semantic categories. Due to missing of annotations of some classes, we report the mIoU of 16 shared semantic classes for the model pretrained on the Synthia dataset.

Comparison with State of the Art We first compare our method with the state-of-the-art approaches. Generally, the compared methods can be divided into two categories: backward-based methods and backward-free methods.

Backward-based methods: TENT [267] performs adapting by minimizing the output entropy and updating the learnable parameters of BN layers. As an extension to TENT [267], EATA [270] proposes to skip the high-entropy samples and only leverage reliable samples during model optimiza-



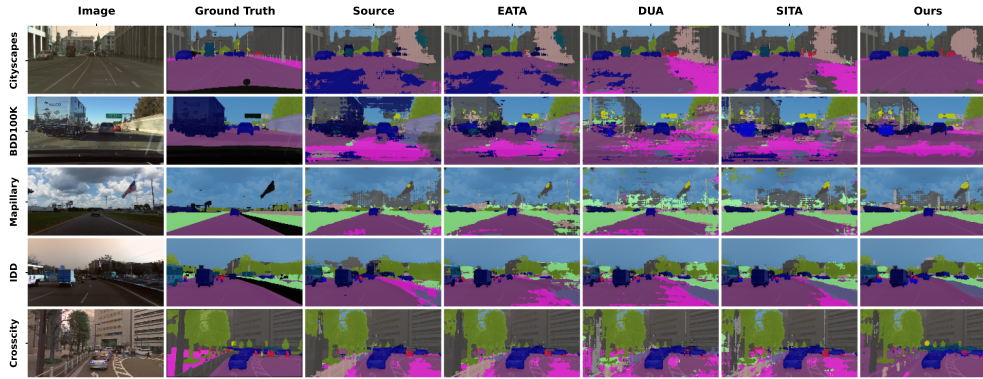


Figure 22. Qualitative comparison of segmentation results.

tion, which can effectively increase testing efficiency. Both of them are initially designed for image classification. We implement them for TTDA-Seg by minimizing the entropy of pixel-level output. For EATA [270], we skip the low-entropy pixels during optimization. *Backward-free methods:* IN [272] uses instance statistics to replace source ones in BN at each testing step. Momentum [273] utilizes the instance statistics to update BN in a momentum-based manner. DUA [274] proposes a decaying strategy to adaptively control the momentum of BN updating. SITA [275] leverages extra augmented samples to obtain stable instance statistics, which are then mixed with the source statistics.

To make a fair comparison, we implement all the methods with the same source models. Note that, we report the results of the compared methods by selecting the best parameters for each source-target pair. In contrast, in our method, we only use one parameter setting for all experiments to better meet the real-world applications.

The following observations can be made from the results reported in Tab. 9. First, backward-based methods can consistently improve the performance when evaluated on CityScapes. However, the improvements on other target domains are limited or even negative. For example, when using Synthia as the source domain, TENT [267] increases the mIoU from 30.87% to 34.89% on CityScapes while largely reduces the mIoU from 21.01% to 16.99% for BDD100K. This indicates that using self-supervision only may not be a good choice for TTDA-Seg. Second, except for IN [272], the backward-free methods are generally effective on CityScapes and BDD100K while failing to achieve consistent improvements on other datasets, even though we have well-tuned them for each target domain. On the other hand, IN [272] largely reduces the average mIoU due to ignoring the source statistics. Third, the proposed DIGA consistently improves the mIoUs of the source models on all settings and outperforms all the compared methods by a large margin in most cases. Specifically, our DIGA is higher than the best competitor (Momentum [273]) by 4.41% and 7.07% in average mIoU for GTA5 and Synthia settings, respectively. In Fig. 22, we provide the qualitative comparison of different methods. It is clear that our DIGA consistently improves the segmentation results of the source model and outperforms other state-of-the-art methods. The above observations demonstrate the effectiveness and universality of the proposed method for solving TTDA-Seg.

6.1.3. Conclusion

Our contributions can be summarized as follows:

- **Efficiency.** We propose a backward-free approach for TTDA-Seg, which can be implemented within one forward propagation with a light computation cost.





- **Effectiveness.** We introduce a considerate approach to adapt the model in both distribution and semantic aspects. In addition, our method takes the mutual advantage of two types of classifiers to achieve further improvements.
- **Usability.** Our method is easy to implement and is model-agnostic, which can be readily injected into existing models.
- **Promising Results.** We conduct experiments on two source domains and five target domains based on driving benchmarks and show that our method produces new state-of-the-art performance for TTDA-Seg. We also study the continual TTDA-Seg and verify the superiority of our method in this challenging task.

6.1.4. Relevant publications

- W. Wang, Z. Zhong, W. Wang, X. Chen, C. Ling, B. Wang, and N. Sebe, Dynamically Instance-Guided Adaptation: A Backward-free Approach for Test-Time Domain Adaptive Semantic Segmentation, CVPR 2023 [283].
Zenodo record: <https://zenodo.org/record/8337069>

6.1.5. Relevant software/datasets/other outcomes

- The Pytorch implementation can be found in <https://github.com/Waybaba/DIGA>

6.1.6. Relevance to AI4Media use cases and media industry applications

Semantic segmentation is an useful tool for providing the first step towards image understanding. Our approach, Dynamically Instance-Guided Adaptation (DIGA) jointly enjoys the effectiveness and efficiency factors. This approach provides a wide applicability to several use cases: (a) by providing solutions to analyze/adapt the visual content, and (b) by providing solutions to discover new visual content and adapt accordingly. These can help to improve tagging and search capabilities.

6.2. Assessing Fine-Grained Modality Alignment in Video-Language Models

Contributing partner: CNR

6.2.1. Introduction and methodology

Video Language Models (VidLMs) have received increasing attention from the research community [284–288]. In principle, VidLMs can visually ground linguistic phenomena which are beyond the reach of image-language models since videos include *dynamically evolving phenomena* (e.g., events, actions, physical processes). Nonetheless, this *temporal dimension* makes learning more complex.

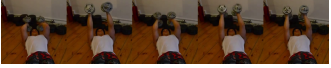




Most efforts to gauge what VidLMs can do rely on *tasks* such as video captioning [289], text-to-video retrieval [290], and video question answering [291]. While such evaluations shed light on task performance and support comparative analysis, they are limited in their ability to reveal the specific visuo-linguistic capabilities that models exhibit *across tasks*.

In this study, we present ViLMA (Video Language Model Assessment), a task-agnostic benchmark that proposes a behavioural evaluation for VidLMs focusing on fine-grained phenomena. We draw inspiration from related benchmarks for Image Language Models (ILMs) [292–294]. However,





Table 10. Overview of data and foiling methods used in each test in ViLMA.

Test (#exs.)	Video Caption / Foil	Foil Generation	Sample Frames
Action Counting (1432)	Someone lifts weights exactly two / five times.	Number replacement	
Situation Awareness (911)	A policeman / blond man holds a blond man / policeman against a wall.	Actor swapping	
	A man in blue holds / chops up a man in green.	Action replacement	
Change of State (998)	Someone folds / unfolds the paper.	Action replacement	
	Initially, the paper is unfolded / folded .	Pre-state replacement	
	At the end, the paper is folded / unfolded .	Post-state replacement	
Rare Actions (1443)	Drilling into / Calling on a phone.	Action replacement	
	Drilling into a phone / wall .	Object replacement	
Spatial Relations (393)	Moving steel glass towards / from the camera.	Relation replacement	

VILMA focuses on tests that require *strong temporal understanding and reasoning*, as time is a unique aspect present in VidLMs but not in ILMs. We adopt a common structure for each *test*: (i) We harvest high-quality examples from existing video-language datasets; (ii) we create counterfactual examples or ‘foils’ [295], so that a *test* requires distinguishing correct from counterfactual video+text pairs; (iii) we create a *proficiency test* to gauge if a model learns the capabilities we deem necessary to solve the main test; (iv) we apply automatic and manual validation of the examples and their counterfactuals to control for biases and to ensure a high-quality evaluation benchmark; (v) finally, we test whether existing VidLMs can solve the proficiency tests and distinguish correct from counterfactual examples in the main tests.

VILMA is designed as a *probing benchmark* divided into five main tests, summarised in Table 10 and described in detail below. It is intended as a zero-shot evaluation benchmark. For each test, we define *specific foiling functions* that target central characteristics of VidLMs, focusing on their *temporal understanding capabilities*.

First, we introduce *proficiency tests* They test criteria that can be considered as *prerequisites for solving the main tests*, by assessing the VidLMs’ capability to successfully navigate and solve simpler objectives before attempting the more demanding main tests. on their *temporal understanding capabilities*.





We then introduce our main tests, which focus on: accurately recognising events that display *temporal regularity/periodicity and recurrence*, i.e., action counting; the recognition of specific *actions or action participants*; the recognition of *action or event subphases*, especially when they induce a change of state; the influence of model biases and frequency effects in VidLM’s understanding of *rare actions*; and distinguishing *spatial relations*, since these often exhibit temporal evolution (e.g. in the case of an object moving *towards* another) and thus alter in their visual appearance over time. Finally, we discuss how we use human validation to guarantee ViLMA’s quality.

Proficiency Tests. Proficiency tests can be considered a preliminary criterion for each of the five main tests below. These tests assess a VidLM’s ability to solve simpler visuo-linguistic tasks that do not require strong temporal modelling, as the main tests do. In contrast, VidLMs are expected to address the primary tests by effectively modelling temporal dynamics. Consequently, foils in the proficiency test are less challenging compared to the main tests, and serve as an additional evaluation criterion. The rationale behind conducting proficiency tests is as follows: When a model can effectively tackle the main test but falls short of passing its corresponding proficiency test, it raises a crucial point of concern. This discrepancy hints that the VidLM may potentially be relying on heuristics that exploit biases inherent within the modalities. These biases, in turn, should presumably be traced back to the early pretraining phase of the models.

Given the individual characteristics of the tests, the proficiency test focuses on specific objectives in each case: For the Spatial Relations, Change of State, and Situation Awareness tests, the aim of the proficiency test is to identify objects mentioned in the captions. On the other hand, in the Action Counting and Rare Actions tests, we shift our attention to action recognition and object existence, respectively.

Action Counting. The Action Counting test probes the ability of models to accurately count the occurrences of actions within a given video input stream. This test requires *spatio-temporal reasoning*, presenting a novel and interesting challenge. To this end, we use the QUVA dataset [296], which comprises 100 videos. Within each video, every occurrence of the target action is annotated with a corresponding frame number that specifies the end of each action. The dataset lacks any textual annotations. Consequently, we curate multiple textual templates per video, incorporating a placeholder for the numerical value (<number>). Our templates incorporate the term *exactly* to indicate precise counting (e.g., someone performs exactly <number> push-ups); cf. [292] for a similar strategy. We avoid overly specific terms, opting for more general descriptors (e.g., *lifting weights* instead of *skull-crushers arm exercise*). A native English speaker checked the manually curated templates and fixed potential syntax errors in them. We replace the number placeholder with the correct numerical value to create captions, and with an incorrect one to create foils. We discard all instances with counts exceeding a predetermined threshold T_c , set at 10.

Situation Awareness. The Situation Awareness test shows how effectively VidLMs grasp the interaction between visual clues and verbal context by testing whether they recognise actors, actions, and their relationships. To this end, we use the VidSitu [297] dataset consisting of 10-second video sequences annotated with information regarding verbs, semantic roles, entity co-references, and event relations. To add captions to this dataset, we use ChatGPT to refine and enhance the template-based sentences generated from the existing annotations. Unlike tests which target verb-argument structure in ILMs, such as SVO-Probes [293] and the verb replacement and actant swap tests in VALSE [292], this video-language task adds a temporal dimension, encapsulating dynamic actions. Unlike static images, videos illustrate unfolding events and track their temporal dynamics via sequences of frames. VidLMs must grasp frame coherence, temporal context, and story structure, assessing the order of occurrences. In contrast, ILMs focus on static imagery with less temporal emphasis. Furthermore, videos introduce audio and motion, which gives the current task broader scope and presents novel challenges for contextual integration. Our Situation Awareness test consists of the Action Replacement and Actor Swapping subtests. Action Replacement tests





whether VidLMs can distinguish various activities, by contrasting phrases that differ only in action verbs. Actor Swapping tests the VidLMs’ ability to recognise the role played by (human) actors in diverse actions, thereby probing the ability to discern the semantic roles of arguments in complex relations.

Change of State. The Change of State test examines the ability of VidLMs (i) to recognise and distinguish different sub-phases of actions, especially those that induce a *change of state (CoS)* of objects or entities involved in it; and (ii) to align the beginning and ending phases of these actions across modalities. Cross-modal alignment of the begin- and end-states of CoS actions is challenging, as they are typically *textually implicit* while being *visually explicit*. We define as *CoS verbs* those verbs that refer to actions that include (or textually imply) an initial situation (or state) that is modified to an outcome situation (or state) (e.g., “*to open (a bottle)*” implies that an initial state of “*(the bottle) being closed*” changes to an outcome state of “*(the bottle) being open*” as a result of an opening action). We further assume that the outcome must differ from the initial state. We collect our target *CoS verbs* starting from a codebase by [298]. While the authors only provide the initial-state for each verb, we expand the list by identifying appropriate outcomes for all actions. Leveraging the list of *CoS verbs* as targets, we collect candidate sentence-video pairs by parsing various multimodal datasets: Something-Something V2 [299], YouCook2 [300], COIN [301], RareAct [302], and STAR [303]. We extract the subject and object from the collected sentences, and generate a caption according to a pre-defined template. We generate foils by replacing one or more sub-phases (action, pre-state or post-state) with their respective opposite expressions. We design four different subtests, in each of which we foil an expression describing a specific element: Action subtest, Pre-state subtest, Post-state subtest, and Reverse subtest, where we swap pre-state and post-state and replace the action with its antonym. This reverses the original linguistic sequence, e.g. turning ‘closed–open–open’ to ‘open–close–closed’, which serves as a linguistically coherent foil for the original action in the video.

Rare Actions. The Rare Actions test probes how well VidLMs identify novel compositions and recognise unusual interactions between human beings and objects. We leverage the RareAct dataset [302] consisting of videos accompanied by action-object pairs describing events within the videos. These action-object pairs are extracted by analysing co-occurrence statistics from the widely used HowTo100M [304] dataset.

To enrich this dataset, we generate simple captions based on the action-object pairs. For instance, given the action-object pair *cut-keyboard*, we create the descriptive caption *cutting a keyboard*. This test offers two subtests: In Action Replacement, we substitute the original action with a more plausible alternative that can be applied to the given object, e.g. *type on* for the previous *keyboard* example. As for Object Replacement, we focus on replacing the object in the action-object pair. For instance, revisiting the previous example, we replace the object *keyboard* with *bread*.

Spatial Relations. The Spatial Relations test focuses on the ability of models to distinguish different spatial and spatio-temporal relations related to the actions carried out in a video (e.g. moving an object ‘*over*’, or ‘*towards*’ another object). It is similar to the relation task introduced in [292], with the notable difference that the model must use temporal information to accomplish the task. We create the foils starting from the Something-Something V2 validation set [299] which contains 174 pre-defined actions with everyday objects. To create a candidate foil, we replace the spatial preposition with an in-distribution alternative, drawn from the set of spatial prepositions in the validation set.

Human Validation. A central requirement for VILMA is to ensure validity, that is, humans should agree that captions are true of the videos, while foils are not. We validated the entire VILMA dataset in two separate stages. For the simpler proficiency tests, we manually checked every video-caption-foil sample, retaining only those in which the foil was unambiguously false





Model	Action Counting			Situ. Awareness			Change of State			Rare Actions			Spatial Relations			Avg. P+T
	P	T	P+T	P	T	P+T	P	T	P+T	P	T	P+T	P	T	P+T	
Random	50.0	50.0	25.0	50.0	38.0	19.0	50.0	50.0	25.0	50.0	50.0	25.0	50.0	50.0	25.0	23.8
GPT-2 [†]	50.3	53.3	27.6	44.6	66.6	31.7	18.0	52.4	10.8	58.4	25.9	17.7	49.1	72.8	43.0	26.2
OPT [†]	56.2	54.6	31.0	51.7	<u>71.4</u>	38.7	23.1	48.0	12.9	59.0	23.9	14.9	59.0	84.7	55.7	30.6
CLIP [‡]	<u>90.5</u>	50.9	46.2	71.0	45.6	33.7	93.0	55.2	<u>52.2</u>	92.7	93.9	87.8	78.6	58.3	44.8	52.9
BLIP2 [‡]	80.9	54.5	43.7	73.4	75.4	55.8	74.5	52.1	38.1	93.8	74.5	70.5	91.1	<u>86.0</u>	<u>79.4</u>	57.5
ClipBERT	56.4	49.6	28.0	54.1	57.0	31.9	63.7	50.0	33.5	43.5	40.7	17.7	39.7	39.8	14.1	25.0
UniVL	73.4	43.6	32.2	52.9	46.7	24.1	81.3	54.3	43.0	77.5	78.0	59.9	62.5	51.7	33.2	38.5
VideoCLIP	79.1	46.4	36.5	61.7	40.4	24.9	49.8	50.8	25.9	84.0	77.8	67.5	67.9	54.7	39.7	38.9
FiT	83.9	52.4	44.6	69.8	40.1	29.2	93.0	52.1	47.8	89.7	89.4	80.7	70.5	51.9	38.7	48.2
CLIP4Clip	91.2	52.3	<u>48.0</u>	<u>73.9</u>	49.1	37.7	<u>94.8</u>	54.1	52.1	83.0	<u>94.1</u>	78.7	79.8	56.7	44.2	52.1
VIOLET	79.6	50.6	36.5	70.3	44.5	32.5	88.2	54.6	49.1	87.1	86.6	74.6	73.3	50.4	38.7	46.3
X-CLIP	84.1	55.1	46.4	63.6	44.9	31.1	85.7	52.7	46.0	83.9	85.7	72.3	74.8	56.2	43.5	47.8
MCQ	81.4	50.4	41.5	67.1	37.1	26.3	90.3	50.3	45.3	91.3	88.7	82.3	79.4	48.9	39.4	47.0
Singularity	79.6	51.1	41.5	68.8	40.9	30.2	92.8	54.6	50.3	92.7	88.4	83.1	80.7	46.8	38.9	48.8
UniPerceiver	50.6	46.4	23.0	51.5	42.2	21.2	67.5	46.1	29.1	58.2	58.8	34.7	45.5	48.0	20.1	25.6
Merlot Reserve	84.2	<u>56.0</u>	46.9	70.6	35.7	25.4	93.4	53.6	50.4	83.8	90.6	77.6	63.1	41.9	29.2	45.9
VindLU	84.5	51.2	43.5	70.6	41.6	31.3	85.4	52.6	45.6	<u>94.2</u>	93.1	<u>88.0</u>	83.2	45.6	39.4	49.5
InternVideo	90.2	54.3	48.7	71.6	41.1	29.5	95.6	<u>57.7</u>	55.1	95.6	96.7	92.7	76.6	59.8	45.3	54.2
mPLUG-2	57.7	49.7	27.7	49.6	37.4	21.5	39.5	47.7	20.8	50.8	47.0	24.0	46.6	48.1	26.5	24.1
Otter	59.4	52.7	30.7	58.8	51.0	29.3	65.7	53.0	34.3	56.1	58.8	35.6	62.9	71.3	47.6	35.5
Video-LLaMA	84.6	56.3	47.3	78.2	67.0	<u>54.0</u>	81.4	59.0	46.8	78.7	71.0	58.6	<u>88.6</u>	88.8	79.6	<u>57.3</u>

Table 11. Pairwise ranking accuracy (acc_r) performance of 12 Video-Language Models on the ViLMA benchmark on the proficiency (P), main (T), and combined (P+T) tasks. In the combined task P+T, a success in T only counts if P is also successful. The final column Avg. is the taskwise average of combined scores P+T among each task. Best (second-best) model per metric are marked in boldface (underlined).

with respect to the input video. This resulted in the removal of 1278 (15.11%) of samples in the proficiency tests. The main tests were validated independently, in a study conducted on AMTurk. Each sample was evaluated by three independent annotators, who were asked to judge which text (caption or foil), if any, was true of the video. We retained only samples for which at least two out of three independent annotators judged only the caption as true of the video, resulting in a final set of 5177 (61.19%) of the initial set.

6.2.2. Experiments

We analyse the performance of 12 architecturally diverse, state-of-the-art VidLMs: ClipBERT [305], UniVL [306], VideoCLIP [307], FiT [284], CLIP4Clip [308], VIOLET [309], X-CLIP [310], MCQ [285], Singularity [286], UniPerceiver [287], Merlot Reserve [311], VindLU [288], InternVideo [312], mPLUG-2 [313], Otter [314] and Video-LLaMA [315]. The models were trained on different tasks and data. We also benchmark two commonly used ILMs: CLIP [114] and BLIP-2 [316], alongside two unimodal baselines: GPT-2 [317] and OPT [318].

Unimodal Results. The unimodal baselines perform close to the random baseline in Counting and Change of State, but not in the remaining tests. In Rare Actions, this outcome is expected given that the captions inherently describe *less likely events*. Similarly, within the proficiency test for the Change of State, we introduce the foiling of low-frequency nouns (e.g., hyponyms) with





high-frequency ones (e.g., hypernyms), which inadvertently biases the model towards favouring the foils. In contrast, unimodal models exhibit a notably superior performance compared to the random baseline in Situation Awareness and Spatial Relations. This can be partially attributed to *plausibility biases* [292, 319] introduced during foil generation. The shared linguistic context between the caption and foil constrains the selection of foiling actions/relations, often leading to the introduction of unlikely or unnatural alternatives.

Image-Language Model Results. Much like the unimodal baselines, the performance of ILMs in the Counting and Change of State tasks is close to random. However, we note that ILMs exhibit proficiency in detecting objects and capturing semantics, as shown in the proficiency test results for Rare Actions and Counting, where the former requires object detection capabilities, and the latter hinges on precise action recognition. In several tasks, ILMs even outperform their VidLM counterparts. For instance, BLIP2 is the best-performing model in Situation Awareness, while in the Rare Actions task, CLIP performs better than all the other models excluding VindLU.

Video-Language Model Results. In the majority of tasks, VidLMs deliver performance levels that closely resemble those of ILMs. This observation raises a critical point: the temporal reasoning capabilities of current VidLMs are evidently far from adequate. Remarkably, in the Counting, Situation Awareness, and Change of State tests, many VidLMs do not show a notably higher performance than the random baseline. Our findings highlight the urgent need for the community to prioritise and enhance the temporal reasoning abilities of these models.

Proficiency Results. The results reveal that both ILMs and VidLMs tend to consistently perform better in the simpler proficiency test, with few exceptions. These tests provide valuable insights by enabling a more robust evaluation of models. An intriguing insight emerges from the evaluation of models in the combined setting, where a striking performance drop occurs. This suggests that in a substantial number of cases, when models predict correct answers in the main tasks, they do so by chance or due to reliance on spurious features, rather than due to a robust understanding of the input.

6.2.3. Conclusion

We introduced *ViLMA*, a video-language foiling benchmark, which probes the capabilities of pretrained VidLMs where both commonsense and temporal reasoning take centre-stage. We have conducted a comprehensive evaluation and comparison of numerous VidLMs as well as ILMs and text-only LMs on our benchmark. Our experiments show that, as far as visually grounded temporal reasoning abilities are concerned, VidLMs do not differ substantially from ILMs. To further refine our benchmark, we introduced proficiency tests, which not only enhance granularity but also provide deeper insights into the models' aptitude. Strikingly, our proficiency task results reveal that a considerable portion of correct predictions appears to be accidental rather than indicative of robust understanding. This highlights that current VidLMs struggle with the intricacies of temporal reasoning. It also underlines the importance of benchmarks like *ViLMA* to identify weaknesses of current VidLMs that need improvement.

6.2.4. Relevant Publications

- I. Kesen, A. Pedrotti, M. Dogan, M. Cafagna, E. C. Acikgoz, L. Parcalabescu, I. Calixto, A. Frank, A. Gatt, A. Erdem and E. Erdem, *ViLMA: A Zero-Shot Benchmark for Linguistic and Temporal Grounding in Video-Language Models*, The Twelfth International Conference on Learning Representations, ICLR 2024 [320] <https://openreview.net/forum?id=liuqDwmbQJ>.





6.2.5. Relevant software/datasets/other outcomes

- The benchmark is archived at <https://github.com/ilkerkesen/ViLMA>

6.2.6. Relevance to AI4Media use cases and media industry applications

This development has not impacted any of the AI4Media use cases since it is very recent. However, it has a potential to influence many media industry applications since it will allow to comparatively evaluate the behaviour, on these applications, of different Video-Language models





7. Neural Architecture Search (Task 3.4)

Contributing partner: UNITN

Deep learning methods are very successful in solving tasks in machine translation, image and speech recognition. However, the search for suitable architectures is a time-consuming, arduous, and error-prone task. The research in this task has focused on transfer neural architecture search (TNAS) which leverages past search experiences on different datasets to accelerate a new search. This overcomes the big problem of traditional NAS methods which start for every new search from scratch. Specifically, we focused on how to leverage parameter sharing or differentiable architecture search in the scope of TNAS and explore its applicability for multi-objective NAS and tasks beyond image classification.

7.1. Budget-Aware Pruning for Multi-Domain Learning

Contributing partner: UNITN

7.1.1. Introduction and methodology

Deep learning has brought astonishing advances to computer vision, being used in several application domains, such as medical imaging [321], autonomous driving [322], road surveillance [323], and many others. However, to increase the performance of such methods, increasingly deeper architectures have been used [324], leading to models with a high computational cost. Also, for each new domain (or task to be addressed), a new model is usually needed [325]. The significant amount of model parameters to be stored and the high GPU processing power required for using such models can prevent their deployment in computationally limited devices, like mobile phones and embedded devices [326–328]. Therefore, specialized optimizations at both software and hardware levels are imperative for developing efficient and effective deep learning-based solutions [329].

For these reasons, there has been a growing interest in the Multi-Domain Learning (MDL) problem. The basis of this approach is the observation that, although the domains can be very different, it is still possible that they share a significant amount of low and mid-level visual patterns [330]. Therefore, to tackle this problem, a common goal is to learn a single compact model that performs well in several domains while sharing the majority of the parameters among them with only a few domain-specific ones. This reduces the cost of having to store and learn a whole new model for each new domain.

Berriel et al. [325] point out that one limitation of those methods is that, when handling multiple domains, their number of parameters is at best equal to the backbone model for a single domain. Therefore, they are not capable of adapting their amount of parameters to custom hardware constraints or user-defined budgets. To address this issue, they proposed the modules named Budget-Aware Adapters (BA^2) that were designed to be added to a pre-trained model to allow them to handle new domains and to limit the network complexity according to a user-defined budget. They act as switches, selecting the convolutional channels that will be used in each domain. However, as mentioned in [325], although the use of this method reduces the number of parameters required for each domain, the entire model is still required at test time if it aims to handle all the domains. The main reason is that they share few parameters among the domains, which forces loading all potentially needed parameters for all the domains of interest.

This work builds upon the BA^2 [325] by encouraging multiple domains to share convolutional filters, enabling us to prune weights not used by any of the domains at test time. Therefore, it is possible to create a single model with lower computational complexity and fewer parameters



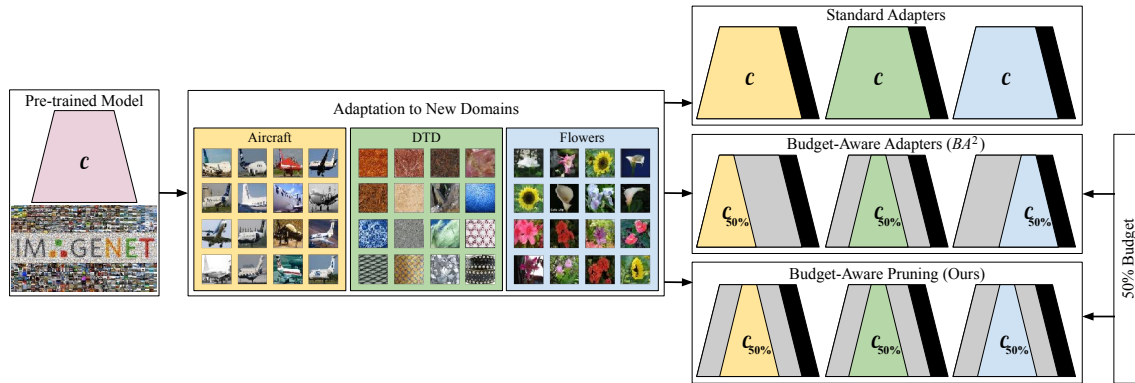


Figure 23. In standard adapters, the amount of parameters from the domain-specific models (indicated in colored C) is equal to or greater than the backbone model (due to the mask represented in black). Budget-Aware Adapters can reduce the number of parameters required for each domain (unused parameters are denoted in gray). However, the whole model is needed at test time if handling distinct domains (colored areas share few parameters). Our model encourages different domains to use the same parameters (colored areas share most of the parameters). Thus, when handling multi-domains at test time, the unused parameters can be pruned without affecting the domains.

than the baseline model for a single domain. Such a model is capable of better fitting the budget of users with limited access to computational resources. Figure 23 shows an overview of the problem addressed by our method, comparing it to previous MDL solutions and emphasizing their limitations. As it can be seen, standard adapters use the entire model, while BA^2 [325] reduces the number of parameters used in each domain, but requiring a different set of parameters per domain. Therefore, the entire model is needed for handling all the domains together and nothing can be effectively pruned. On the other hand, our approach increases the probability of using a similar set of parameters for all the domains. In this way, the parameters that are not used for any of the domains can be pruned at test time. These compact models have a lower number of parameters and computational complexity than the original backbone model, which facilitates their use in resource-limited environments. To enable the generation of the compact models, we propose a novel loss function that encourages the sharing of convolutional features among distinct domains. An overview of our strategy for sharing parameters among domains is presented in Figure 24.

Our proposed approach was evaluated on two well-known benchmarks, the Visual Decathlon Challenge [330], comprised of 10 different image domains, and the ImageNet-to-Sketch setting, with 6 diverse image domains. Results show that our proposed loss function is essential to encourage parameter sharing among domains, since without direct encouragement, the sharing of parameters tends to be low. In addition, results also show that our approach is comparable to the state-of-the-art methods in terms of classification accuracy, with the advantage of having considerably lower computational complexity and number of parameters than the backbone.

7.1.2. Experiments

Our approach was validated on two well-known MDL benchmarks, the Visual Decathlon Challenge [330], and the ImageNet-to-Sketch. For the sake of space, we present here only the results obtained on the Visual Decathlon Challenge.

The Visual Decathlon Challenge comprises classification tasks on ten diverse well-known image datasets from different visual domains: ImageNet, Aircraft, CIFAR-100, Daimler Pedestrian (DPed), Describable Textures (DTD), German Traffic Signs (GTSR), VGG-Flowers, Omniglot, SVHN, and UCF-101. Such visual domains are very different from each other, ranging from people, objects,



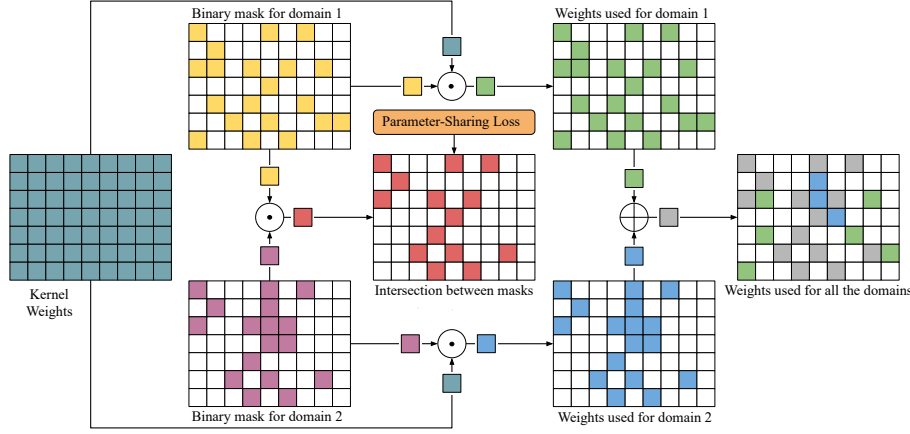


Figure 24. Overview of our strategy for sharing parameters among domains. Colors represent data (i.e., weights, masks, etc), therefore, the colored squares denote the input data for each operation as well as its resulting output.

and plants to textural images.

In order to evaluate the classification performance, we use the accuracy on each domain, and the S-score [330] metric. Proposed by Rebuffi et al. [330], the S-score metric rewards methods that have good performance over all the domains compared to a baseline, and it is given by Equation 5:

$$S = \sum_{d=1}^N \alpha_d \max\{0, Err_d^{max} - Err_d\}^{\gamma_d} \quad (5)$$

where Err_d is the classification error obtained on the dataset d , Err_d^{max} is the maximum allowed error from which points are no longer added to the score and γ_d is a coefficient to ensure that the maximum possible S score is 10.000 [330].

To assess the computational cost of a model, we considered its amount of parameters and computational complexity. For the number of parameters, we measured their memory usage, excluding the classifier and encoding float numbers in 32 bits and the mask switches in 1 bit. For the computational complexity, we used the THOP library to calculate the amount of multiply-accumulate operations (MACs⁴) for our approach, while we reported the values from [325] for their work. All reported values are relative to the backbone size, as in [325]. Similar to [325], in order to assess the trade-off between effectiveness on the MDL problem and computational efficiency, we consider two variations of the S score, named as S_O , which is the S score per operation; and S_P , the S score per parameter.

After obtaining the best hyperparameter configuration, we compared our work to the baseline strategies of using the pre-trained model as a feature extractor, only training the classifier (named feature), and finetuning one model for each domain (finetune). We also compared to the state-of-the-art method BA², since it is one of the only works that take into consideration computational cost constraints. The main focus of our work is the scenario where there is a budget set by the user, and other works except BA² do not take into consideration this restriction. Despite the lack of attention that tackling multi-domain learning with budget restrictions has received, it is a promising topic that is paramount for the application of these methods in environments with limited computational power. Experiments were run using V100 and GTX 1080 TI NVIDIA GPUs, Ubuntu 20.04 distribution, CUDA 11.6, and PyTorch 1.12. After obtaining the best hyperparameter

⁴We follow Berriel et al. [325] and report results in FLOPs (1 MAC = 2 FLOPs).



configuration, the model was trained on both training and validation sets and evaluated on the test set of the Visual Domain Decathlon. The comparison of the results with baseline strategies and a state-of-the-art method, BA², is shown in Table 12.

Table 12. Computational complexity, number of parameters, accuracy per domain, S , S_O and S_P scores on the Visual Domain Decathlon, best in bold, second best underlined

Method	FLOP	Params	ImNet	Airc.	C100	DPed	DTD	GTSR	Flwr.	Oglt.	SVHN	UCF	S-score	S_O	S_P
Baselines [330]:															
Feature	1.000	1.00	<u>59.7</u>	23.3	63.1	80.3	45.4	68.2	73.7	58.8	43.5	26.8	544	544	544
Finetune	1.000	10.0	59.9	60.3	82.1	92.8	55.5	97.5	81.4	87.7	96.6	51.2	2500	2500	250
BA ² [325]:															
$\beta = 1.00$	0.646	1.03	56.9	<u>49.9</u>	78.1	<u>95.5</u>	55.1	99.4	86.1	88.7	96.9	<u>50.2</u>	3199	4952	3106
$\beta = 0.75$	0.612	1.03	56.9	47.0	78.4	95.3	55.0	<u>99.2</u>	<u>85.6</u>	<u>88.8</u>	<u>96.8</u>	48.7	<u>3063</u>	5005	2974
$\beta = 0.50$	0.543	1.03	56.9	45.7	76.6	95.0	55.2	99.4	83.3	88.9	96.9	46.8	2999	5523	2912
$\beta = 0.25$	<u>0.325</u>	1.03	56.9	42.2	71.0	93.4	52.4	99.1	82.0	88.5	96.9	43.9	2538	<u>7809</u>	2464
Ours:															
$\beta = 1.00$	0.837	1.03	56.9	37.3	<u>80.2</u>	95.1	57.9	98.6	84.6	83.8	96.0	45.8	2512	3001	2438
$\beta = 0.75$	0.645	0.921	56.9	42.6	75.3	95.0	<u>56.1</u>	98.6	82.8	87.2	96.0	44.7	2444	3789	2654
$\beta = 0.50$	0.447	<u>0.783</u>	56.9	42.1	73.7	96.8	51.3	98.7	81.4	87.1	96.1	45.4	2552	5709	<u>3259</u>
$\beta = 0.25$	0.238	0.531	56.9	33.6	67.9	95.3	44.9	98.2	75.1	87.4	96.1	43.0	1942	8159	3657

Compared to the baseline strategies, our method was able to vastly outperform the feature extractor only, while achieving similar S-score to finetune for the budgets of $\beta = 1.0$, 0.75 and 0.50, but with almost 10 times less parameters.

Compared to BA², we obtained similar accuracy in most domains, but faced some drops in accuracy in some domains compared to [325]. We believe the main reason for this drop in accuracy is the simultaneous training procedure, as we observed similar drop when switching from individual to simultaneous training without the addition of our loss function, but we kept it since it is necessary to enable parameter sharing. The domains with the biggest accuracy drops were the smaller datasets, like aircraft, DTD, VGG-Flowers, and UCF-101. Other works, like Rebuffi et al. [330, 331] also mention subpar performance on these datasets, identifying the problem of overfitting.

The S-score also dropped up to 687 points for the same issues. The drop is harsher since the metric was designed to reward good performance across all datasets, and the small datasets we mentioned had a subpar performance. Despite facing small drops in accuracy and S-score, our method offers a good trade-off between classification performance and computational cost.

When comparing computational complexity (FLOP on Table 12), for the budgets of $\beta = 1$ and 0.75, the original BA² had lower values, but for the harsher budgets of $\beta = 0.5$ and 0.25, our methods obtained the lower complexity. This happens due to the fact that the original BA² tends to discard more weights than the demanded when the budget is higher, while our methods tend to stay closer to the amount defined by the budget. It also must be noted that all our methods obtained lower complexity than the value defined by the budget, showing that it is a great tool to adapt a backbone model to the resources available to the user.

By comparing the S_O metric, we can observe that both methods have a good trade-off between computational complexity and S-score, as this metric greatly increases as the budget is reduced, showing that the reduction in computational complexity is considerably greater than the loss in S-score. As expected, our method had better S_O for the harsher budgets of $\beta = 0.50$ and 0.25 while BA² achieved superior results on the budgets of $\beta = 1.00$ and 0.75.

The main advantage of our proposed method is the reduction on the number of parameters of the model, as it is, to our knowledge, one of the few methods that is capable of tackling the problem of multiple-domain learning, while also reducing the number of parameters in relation to the backbone model. Other methods can reduce the amount of parameters for a single domain,



but since the parameters are not shared, to handle all of them during test time, the entire model must be kept. As we can see (column Params of Table 12), the original BA² had similar amount of parameters to the backbone model, being 3% more for all budgets. For the budget of $\beta = 1.00$, we obtained the same result, while for the budget of $\beta = 0.75$ we reduce the amount of parameters compared to the backbone model in 7.9%, for budget $\beta = 0.50$ the reduction was of 22.7% and for budget $\beta = 0.25$ there were 49.9% less parameters. These results show that our method was successful in encouraging the sharing of parameters among domains and that this approach can lead to considerable reductions on the amount of parameters of the network. The S_P metric also shows this result, as for the budgets of $\beta = 0.50$ and 0.25 our method was able to outperform BA² by considerably reducing the amount of parameters.

7.1.3. Conclusion

In this research, we addressed the multi-domain learning problem while taking into account a user-defined budget for computational resources, a scenario addressed by few works, but of vital importance for devices with limited computational power. We proposed to prune a single model for multiple domains, making it more compact and efficient. To do so, we encouraged the sharing of parameters among domains, allowing us to prune the weights that are not used in any of them, reducing both the computational complexity and the number of parameters to values lower than the original baseline for a single domain. Performance-wise, our results were competitive with other state-of-the-art methods while offering good trade-offs between classification performance and computational cost according to the user's needs. In future work, we intend to evaluate different strategies for encouraging parameter sharing, and test our method on different network models and benchmarks.

7.1.4. Relevant publications

- S. Felipe dos Santos, R. Berriel, T. Oliveira-Santos, N. Sebe, and J. Almeida, Budget-Aware Pruning for Multi-Domain Learning, International Conference on Image Analysis and Processing ICIAP 2023 [332].
Zenodo record: <https://zenodo.org/record/8337417>

7.1.5. Relevance to AI4media use cases and media industry applications

Our research addresses the multi-domain learning problem while taking into account a user-defined budget for computational resources, a scenario of vital importance for devices with limited computational power. This approach provides a wide applicability to several use cases: (a) by providing efficient solutions to analyze the visual content, and (b) by providing solutions to efficiently discover new visual content.





8. AI at the Edge, decentralised and distributed learning (Task 3.5)

Contributing partners: CERTH, JR

T3.5 focuses on the application of AI directly on edge devices and servers, for both model inference and training. This is in contrast to the current dominant paradigm of deployment at centralized infrastructures. AI at the edge is attractive due to the increased costs of aggregating computational resources and data at cloud infrastructures, as well as the privacy and confidentiality requirements of end user data. On the other hand, it brings challenges related to device heterogeneity and limitations of resources like bandwidth, memory, processing power, and energy [333–336], which are also time-variable [337–342].

Based on the above, our research spans three axes. *a) Collaborative learning*, which includes distributed, federated, and gossip learning [343–346] and which distributes the learning process of an AI model across multiple devices. *b) Model compression* [347–349], which reduces the size and execution time of AI models to fit on end devices, possibly with a tolerable sacrifice in accuracy. *c) In-device processing*, which further facilitates the execution of AI operations at end devices.

8.1. Frameworks for graph analysis and learning at the edge

Contributing partner: CERTH

8.1.1. Introduction an methodology

Organizing relational data in graphs is a popular paradigm for analysis and learning. Graph algorithms can be computationally efficient when edges linking nodes are stored in sparse matrix formats; various operations can be written as sparse-dense matrix multiplications that are executed in times proportional to the number of edges (instead of the square of the number of nodes). This efficiency is ideal for training AI models on edge devices of limited processing and memory resources, like smartphones and systems-on-chips (e.g., Raspberry PI [350,351]). We thus create computing frameworks that simplify implementation of graph algorithms in Java (Paragraph 8.1.1.1) and Python (Paragraph 8.1.1.2). The first framework supports GNNs and overcomes Java’s verbosity by providing a scripting language for AI model declarations. The second framework is also accompanied by fast auto-tuning strategies for algorithm selection on-the-fly when analyzing local device data.

8.1.1.1. JGNN: GNNs at the edge GNNs start from representation matrices whose rows are either node features or end-to-end trained embeddings, define operations that pool (e.g., average) neighbor representations, combine them with existing representations, and pass the combination through dense transformations. These concepts are often expressed in the family of message-passing GNNs [352, 353], whose operations are often expressed in matrix form and engage symmetrically normalized or unnormalized versions of binary graph adjacency matrices. The combination mechanism is typically a row-wise concatenation, but a simplification is to not concatenate the outcomes of aggregation with the node’s features, and instead apply the renormalization trick [354] that adds numerical stability with self-loops to all nodes. Some approaches focus on making architectures decouple training from message passing operations [355], be deeper [356], account for heterophily (dissimilar nodes linking to each other) [357], apply neighbor attention [358,359], or create variations of stronger expressive power [360–362].

We enable training and deployment of machine learning architectures (including GNNs) in native Java - a language broadly supported by edge devices - with a library called JGNN. In this,





core primitives (vectors and matrices) support common arithmetic operations, such as element-by-element addition and multiplication and in-place variations that save on memory allocation. Sparse and dense data are made interoperable by internally determining the type of arithmetic operation results. For example, addition involving dense vectors yields dense outputs, whereas the type of matrix multiplication outcome is determined via stochastic estimation of the result's expected density. We also apply optimizations like iterating over only non-zero sparse data elements, object reuse, and transparent access to parts of data structures. For example, the expression $M.accessRow(0).asColumn()$ accesses the first row of a matrix M and turns it into a column matrix without any data copying.

Core primitives are used to implement machine learning components that can be combined via Java code to define complex architectures, such as GNNs for node and graph classification. Training these architectures involves parsing (batch) inputs, desired outputs, loss functions, and optimizers, running over many epochs, and eventually selecting parameters based on validation data. JGNN provides helper classes to define and run such procedures while leveraging multiple processors and Single Instructions per Multiple Data (SIMD) optimizations, if available. To aid debugging, named dimensions are supported and automatically extrapolated for operation outcomes, whereas architectures can print the routing between components, export execution graphs into graphviz format for visualization [363], check for unused computational branches, and fill missing dimension names of parameters. Error messages for incompatible dimension sizes or names also point to the neural component where the issue occurred, and its current state and inputs.

To avoid hand-wiring operations, and therefore reduce errors and simplify usage patterns of JGNN, we also created a scripting language for architecture declarations and respective parsers. The language is called *NeuraLang* and offers two main features: a) functional declaration of layers, and b) inline declaration of learnable parameters. It is not Turing-complete because it does not support unbounded loops (it does not support architectures with unbounded computational complexity). Figure 25 shows a NeuraLang definition of the Graph Convolutional Network (GCN) architecture [354] and the boilerplate Java code loading a function from a file as a JGNN model. All function calls correspond to equivalent numeric operations, with the difference that *matrix* and *vector* declare learnable parameters; *matrix*'s arguments define the dimensions and loss regularization constant. Keyword arguments correspond to hyperparameters with defaults annotated with “:” that can be overwritten upon function call. The *extern* default is retrieved from the importing Java code. Dimension sizes that can be automatically inferred by the *autosize* method of the JGNN parser are annotated with “?”.

<pre> fn gcnlayer(A, h, hidden: 64, reg: 0.005) { return relu(A@h@matrix(?, hidden, reg) + vector(hidden)); } fn gcn(A, h, classes: extern) { h = gcnlayer(A, h); h = gcnlayer(A, h, hidden: classes); return h; } </pre>	<pre> Dataset dataset = new Cora(); dataset.graph().setMainDiagonal(1).setToSymmetricNormalization(); ModelBuilder modelBuilder = new Neuralang() .parse(Paths.get("../architectures/gcn")) .constant("A", dataset.graph()) .constant("h", dataset.features()) .var("nodes") .config("classes", dataset.labels().getCols()) .config("hidden", 16) .out("classify(nodes, gcn(A,h))") .autosize(new EmptyTensor(dataset.samples().getSlice(0).size())); ModelTraining trainer = new ModelTraining() .configFrom(modelBuilder) .setVerbose(true) .setLoss(new CategoricalCrossEntropy()) .setValidationLoss(new CategoricalCrossEntropy()); </pre>
--	---

Figure 25. NeuraLang code (left) and the JGNN code that prepares to train it on a dataset (right).

8.1.1.2. Pygrank: node ranking in large graphs Node ranking algorithms build on the notion of graph signals, which map graph nodes to corresponding non-negative scores. Signal priors hold personalized node information, such as probabilities that nodes are members of metadata





communities [364–367]. Non-personalized variations set the same prior scores to all nodes. For normalized adjacency matrices, graph filters [368–370] yield new posterior signals that score how proximate nodes are to high-scored priors. Proximity is defined either by modeling ad-hoc signal diffusion strategies (e.g., random walks with restart [371]) or by examining the impact of filters on the adjacency matrix’s eigenvalue spectrum [369]. Filters are parameterized by weighing an aggregation of k -hop signal propagations. Two well-known filters are personalized PageRank [372, 373] and heat kernels [374]. Based on the theoretical properties, filter inputs may be pre-processed and outcomes post-processed to convert structural proximity scores to predictions about node classification [375], community detection [376], or link prediction. Different base filters or many rounds of post-processing may also be applied to improve qualitative characteristics first [367], each improving different facets of structural proximity assumptions. For example, graph filters that place higher importance to more hops away are needed to identify large structural communities, whereas the type of normalization is often chosen to reflect graph properties. Thus, organized experimentation is required to select the best base node ranking algorithms and post-processors to be applied in each domain or application.

To simplify usage of node ranking algorithms in many settings we created a Python package called *pygrank*. In this, graph signals behave like practical data structures (e.g., can be constructed from and accessed as hash maps between nodes and scores) but internally hold computationally efficient backend primitives to keep track of scores and exchange them fast. Graph filters are initialized from their types and parameters, such as the type of graph normalization (including renormalization [354]), caching strategies, and whether Lanczos approximations [377] are used for speedup. Filters or more complex algorithms are used as callables with graph signal inputs and outputs. Post-processors are similarly created and attached on existing algorithms. Supervised and unsupervised measures assess node score quality on various tasks by implementing stochastic generalizations that handle binary posterior scores (if thresholded with appropriate post-processors) or scores if needed. For experimentation, helper methods load graph datasets following the SNAP format [378], download and prepare certain datasets from online sources, perform training-test splits as masks applied on signals, and even automate GNN training for node classification. A benchmarking interface compares multiple algorithms across many datasets with many training-test split ratios and under designated measures.

To support a broad range of hardware, and thus deployment of graph analysis algorithms in many types of edge computing that support Python, *pygrank* can run on several backends, which refer to ecosystems of libraries that use different hardware and resources. Interoperability (up to each backend’s numerical tolerance) is achieved by writing all algorithms with the same base operations and creating an implementation of the latter in each backend. In addition to efficient implementations based on *numpy* and code optimized for multithreaded analysis of graphs with few edges with the *matvec* library that we developed, there is support for GPU computing frameworks like *pytorch* and *tensorflow* that may also switch between fast dense representations of adjacency matrices, where sparsity is a little slower but memory efficient. A final backend relies on the *dask* framework for distributed computation across several devices.

We have also investigated the option of constructing node ranking algorithms on-the-fly, to make graph-based algorithms dynamically adapt on the data. To achieve this, we mask available personalization to create “training” graph signals p_{train} to be used as filter inputs and output validation signals p_{valid} , and employ supervised evaluation measures $\mathcal{M}(\cdot, \cdot)$ to find algorithms with high $\mathcal{M}(r_{train}, p_{valid})$ for node scores r_{train} created from p_{train} ; in the simplest case, we would tune graph filters $H(\hat{A})$ of normalized adjacency matrices \hat{A} , for which $r_{train} = H(\hat{A})p_{train}$. To avoid overfitting, M is computed only across nodes with zero values at p_{train} . For graphs that exhibit homogeneous correlations between edges and ideal posteriors (e.g., between edges and communities), node scores maximizing validation evaluation are expected to also maximize $\mathcal{M}(r, p_{test})$ on the





non-training nodes, where p_{test} are unknown ideal test scores.

We follow two algorithm selection strategies, both of which require the computation of multiple node score predictions runs that pygrank can quickly perform. First is to select among a list of popular node ranking algorithms. The second strategy starts with a parameterized form of node ranking algorithms and tunes a vector of hyperparameters $h = [h_0, h_1, \dots, h_K]^T$. For filters, we explore non-negative correlations between hops and node ranks and, without loss of generality, tune hop weights in the range $[0, 1]$. Generic black box optimization algorithms [379, 380] do not guarantee convergence, and we work with potentially non-convex objectives where adjusting one hyperparameter could drastically affect the validity of rest. Thus, in Algorithm 1 we created a novel tuning strategy that maintains a broad parameter search space while converging in finite time.

Algorithm 1 Parameter tuning for node ranking algorithms

Inputs: parameter loss $\ell(h)$, tolerance ϵ , line search partitions P , range shrinking T
Outputs: near-optimal vector of K parameters
 $h \leftarrow [0.5] \times K$, $\Delta h \leftarrow [0.5] \times K$, $err \leftarrow [\infty] \times K$, $i \leftarrow 0$
while $\max_i err[i] > \epsilon$ **do**
 $ateu_i \leftarrow$ unit vector with $u_i[j] = \{1 \text{ if } i = j, 0 \text{ otherwise}\}$
 $H_{search} \leftarrow \{max(0, min(1, h + u_i \cdot \Delta h[i] \cdot (p/P - 1))) \mid p = 0, 1, \dots, 2P\}$
 $err[i] \leftarrow \ell(h) - \min_{h \in H_{search}} \ell(h)$
 $h \leftarrow \arg \min_{h \in H_{search}} \ell(h)$
 $\Delta h[i] \leftarrow \Delta h[i]/T$
 $i \leftarrow (i + 1) \bmod K$
end while
return h

This algorithm cycles through parameters and progressively minimizes a loss function $\ell(h) = 1 - \mathcal{M}$ by finding the best permutation with coarse linear search. As tuning progresses, we shrink the search range [381] so that small permutations around ideal values are eventually found, effectively moving the center of the selected range around each parameter [382] based on subsequent parameters. If shrinking is slow enough, by the time when parameter permutation breadths become small, combinations with drastically different permutations of other parameters have already been considered. For connected graphs and L -Lipschitz loss $\ell(h)$, the amortized running time is:

$$O(K^2 E (\log_T L - \log_T \epsilon))$$

where E is the number of edges, ϵ the desired tolerance, and K the number of parameters. We set defaults $P = 2, T = 1.01$, which suffice to minimize the Beale and Booth functions often used in optimization benchmarks [383] to 10^{-6} parameter tolerance.

8.1.2. Results

8.1.2.1. JGNN resource consumption JGNN combines the advantages of non-collaborative (autonomous if needed) libraries of similar capabilities [384–387] that cannot train GNNs at the edge individually. In particular, it runs on Java, supports training at the edge, provides deep learning components, parses large graphs with sparse matrices, and declares custom architectures. For validation, we train the GCN [354] and APPNP [375] architectures for node classification within edge devices on the well-known Cora dataset [388] of 2,708 nodes and 57,884 edges. We also train lightweight architecture variations that reduce layer dimensions to the number of output classes.

Table 13 shows resource consumption and test set accuracy on a 60-20-20 train-validation-test split after 300 epochs compared to GPU Tensorflow implementations in Python that also use sparse





matrices. Everything runs on a 2.6 GHz CPU and DDR3 RAM, whereas Tensorflow also leverages a 1.2 GHz GPU with 1,920 cores and DDR6 memory. Implementations exhibit similar accuracy and we attribute differences to JGNN running on 64-bit (instead of 32-bit) arithmetics. Overall, JGNN runs in less than 70 times the memory needed by Tensorflow, and architectures with reduced parameters are trained in time comparable to GPUs with minimal accuracy drop.

GNN	Tensorflow			JGNN			JGNN reduced		
	Acc	Time	Memory	Acc	Time	Memory	Acc	Time	Memory
GCN	76.8%	26sec	2,560MB	87.5%	249sec	40MB	87.5%	85sec	14MB
APPNP	89.1%	43sec	2,304MB	88.2%	306sec	40MB	87.8%	117sec	28MB

Table 13. GNN training resources on the Cora dataset.

8.1.2.2. Improvements from auto-tuning To evaluate auto-tuning, we investigate whether it can select graph filters for community node ranking at runtime, after graphs and example community members become known and therefore can be used to understand underlying structural features. We consider best-performing filters those with higher node score quality, for instance measured with the area under curve of the receiver operating characteristics (AUC) [389] and the normalized discounted cumulative gain across all graph nodes (NDCG) [390]. We experiment on setting those two measures as objectives and comparing the PageRank (ppr) and HeatKernel (hk), filters with four variations of their diffusion parameter each, against a *selection* of the best on-the-fly, creating a best-performing filter with *tuned* diffusion up to 40 hops away, and a variation of tuning that uses the popular L-BFGS-B optimizer [379] instead of ours. The three last auto-tuning strategies use 10% of known community members for validation.

Experiments are conducted on nine graph communities across three different datasets, where up to 30% members are considered known and node scores should be higher for the rest. Nine experiments per community are conducted. Here, we report the average AUC or NDCG across all experiments when optimizing for the respective measure. Details on the datasets, methodology, and results can be found in the respective publication (Subsection 8.1.4); in general, our proposed *tune* method outperforms *all* the rest in 31/54 scenarios for AUC and 49/54 scenarios for NDCG. When not creating improvements, it lags behind by at most 0.007 for AUC or NDCG compared to the best alternative, and often by much less. Experiments also reveal the value of auto-tuning (given that least 50 non-zero scores are available to ensure robustness), as no standalone filter would be near-best in all scenarios. Finally, Algorithm 1 outperforms the L-BFGS-B optimizer.

Measure	Ad-hoc								Autotune		
	ppr0.5	ppr0.85	ppr0.9	ppr0.99	hk2	hk3	hk5	hk7	select	tune	tuneLBFGB
Avg. AUC	0.897	0.903	0.905	0.898	0.896	0.897	0.901	0.904	0.909	0.912	0.908
Avg. NDCG	0.880	0.883	0.883	0.863	0.878	0.882	0.885	0.883	0.888	0.899	0.877

Table 14. Average measure values for competing graph filters for community recommendation. Best method in bold.

8.1.2.3. The capabilities of pygrank In practice, popular ad-hoc node ranking algorithms and post-processors are implemented in graph management or deep learning packages that support backpropagation. Table 15 compares *pygrank* to alternatives that could run node ranking algorithms in terms of a) provision of *ad-hoc* graph filters, b) supported *backends*, c) ability to define *general-purpose* filters with no additional coding, d) *postprocessing* to improve outcomes, e) online *tuning*,





f) *backpropagation* support, and g) *evaluation* measures of graph-based algorithm quality to enable both supervised and unsupervised assessment depending on business needs.

We consider only base capabilities that pertain to node ranking and are usable by non-experts without additional development. For example, external autoML packages, such as *autogluon* [391], require more coding to use in deep graph learning setups. Overall, *pygrank* introduces new functionality and combines advantages of other packages when it comes to node ranking; our implementations can be plugged in graph analysis and machine learning applications to assemble algorithms from novel or existing graph filters and postprocessors. The package further simplifies writing benchmark experiments for the selection of best algorithms and can apply the previously evaluated auto-tuning, even during GNN training.

package	ad-hoc	backends	general	postpr.	tuning	backpr.	eval.
networkx [392]	✓	numpy					✓
igraph [393]	✓	custom C++					✓
pygsp [394]	✓	numpy	✓				
dgl [395]	✓	mxnet, pytorch, tensorflow				✓	
pyg [396]	✓	pytorch				✓	
tfgnn [397]	✓	tensorflow				✓	
spektral [398]	✓	tensorflow				✓	
pygrank	✓	numpy, pytorch, tensorflow	✓	✓	✓	✓	✓

Table 15. Comparison of *pygrank* with alternatives for node ranking algorithms.

8.1.3. Conclusions

We have presented various frameworks that facilitate graph analysis and learning at the edge with out-of-the box components that can be combined to build complex models. JGNN can run on edge devices with low resources, and *pygrank* is a computationally efficient framework for devices of greater capabilities, like systems-on-chips. *Pygrank* also supports auto-tuning to adapt to the dynamics of graph data on-the-fly.

8.1.4. Relevant publications

- Krasanakis, E., Papadopoulos, S., Kompatsiaris, I., Symeonidis, A. L. *pygrank*: A Python package for graph node ranking. *SoftwareX*, 20, 101227, 2022. <https://zenodo.org/records/7229677>
- Krasanakis, E., Papadopoulos, S., Kompatsiaris, I. *Autogf*: Runtime graph filter tuning for community node ranking. In *International Conference on Complex Networks and Their Applications* (pp. 189-202). Cham: Springer International Publishing, 2022. <https://zenodo.org/records/6836116>
- Krasanakis, E., Papadopoulos, S., Kompatsiaris, I. *JGNN*: Graph Neural Networks on native Java. *SoftwareX*, 23, 101459, 2023. <https://zenodo.org/records/8192070>

8.1.5. Relevant software

- *pygrank*'s documentation is publicly available as a readthedocs page at <https://pygrank.readthedocs.io>, whereas the open source repository for its development is available on github at <https://github.com/MKLab-ITI/pygrank>.





- JGNN documentation and its open source implementation can be found in its repository on github at <https://github.com/MKLab-ITI/JGNN>.

8.1.6. Relevance to AI4Media use cases and media industry applications

Immediate beneficiaries of the JGNN and pygrank libraries are software developers that deploy cross-platform AI solutions on edge devices, especially if these need to adapt to local personalized content. In particular, the libraries serve as prototyping and production-ready tools for graph algorithms and support AI training and fine-tuning on the edge devices. By extension, the media industry can now employ efficient tools for processing and learning from graph-structured data across a variety of hardware and software. New tools running on the frameworks can operate within the confines of edge devices, thus enabling personalized AI that maintains data and preference privacy and exhibits improved autonomy when communication is challenging or insecure.

8.2. GNN architectures trained at the edge

Contributing partner: CERTH

8.2.1. Introduction and methodology

We devise two GNN training strategies for edge devices: a) exploring collective learning in peer-to-peer networks, where each device aims to learn some information about its user based on the coincidence of friendship and communication graphs (Paragraph 8.2.1.1), and b) creating computationally simple yet effective architectures that are suited to learning black-box attributed graph functions for lightweight in-device training on unknown tasks (Paragraph 8.2.1.2).

8.2.1.1. Peer-to-peer GNN for decentralized classification We consider decentralized peer-to-peer networks of infrequently communicating devices. Devices/nodes hold vectors of a shared feature space, such as average word embeddings of local media content, and some training devices in the networks hold manually provided class labels, and we aim to predict the labels for *all* devices so that they match true class labels with high accuracy. To avoid centralization, each device creates predictions about itself, for instance to estimate its user's interests among a list of topics, while only viewing information transmitted by those it communicates with. In a centralized setting, GNN classifiers could take into account the peer-to-peer connectivity graph to improve classification accuracy, given that communication links often reflect real-world relations. Communication is described by time-evolving matrices $A_{com}(t)$, whose non-zero elements indicate exchanges through the corresponding links at time stamps $t = 0, 1, 2, \dots$:

$$A_{com}(t)[u, v] = \{1 \text{ if } u, v \text{ communicate at time stamp } t, \text{ else } 0\}$$

We develop GNNs that let peer-to-peer devices classify themselves by running fragments of GNN architectures. The main challenge is that fragments can account only for local features and can communicate only with the fragments of linked neighbors. Each fragment learns parameters that approximate centralized estimations for the device hosting it. In Algorithm 2, we provide a communication protocol in which we make peer-to-peer nodes learn to classify themselves by exchanging information through channels represented by time-evolving communication matrices. The framework waits for the timeframes when channels become active and executes the broadly popular Send-Receive-Acknowledge communication protocol to exchange information.

We look at decoupled GNNs, which separate the challenges of training base classifiers with leveraging network links to improve predictions. In particular, they consider base classifiers that





Algorithm 2 Send-Receive-Acknowledge protocol

Inputs: devices $u \in V$ with identifiers $u.id$, time-evolving $A_{com}(t) : V \times V \rightarrow \mathbb{R}$

for $t = 0, 1, 2, \dots$ **do**

for all (u, v) such that $A_{com}(t)[u, v] = 1$ **do**

message $\leftarrow u.SEND(v.id)$

reply $\leftarrow v.RECEIVE(u.id, message)$

$u.ACKNOWLEDGE(v.id, reply)$

end for

end for

process features matrices X to output matrices $R_\theta(X)$ with rows holding the predictions of respective feature rows $R_\theta(X)[u] = R_\theta(X[u])$, where $X[u]$ is the feature vector of node u . If base classifiers are trained on the features and labels of node sets V_{train} , we build on the FDiff-scale architecture [355], which stacks two node representation diffusion sub-operations on top of original predictions. We theoretically transform these sub-operations to one unifying expression that should be decentralized:

$$\pi_\infty = (\mathcal{I} - aD^{-d}A_{mask}D^{d-1})^{-1}P_\gamma\pi_0$$

where a, d are helper variables to express both sub-operations with one formula, D is the diagonal matrix of node degrees in the peer-to-peer connectivity graph A , $A_{mask}[u, v] = \{A[u, v] \text{ if } u = v \text{ or } v \notin V_{train}, \text{ else } 0\}$, and $P_\gamma = \frac{1}{1-\beta}\text{diag}(\{1 - \beta \text{ if } u \in V_{train}, \text{ else } 1 - \gamma\}_u)$. The sub-operation performed first sets $d = 0, a = 1, \gamma = 1$, whereas the second sets $d = 0.5, a = \beta, \gamma = \beta$.

Since peers coincide with nodes, we cannot follow works that let them hold fragments of the network to be merged upon communication [373, 399–401]; this would require bandwidth-intensive exchanges of large segments of the network. Instead, we devise a computational scheme that only holds and exchanges local estimations of neighbor representations at each device - a strategy inspired by earlier work on decentralized non-personalized PageRank [402]. In particular, we describe decentralized operations in peer-to-peer networks with a data structure we dub *decentralized graph signals*; these are matrices S with multidimensional vector elements $S[u, v] \in \mathbb{R}^C$ (here, C is the number of classes) that hold in devices u estimates of device v representations. Rows $S[u]$ are stored in devices u and only cross-column operations are affected by infrequent communication.

We now introduce a scheme that updates decentralized graph signals $S(t)$ at times t per:

$$S(t)[u, v] = S(t-1)[u, v] + A_{com}(t)[u, v] \left(\frac{S(t-1)[v][v]}{D[u, u]^d} - S(t-1)[u, v] \right)$$

$$S(t)[u, u] = P_\gamma[u, u]S_0(t)[u] + a \sum_v \frac{A_{mask}[u, v]}{D[u, u]^{1-d}} S(t)[v][v]$$

where $S_0(t)[u] \in \mathbb{R}^C$ are time-evolving representations of nodes u . The first of the above equations describes node representation exchanges between devices based on the communication matrix, whereas the second one performs a local update of personalized PageRank estimation given the last updated neighbor estimation that involves only data stored on devices u . For linear or faster convergence rate of $S_0(t)[u]$, we conduct theoretical analysis in the respective publication to show that the diagonal of above scheme also converges with linear or faster rate to zero average error to $\lim_{t \rightarrow \infty} S(t)[u, u] = \pi_\infty[u]$. Finally, to implement the GNN, we employ the two necessary decentralized PageRank algorithms, where the output of the first one is passed as an evolving personalization into the other.

8.2.1.2. Universal local attractors on graphs GNNs (Paragraph 8.1.1.1) see empirical success in many tasks, but there are emerging concerns on their general applicability. Tests of





expressive power is often taken as a golden standard [403], but do not ensure that losses corresponding to training objectives are straightforward to minimize [404], or that the expressive power is enough to replicate arbitrary objectives. Thus, applied GNNs should also easily find deep loss minima, which refer to empirically achieving small loss values compared to other local minima. These values may be “bad local valleys” [405, 406] that are not close to the global minimum, but can still be considered better training outcomes compared to alternatives. By employing architectures that are not perfect but easy to optimize, we aim to limit resource demands during training at the edge.

Formally, we search for architectures whose parameters lie in a connected compact set such that a transformation of parameters achieves sufficiently small derivatives. We call this property *local attraction*. If there exist multiple applicable transformation mechanisms, one can select those that create plateaus that are more difficult to escape from when they encompass deep minima but are easier to escape from when they contain only shallow local minima. Below we summarize our theoretical results that derive a GNN that is a local attractor in most situations. *Details of principled mathematical analysis can be found in the respective publication* (Subsection 8.2.4).

We start by satisfying Universal Local Attraction (ULA) in the one-dimensional (1D) setting. To this end, we look at positive definite graph filters $F(\hat{A})$, i.e., that create positive diffusion eigenvalues, and show that optimization trajectories on the filter’s outputs are tightly followed if we apply appropriate prior graph signal modifications. Small inexactness is absorbed by the filter’s stability, asymptotically leading to zero gradients under some mild conditions. We approximate the end-result of this procedure with a GNN called ULA1D. This consists of a Multi-Layer Perceptron MLP_θ with one-dimensional outputs and sufficiently many layers for a universal approximation property (e.g., the one presented in [407]) to hold and therefore find the ideal convergent point of priors. Near-ideal priors then pass through the filter to obtain predictions that approximate zero gradients at any desired tightness level per:

$$R = F(\hat{A})\text{MLP}_\theta(F(\hat{A})X|X)$$

where X are the one-dimensional priors and $|$ indicates horizontal concatenation.

The same architecture approximates functions defined on finite sets of graphs with one-dimensional node features and bounded number of nodes by organizing those graphs in one larger one, concatenating their features, and minimizing an equivalent loss on the combination. This result also extends to infinite sets of graphs with a bounded number of nodes by discretizing bounded node feature values; as long as MLP_θ and loss gradients are Lipschitz with Lipschitz derivatives almost everywhere (e.g., this holds true for architectures with *relu* activations) we can create an arbitrarily good but finite discretization of the node feature domain on which local attraction holds in a connected neighborhood of local minima.

We next extend ULA1D to multiple node feature dimensions by treating each of them as a separate graph signal defined on copies of the graph stacked together in a supergraph. This lets us optimize all dimensions by creating a supergraph of them. This also creates a signal that stacks all dimensions on top of each other. Computational invariance with respect to node features (permuting a node permutes corresponding features), which is a desired property when approximating attributed graph functions, is maintained by concatenating on new nodes’ features an embedding of $\text{ceil}(\log_2 K)$ elements indicating which feature dimension they correspond to, where K is the number of starting node feature dimensions. To facilitate a different number of input and output feature dimensions, we finally two additional layers as first and last steps, where the first of those should have enough degrees of freedom so as to be invertible but small enough output dimension K to enable computational tractability. The new loss needs to still have Lipschitz gradients, which in turn means that input and output transformations should *not* be able to approximate arbitrary functions, i.e., they cannot be MLPs. For ease of implementation, we make





the transformations trainable, but they may also be extrapolated from informed ad hoc strategies, like node embeddings for matrix factorization.

8.2.2. Results

8.2.2.1. Evaluating peer-to-peer GNNs To show the ability of our proposed methodology in replicating centralized settings, we experiment on three datasets that are often used to assess the quality of GNNs [408]; these are retrieved from the public programming interface of the DGL library [395] and comprise thousands of nodes and node features features, and up to 88k edges and 7 class labels. They also come along training-validation-test sets commonly used in GNN literature experiments. We simulate peer-to-peer networks with the nodes, node features, and links of the datasets and set fixed probabilities for communication through links at each time step, uniformly sampled from the range $[0, 0.1]$. In total, we experiment on a two-layer perceptron (MLP) and logistic regression (LR) classifiers that are either pre-trained in a central service and shared with devices or trained with the decentralized gossip averaging protocol, where communicating nodes average trainable parameters between themselves [345, 346]. We also consider label diffusion using personalized PageRank that discounts node features. Our goal is to replicate a fully centralized GNN with our scheme, which we dub p2pGNN, outperforming the base classifiers (MLP and LR).

Personalized PageRank’s diffusion parameter is set to $\beta = 1$, and we set $s = 1$ to create zero errors for training nodes. All trained layers are 64-dimensional, have 50% training dropout, and are trained with an Adam optimizer with learning rate 0.01 and 0.0005 L2 weight regularization (these options are standard defaults for GNNs). Finally, in Table 16 we measure classification accuracy of test labels after 1,000 time steps (all algorithms converge much sooner) and report its average across five experiment repetitions. Similar results are obtained for communication rates sampled from different range intervals. Experiments run on a machine running Python 3.6 with 64GB RAM and 32x 1.80GHz CPUs. Overall, p2pGNN successfully improves the accuracy of base classifiers by 7%-47% relative increase, and lets them resemble centralized diffusion. Improvements are mixed for gossip training, because it already implicitly incorporates diffusion that makes reported accuracy less reliable in learning from features; adoption of our framework should be preferred with pre-trained base classifiers. In terms of communication overhead, after exchanging initial data like the shared classifier architecture and pre-trained parameters, device messages are increased by at most 350 bytes each to apply our strategy, and up to 2MB for gossip training.

	Base			p2pGNN			Centralized FDiff-scale		
	Citeseer [388]	Cora [409]	Pubmed [410]	Citeseer	Cora	Pubmed	Citeseer	Cora	Pubmed
Pre-trained									
MLP	52.3%	54.9%	70.9%	67.8%	81.5%	76.0%	69.0%	84.0%	81.2%
LR	59.4%	58.7%	72.2%	70.5%	82.0%	77.3%	70.3%	85.7%	81.5%
Gossip-trained									
MLP	63.1%	66.3%	74.9%	61.3%	80.8%	78.0%	69.0%	84.0%	81.2%
LR	61.8%	79.9%	78.7%	61.4%	80.8%	78.7%	70.3%	85.7%	81.5%
Labels	15.9%	11.6%	22.0%	61.1%	80.8%	71.5%	61.5%	78.9%	78.6%

Table 16. Comparing the accuracy of different types and training schemes of base algorithms and their combination with the diffusion of p2pGNN. Accuracy is computed after 1,000 time steps and averaged across 5 peer-to-peer simulation runs.





8.2.2.2. Evaluating universal local attractors To evaluate the ULA architecture in black-box settings, we created a corpus of 9 synthetic tasks on 500 graphs of up to 100 nodes and 10,000 edges. To this corpus we added the three node classification tasks of Subsection 8.2.2.1. The synthetic tasks range from learning score diffusion mechanisms to learning complicated algorithms like the length of longest shortest paths. We use a 50%-25%-25% train-validation-test split. ULA can be written as a message-passing GNN, and thus is compared to alternatives of similar or greater expressive power (WL-1) while using the same common GNN defaults of Subsection 8.2.2.1 and a two-layer MLP_θ . To not disadvantage local attraction as it tries to overcome small but non-negligible derivatives, we apply a late stopping criterion in which training halts only if both training and validation losses do not increase for 300 epochs. Training convergence is demonstrated for an example setting in Figure 26.

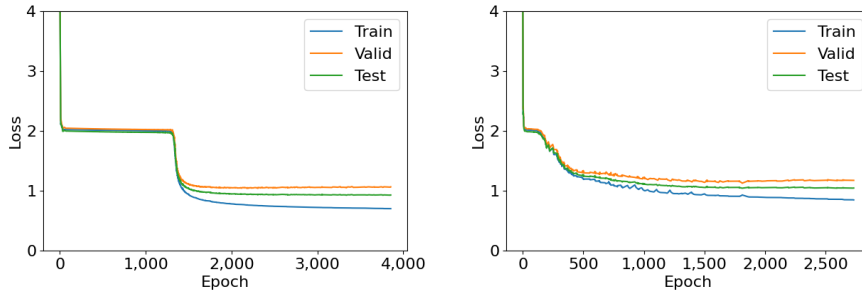


Figure 26. Example convergence of train, validation, and test set losses of ULA (left) and GCNNRI (right).

A summary of comparisons across all tasks is provided in Table 17, where the average Nemenyi ranks compare methods across different types of evaluation measures like mean absolute error and accuracy (rank 1 indicates methods outperforming others on all experiments, rank 2 means indicates that a method is on average second-best, etc). Per the Bonferroni-Dunn test [411, 412], statistically significant differences at the 0.05 p-value level occur when ranks exceed a critical difference of 1.1 between ULA and alternatives (the test does not correct for differences between alternatives).

	MLP GCN [354]	APPNP [375]	GAT [413]	GCNII [356]	GCNNRI [414]	ULA
Minimization	4.2	3.0	5.6	5.0	6.0	2.6
Generalization	5.2	3.5	4.7	4.8	4.7	3.5

Table 17. Average Nemenyi ranks of GNN architectures across 12 predictive tasks.

We compare algorithms in terms of directly minimizing losses (by making all nodes belong to both the training, validation, and test sets) and of how well found minima generalize in terms of predictive performance. In this multi-task corpus, GCN and GCNNRI (where the latter has WL-3 expressive power instead of at most WL-1 of the other architectures) outperform alternatives devised for the analysis of single graphs that exhibit strong homophily, i.e., tendency to link similarly-labeled nodes. ULA is on average the best in our particular experiments with statistical significance and should therefore be considered for practical adoption, even when greater expressive power is available. Theoretical analysis also reveals that it performs lightweight computations by consuming time and memory $O(ENK \log K + VK(\log^2 K + \text{col}(X) + \text{col}(Y)))$ where N is the number of MLP_θ layers, E the number of edges, V the number of nodes, and $\text{col}(X)$ and $\text{col}(Y)$ the input and output dimensions respectively.





8.2.3. Conclusions

We have introduced p2pGNN as a methodology with which to perform privacy-aware machine learning within peer-to-peer networks by leveraging communication links. We have also introduced the ULA architecture as a lightweight GNN that leverages the local attraction property to perform similarly or better than several state-of-the-art alternatives in learning black-box attributed graph functions. Limitations of these methodologies and concerns that should be accounted for before their adoption are described in respective publications (see below).

8.2.4. Relevant publications

- Krasanakis, E., Papadopoulos, S., Kompatsiaris, I. p2pgnn: A decentralized graph neural network for node classification in peer-to-peer networks. IEEE Access, 10, 34755-34765, 2022. <https://zenodo.org/records/6375643>
- Krasanakis, E., Papadopoulos, S., Kompatsiaris, I. Universal Local Attractors on Graphs. Applied Sciences, 14(11), 4533, 2024. <https://zenodo.org/records/11382581>

8.2.5. Relevant software

- The implementation and experiments for p2pGNN can be found in a github repository at <https://github.com/MKLab-ITI/decentralized-gnn>.
- The experiments for universal attractors on graphs can be found in a github repository at <https://github.com/MKLab-ITI/ugnn>.

8.2.6. Relevance to AI4media use cases and media industry applications

Both p2pGNN and the ULA architecture can be used to create more accurate GNNs at the edge by respectively supporting collective learning over edge device data, for example to better detect media preferences of smartphone users to make recommendations, and better minimizing machine learning losses with a relatively simple architecture to improve graph-based AI.

8.3. Genetic Algorithms For Federated Learning

Contributing partner: CERTH

8.3.1. Introduction

Federated learning has emerged as a promising paradigm for training machine learning models on the edge [415]. In federated learning, copies of an AI model are trained locally at edge devices or servers with local data samples, and then aggregated at a central node. The final model is computed through successive rounds of local training and aggregation, sharing only model parameters instead of data samples [344]. Therefore, this approach is particularly valuable when the end users or clients have their own distributions of data, which cannot be centralized due to privacy concerns, regulatory restrictions, or practical limitations [416]. Potential drawbacks are a high communication overhead, impaired convergence, and poor adaptation to heterogeneous node conditions.

Genetic algorithms have the potential to address these challenges while preserving privacy [417]. In particular, as federated learning often involves distributed clients with varying computational resources, network conditions, and data distributions, genetic algorithms offer potential advantages





in terms of reduced communication overhead, improved convergence rates, and adaptability to heterogeneous data environments [418].

Our work investigates the application of genetic algorithms in federated learning scenarios, focusing on situations where data privacy and network optimization are paramount. Our experiments compare the performance of traditional Federated Averaging (FedAvg) with three nature-inspired optimization algorithms: Federated Particle Swarm Optimization (FedPSO), Federated Ant Colony Optimization (FedACO), and Distributed Differential Evolution (DDE).

8.3.2. Methodology

We implement and compare four algorithms in a federated learning context; three nature-inspired and the standard federated averaging. These algorithms are designed to balance the trade-offs between model performance, data privacy, and communication efficiency, as described below.

Federated Averaging: FedAvg, proposed by McMahan et al. [419], serves as the baseline algorithm. In FedAvg, clients train local models on their respective datasets for a number of local epochs. The central server then aggregates these models by computing a weighted average of their parameters, where the weights are proportional to the size of each client's dataset. The update rule for the global model \mathbf{w} at round $t + 1$ is given by:

$$\mathbf{w}^{(t+1)} = \sum_k \frac{n_k}{n} \mathbf{w}_k^{(t+1)}$$

where n_k is the number of samples at client k , n the total number of samples, and $\mathbf{w}_k^{(t+1)}$ the updated model of client k . FedAvg preserves privacy by keeping raw data on local devices and only sharing model updates. Communication takes place only during model aggregation but sharing the full model parameters may be costly, depending on the model size.

Distributed Differential Evolution: DDE adapts the Differential Evolution algorithm [420] to a federated setting. In DDE, each client maintains a population of candidate solutions (model parameters), evolving them locally through mutation, crossover, and selection steps. This approach preserves privacy by keeping raw data on local devices and only sharing the best solutions periodically. Sharing the best solutions to the central server reduces the communication cost compared with sharing the full model parameters.

Federated Ant Colony Optimization: FedACO adapts the Ant Colony Optimization algorithm [421] to federated learning. In this approach, each client performs local ACO, where artificial ants construct solutions (model parameters) based on pheromone trails and heuristic information. FedACO preserves privacy by keeping local data confidential and only exchanging aggregated pheromone information. This strategy reduces network usage by sharing summarized pheromone information instead of raw data or full model parameters.

Federated Particle Swarm Optimization: FedPSO adapts the Particle Swarm Optimization algorithm [422] to federated learning. This method leverages the collective intelligence of distributed particles to optimize model parameters while maintaining data privacy and reducing communication overhead. In FedPSO, each client maintains a swarm of particles, where each particle represents a set of model parameters, and the algorithm operates through an iterative process of client and server updates (see box below). FedPSO preserves privacy by keeping raw data and individual particle positions on local devices, sharing only aggregated knowledge rather than raw data. Network usage is significantly reduced compared with traditional federated learning approaches, as clients send their local best solutions to the server.





Initialization:

- Initialize weights (**w**), personal best (**pbest**), global best (**gbest**), and global best identifier (**gid**).

Client Update:

- Each client performs local learning using its data and updates particle positions.
- Clients calculate their best performance score (**pbest**) and send it to the server.
- Particle velocities and positions are updated using the following equations:

$$\mathbf{V}_t = \alpha \cdot \mathbf{V}_{t-1} + c1 \cdot \text{rand1} \cdot (\mathbf{pbest} - \mathbf{V}_{t-1}) + c2 \cdot \text{rand2} \cdot (\mathbf{gbest} - \mathbf{V}_{t-1})$$

$$\mathbf{w}_t = \mathbf{w}_{t-1} + \mathbf{V}_t$$

Where:

- α is the inertia weight
- $c1$ and $c2$ are acceleration constants
- rand1 and rand2 are random values between 0 and 1
- **pbest** is the best solution found by an individual particle
- **gbest** is the best solution found by the entire swarm

Server Update:

- The server collects **pbest** values from all clients.
- It identifies the global best score (**gbest**) and requests the model weights from the client with the highest **pbest**.
- The server updates the global model using the received weights.

Iteration:

- The process repeats for a predetermined number of rounds or until convergence.

The FedPSO algorithm offers several especially attractive advantages in the federated learning context, which are described below:

- **Adaptive exploration:** Particles can efficiently explore the solution space, adapting to local data distributions.
- **Scalability:** The algorithm can handle a large number of clients with diverse data distributions.
- **Robustness:** PSO's stochastic nature helps in avoiding local optima, which is particularly useful in non-IID data scenarios common in federated learning.





- **Reduced communication:** By only sharing best scores and selectively updating the global model, FedPSO significantly reduces the amount of data transferred between clients and the server.

8.3.3. Experiments

We conducted experiments with a ResNet50 model [26] as the baseline architecture. The CIFAR-10 dataset [423], consisting of 60,000 32x32 color images in 10 classes, was used for training and evaluation. The federated learning setup involved 5 users (clients) and training was performed for 50 epochs. Two scenarios were explored:

1. **Training from scratch:** The ResNet50 model was initialized with random weights.
2. **Transfer learning:** The ResNet50 model was initialized with pre-trained weights from ImageNet [96].

Non-IID Data Distribution: In real-world federated learning scenarios, data is often not independently and identically distributed (non-IID) across clients [424]. To simulate this, we implemented a non-IID data distribution for the CIFAR-10 dataset using a Dirichlet distribution [425]. This approach allows for controlled heterogeneity in the class distribution across clients, mimicking real-world scenarios where different clients may have biased subsets of the data.

The degree of non-IID is controlled by a concentration parameter α . Lower values of α result in more skewed distributions, while higher values lead to more uniform distributions. This setup enables us to evaluate the robustness of the algorithms under varying degrees of data heterogeneity, which is crucial for assessing their real-world applicability.

Results: The experiments conducted on the CIFAR-10 dataset using a ResNet50 model architecture yielded interesting insights into the performance of different federated learning algorithms. Each client had a non-IID subset of the original dataset, simulated using a Dirichlet distribution. Table 18 summarizes the accuracy results for each algorithm, both with and without transfer learning.

Table 18. Genetic Algorithms' models Accuracy Results

Model	Accuracy (with Transfer Learning)	Accuracy (without Transfer Learning)
FedAVG	77.06%	63.37%
FedPSO	74.74%	55.59%
FedACO	70.38%	50.11%
DDE	72.08%	51.08%

Several key observations can be made from these results:

1. **Transfer Learning Impact:** All algorithms showed significant improvement when using transfer learning with pre-trained weights from ImageNet. This highlights the effectiveness of transfer learning in federated settings, particularly for image classification tasks.
2. **Algorithm Performance:**
 - (a) **FedAVG:** The traditional Federated Averaging algorithm outperformed the nature-inspired algorithms in both scenarios, achieving the highest accuracy of 77.06% with transfer learning and 63.37% without transfer learning.





- (b) **FedPSO:** Among the nature-inspired algorithms, Federated Particle Swarm Optimization showed the best performance, achieving 74.74% accuracy with transfer learning and 55.59% without transfer learning.
 - (c) **DDE:** Distributed Differential Evolution performed slightly better than FedACO, with accuracies of 72.08% and 51.08% for transfer and non-transfer learning scenarios, respectively.
 - (d) **FedACO:** Federated Ant Colony Optimization showed the lowest accuracy among the tested algorithms, but still achieved reasonable performance with 70.38% accuracy with transfer learning and 50.11% without transfer learning.
3. **Performance Gap:** While the nature-inspired algorithms did not surpass FedAVG in terms of accuracy, they still achieved respectable performance. The gap between FedAVG and the best-performing nature-inspired algorithm (FedPSO) was 2.32% with transfer learning and 7.78 percentage points without transfer learning.
 4. **Transfer Learning Efficiency:** The improvement in accuracy due to transfer learning was substantial across all algorithms, with increases ranging from 13.69 percentage points to 21.15 percentage points. This underscores the value of leveraging pre-trained weights in federated learning scenarios.

These results provide valuable insights into the performance characteristics of different federated learning algorithms. While FedAVG demonstrated superior accuracy, the nature-inspired algorithms showed promising results, particularly when combined with transfer learning. It's important to note that accuracy is just one aspect of algorithm performance in federated learning. Other factors such as communication efficiency, privacy preservation, and adaptability to non-IID data distributions should also be considered when evaluating the overall effectiveness of these algorithms.

8.3.4. Conclusions

This study demonstrates the potential of genetic algorithms in federated learning scenarios. The nature-inspired optimization techniques (FedPSO, FedACO, and DDE) showed promising results in comparison to the traditional FedAvg algorithm, particularly in terms of adaptability to non-IID data distributions and potential for reduced communication overhead.

The transfer learning scenario, utilizing pre-trained weights from ImageNet, generally resulted in improved performance across all algorithms, highlighting the benefits of leveraging existing knowledge in federated learning settings.

While FedAVG outperformed the nature-inspired algorithms in terms of accuracy, the latter showed potential advantages in terms of privacy preservation and communication efficiency. FedPSO, in particular, demonstrated the best performance among the nature-inspired algorithms, suggesting its potential for further optimization in federated learning contexts.

The implementation of non-IID data distribution using the Dirichlet distribution provided a more realistic simulation of real-world federated learning scenarios. This approach allowed for a more comprehensive evaluation of the algorithms' robustness to data heterogeneity, which is crucial for their practical application.

While these genetic algorithms show promise, further research is needed to fully understand their behavior in diverse federated learning environments, particularly with larger numbers of clients and more complex datasets. Future work could explore adaptive hybridization of these algorithms [426] and investigate their performance under varying degrees of data heterogeneity [427].

In conclusion, this study provides valuable insights into the application of genetic algorithms in federated learning, offering potential alternatives to traditional methods for scenarios where data





privacy, communication efficiency, and adaptability to non-IID data are critical considerations. The results underscore the importance of considering multiple performance metrics beyond just accuracy when evaluating federated learning algorithms.

8.3.5. Relevant Publications

- No relevant publications published yet.

8.3.6. Relevance to AI4Media use cases and media industry applications

Genetic algorithms for federated learning are particularly effective in scenarios where data privacy and network communication efficiency are paramount. These algorithms can be integrated into use cases, such as a decentralized content personalisation system for a media conglomerate, where multiple TV networks and streaming services need to train AI models to recommend shows and movies to users without sharing their individual viewing data, allowing for more accurate and targeted recommendations while maintaining user privacy. Each participating user possesses a non-IID distribution of data that cannot be shared, and network bandwidth efficiency is crucial. This approach ensures robust model training while preserving data privacy and optimizing communication overhead.

8.4. InDistill: Information flow-preserving knowledge distillation for model compression

Contributing partner: CERTH

8.4.1. Introduction

The constant effort to improve the performance of DNNs has resulted in deeper architectures that require massive compute power and memory to train and deploy. Also, several applications necessitate the deployment of DNNs at hardware with limited resources, which has sparked research interest in model compression. During the past years, many pertinent approaches have been proposed such as parameter pruning, quantization, low-rank factorization, and knowledge distillation (KD) [428, 429].

The objective of KD, one of the most effective ways for model compression [172, 429], is to transfer knowledge from a larger “teacher” network to a smaller and faster “student” network, in order to improve the student’s performance. Early KD approaches transferred only the teacher’s last layer [160]. This allowed the student to perceive the inputs like the teacher but forewent how the information flows through the teacher’s layers, thus reducing the student’s generalization capabilities [430]. As a remedy, recent KD methods [431, 432] transfer intermediate layers but require encoding stages to overcome the width discrepancy between the teacher’s and the student’s layers, impairing the preservation of information flow.

In this context, CERTH has developed InDistill, a method that can be employed in combination with any KD approach. We show that properly pruning the teacher’s intermediate layers can enable direct knowledge transfer and effectively distill the teacher’s information flow paths to the student. Furthermore, inspired by curriculum learning [433], we develop a simple, yet effective way to distill multiple layers from a teacher to a student model, while taking into consideration the critical learning periods [430]. Specifically, we show that distilling each intermediate layer separately and in ascending transferring difficulty order (i.e., from shallow to deep layers) can enhance the KD effectiveness and consequently the student’s performance. Indistill is evaluated on a wide





range of classification and retrieval tasks from widely adopted benchmarks (i.e., CIFAR-10 [42], CIFAR-100 [42], and ImageNet [43]) and is found to outperform combined various state-of-the-art KD approaches in most of the conducted experiments.

8.4.2. Methodology

8.4.2.1. Problem formulation The problem of knowledge distillation for image processing models is formulated as follows. Let $\mathbf{X} \in \mathbb{R}^{3 \times h \times w}$ denote an input image with h and w height and width, respectively, $d(\cdot)$ the teacher model, and $l=1, \dots, L_d$ the layer's index. Then, $\mathbf{T}^{(l)} = d(\mathbf{X}, l) \in \mathbb{R}^{n_{d,l} \times h_{d,l} \times w_{d,l}}$ denotes the teacher's l layer output, where $n_{d,l}$, $h_{d,l}$, and $w_{d,l}$ is its number of channels and spatial dimensions. Accordingly, consider a student model $g(\cdot)$ with θ learnable parameters, L_g layers and $\mathbf{S}^{(l)} = g(\mathbf{X}, l) \in \mathbb{R}^{n_{g,l} \times h_{g,l} \times w_{g,l}}$ the l layer's output. Let $\mathbf{q}_t, \mathbf{q}_s \in \mathbb{R}^C$ be the class probability distributions of the teacher and student model respectively, with C the number of output classes. The goal of single-layer KD is to either match the teacher's and student's class probability distributions ($\mathbf{q}_t \simeq \mathbf{q}_s$) or force the penultimate layer representations to share maximum amount of information, namely $\max_{\theta} \mathcal{I}(\mathbf{T}^{(L_d)}; \mathbf{S}^{(L_g)})$. In addition to that, the intermediate layer KD provides extra supervision to the main target by matching several teacher's and student's intermediate layer pairs.

In cases of teacher-student pairs with large capacity gap, different number of layers $L_d \neq L_g$ or structurally different architectures (e.g., a teacher with and a student without residual connections), we follow other works [430, 434] that make use of an auxiliary model built from the teacher model using typical knowledge distillation. $f(\cdot)$ denotes the auxiliary model, L_f the number of layers (here $L_f=L_g$), and $\mathbf{A}^{(l)} = f(\mathbf{X}, l) \in \mathbb{R}^{n_{f,l} \times h_{f,l} \times w_{f,l}}$ its output feature maps. Note that $h_{f,l}=h_{g,l}$ and $w_{f,l}=w_{g,l}$ as the networks share kernel sizes. Finally, the auxiliary's class probability distributions are denoted by \mathbf{q}_a .

8.4.2.2. Channel pruning Channel pruning is used here to improve the effectiveness of KD, by forcing architectural alignment. The typical criterion for evaluating the importance of a filter is the l_1 -norm or l_2 -norm [435–437]. Here, we opt for the approach of [437] that applies structured channel pruning based on l_1 -norm. Specifically, let $\mathbf{F}_i \in \mathbb{R}^{n_i \times h_i \times w_i}$ denote the input features and $\mathbf{F}_o \in \mathbb{R}^{n_o \times h_o \times w_o}$ denote the output features of a layer, respectively. The layer's kernel can be denoted by $\mathbf{K} \in \mathbb{R}^{n_i \times n_o \times k \times k}$, where k is the kernel size. Then, we follow the pruning procedure described in Algorithm 3.

Algorithm 3 Channel pruning procedure.

Inputs: filters \mathbf{K} and the number of filters to prune p
Output: the pruned filters $\mathbf{K}' \in \mathbb{R}^{n_i \times (n_o-p) \times k \times k}$
for $i \in \mathbf{K}$ **do**
 $\mathbf{s}_i \leftarrow \sum_{j=1}^{n_i} \sum_{l=1}^k \sum_{m=1}^k |\mathbf{K}_{j,i,l,m}|$
end for
 $\mathbf{s}, \mathbf{x} \leftarrow \text{sort}(\mathbf{s})$ { \mathbf{x} denotes the sorted indices array}
 $\mathbf{x} \leftarrow \mathbf{x} \setminus \mathbf{x}[1:p]$ {remove the first p indices}
 $\mathbf{K}' \leftarrow \mathbf{K}[\mathbf{x}]$

8.4.2.3. Information flow-preserving KD When an auxiliary teacher is necessary (see Section 8.4.2.1), we design it to have the same number of layers and the same number of output channels as the student, after applying pruning with pruning rate q to all intermediate layers. As a result, the auxiliary model's feature maps sizes $n_{f,l}$ depend on the student model's feature maps



sizes $n_{g,l}$ and the pruning rate $q \in [0, 1)$, namely $n_{f,l} = \frac{n_{g,l}}{1-q}$. Having designed the auxiliary model, typical KD [160] is applied to transfer the knowledge from the teacher model to the auxiliary model. The following parts of the methodology are identically applied to cases with and without an auxiliary teacher, thus we keep the teacher's notation for simplicity. Channel pruning is applied to all teacher's intermediate layers to force architectural alignment. After applying Algorithm 3 with $p = q \cdot n_{d,l}$, the teacher's pruned feature maps $\mathbf{P}^{(l)} \in \mathbb{R}^{n_{g,l} \times h_{g,l} \times w_{g,l}}$ are of equal size to the student's feature maps, which allows for direct knowledge transfer involving no encoding. The loss between models' feature maps, $\mathbf{P}^{(l)}$ and $\mathbf{S}^{(l)}$ is defined as:

$$\mathcal{L}_{MSE}^{(l)} = \|\mathbf{P}^{(l)} - \mathbf{S}^{(l)}\|_2^2, \quad (6)$$

where $\|\cdot\|_2$ denotes the l_2 -norm. Given that InDistill is only applied to intermediate layers, any of the existing KD methods can be employed for the last layer. Considering the original KD method [160] that transfers the class probability distributions using the Kullback-Leibler (KL) divergence loss, let \mathbf{u} and \mathbf{v} denote the teacher's and student's logits, respectively. Then, the teacher's and student's probability distributions are defined by $\mathbf{q}_{t,i} = \frac{e^{\mathbf{u}_i/T}}{\sum_j e^{\mathbf{u}_j/T}}$ and $\mathbf{q}_{s,i} = \frac{e^{\mathbf{v}_i/T}}{\sum_j e^{\mathbf{v}_j/T}}$, respectively, where T is the temperature term. Finally, the KL loss is defined as follows:

$$\mathcal{L}_{KL} = \sum_i \mathbf{q}_{t,i} (\log \mathbf{q}_{t,i} - \log \mathbf{q}_{s,i}) \cdot T^2, \quad (7)$$

8.4.2.4. Curriculum learning Curriculum learning suggests dividing a hard task into sub-tasks w.r.t. their difficulty and performing training sequentially in ascending difficulty order. When applying intermediate layers KD, one could hypothesize that transferring shallow layers is easier than transferring deep layers, for two reasons. First, shallow layers hold in general low-level information (e.g., edges and corners) while deep layers hold task-specific high-level information. Second, deep layers' transfer also requires transferring the knowledge obtained by all previous layers as well as the information flow paths until then. Given the above, we propose a curriculum learning scheme that considers the transferring of each layer as a separate sub-task. Particularly, let L_g be the number of layers (i.e., sub-tasks) that determine the training schedule. If the total number of training epochs is E , then the number of epochs corresponding to each sub-task (i.e., layer) can be calculated as:

$$e_i = \begin{cases} a + ib & i \neq L_g \\ E - \sum_{i=1}^{L_g-1} e_i & i = L_g \end{cases}, i \in \{1, 2, \dots, L_g\}, \quad (8)$$

where parameter a indicates a threshold for each layer's training epochs and b is a parameter that increments the number of epochs w.r.t. the sub-task's difficulty. Also, the set of epochs that corresponds to each sub-task i is $\mathcal{S}_i = \{r_i + 1, r_i + 2, \dots, r_i + e_i\}$ where:

$$r_i = \begin{cases} 0 & i = 1 \\ \sum_{k=1}^{i-1} e_k & i > 1 \end{cases}, i \in \{1, 2, \dots, L_g\}. \quad (9)$$

By adopting the proposed curriculum learning scheme, the first $\sum_{i=1}^{L_g-1} e_i$ epochs are dedicated to the intermediate layers KD. In particular, the final loss is calculated as:

$$\mathcal{L} = \begin{cases} \mathcal{L}_{MSE}^{(i)} & i \neq L_g \\ \mathcal{L}_{task} & i = L_g \end{cases}, i \in \{0, 1, 2, \dots, L_g\}, \quad (10)$$

where \mathcal{L}_{task} denotes the distillation loss. This way, the student model can effectively form the critical connections that significantly facilitate the KD task.





8.4.3. Experimental setup

The developed method has been evaluated on three well-known datasets: CIFAR-10 [42], CIFAR-100 [42], and ImageNet [43]. Regarding the model architectures, we employ the typical ResNet pairs that are widely used for evaluating KD methodologies [438]. In addition to them, we employ the ResNet50-ResNet18 and ResNet32x4-ResNet32 pairs (i.e., pairs of different width) in order to assess the contribution of pruning to information flow preservation. Using an auxiliary teacher is redundant for these pairs, as the teacher and student models belong to the same model family (ResNets) and the capacity gap between them is not significant. Furthermore, note that channel pruning is applied at the last layer of each block instead of each layer, thus enabling to apply InDistill to ResNets of different depths.

For the less complex dataset, namely CIFAR-10, we consider ResNet18 as the teacher model (consisting of around 11 million trainable parameters) and a tiny CNN consisting of 3 convolutional layers as the student model (CNN-S). Due to the structural differences as well as the large capacity gap between teacher and student, we additionally consider an auxiliary teacher model (CNN-A). For CNN-A and CNN-S, batch normalization is applied after each convolutional layer. More details regarding the implementation details can be found in the preprint version of the paper presenting InDistill [439].

Table 19. Top-1 accuracy on CIFAR-100.

Teacher	WRN-40-2	WRN-40-2	ResNet56	ResNet110	ResNet110	ResNet32x4
Student	WRN-40-1	WRN-16-2	ResNet20	ResNet32	ResNet20	ResNet8x4
Teacher	75.61	75.61	72.34	74.31	74.31	79.42
Student	71.98	73.26	69.06	71.14	69.06	72.50
KD [160]	73.54	74.92	70.66	73.08	70.67	73.33
KD+InDistill	75.09 (+1.55)	76.17 (+1.25)	72.16 (+1.50)	74.78 (+1.70)	72.29 (+1.62)	75.73 (+2.40)
PKT [440]	73.45	74.54	70.34	72.61	70.25	73.64
PKT+InDistill	73.28 (-0.17)	74.59 (+0.05)	70.36 (+0.02)	72.92 (+0.31)	70.38 (+0.13)	73.92 (+0.28)
CRD [438]	74.14	75.48	71.16	73.48	71.46	75.51
CRD+InDistill	74.22 (+0.08)	75.54 (+0.06)	71.79 (+0.63)	73.95 (+0.47)	71.85 (+0.39)	75.73 (+0.22)
DKD [441]	74.81	76.24	71.97	74.11	71.06	76.32
DKD+InDistill	75.39 (+0.58)	76.12 (-0.12)	71.95 (-0.02)	74.59 (+0.48)	71.98 (+0.92)	76.46 (+0.14)
MLKD [442]	75.35	76.63	72.19	74.11	71.89	77.08
MLKD+InDistill	75.71 (+0.36)	76.92 (+0.29)	72.64 (+0.45)	74.68 (+0.57)	72.06 (+0.17)	76.99 (-0.09)

8.4.4. Results

First, we evaluate InDistill on the CIFAR-100 dataset, employing the standard teacher-student pairs used for benchmarking KD methodologies. The results are summarized in Table 19. For each teacher-student pair, we report the performance of the baselines compared to the performance of the baselines combined with the proposed InDistill method. InDistill consistently improves the performance across most methods, demonstrating significant gains. Notably, KD+InDistill shows a marked improvement in all teacher-student pairs, with the highest gain of +2.40% when WRN-32x4 is the teacher and WRN-8x4 is the student. Similarly, for CRD, the gain achieved by adding InDistill is notable, especially with a +0.63% improvement for ResNet56 to ResNet20.

Table 20 presents the results on the ImageNet dataset. The performance metrics include top-1





and top-5 accuracy for different methods, with and without the InDistill approach. The addition of InDistill yields improvements across all methods. For instance, KD+InDistill shows an increase of +0.31% in top-1 accuracy and +0.37% in top-5 accuracy. Similarly, for the top-performing method MLKD, MLKD+InDistill achieves gains of +0.13% in top-1 accuracy and +0.22% in top-5 accuracy, demonstrating the robustness and efficacy of the InDistill approach.

Table 20. Classification evaluation on ImageNet. ResNet34 and ResNet18 are employed as teacher and student networks.

method	top-1 acc	top-5 acc
KD [160]	71.03	90.05
KD+InDistill	71.34 (+0.31)	90.42 (+0.37)
PKT [440]	70.41	89.48
PKT+InDistill	71.13 (+0.72)	90.24 (+0.76)
CRD [438]	71.17	90.13
CRD+InDistill	71.24 (+0.07)	90.37 (+0.24)
DKD [441]	71.70	90.41
DKD+InDistill	71.87 (+0.17)	90.65 (+0.24)
MLKD [442]	71.90	90.55
MLKD+InDistill	72.03 (+0.13)	90.77 (+0.22)

Table 21. Metric learning and classification evaluation on ImageNet and CIFAR-100. ResNet50 as a teacher and ResNet18 as a student are selected for the ImageNet dataset, while ResNet32x4 as a teacher and ResNet32 as a student are selected for the CIFAR-100 dataset.

	method	mAP	P@100	accuracy
ImageNet	KD [160]	30.91	31.98	71.33
	KD+InDistill	31.52 (+0.61)	32.53 (+0.55)	71.54 (+0.21)
	PKT [440]	24.67	26.41	70.66
	PKT+InDistill	25.75 (+1.08)	27.46 (+1.05)	70.98 (+0.32)
	CRD [438]	25.99	27.46	70.49
	CRD+InDistill	31.10 (+5.11)	32.19 (+4.73)	71.36 (+0.87)
CIFAR-100	KD [160]	56.59	65.08	72.11
	KD+InDistill	59.79 (+3.20)	67.45 (+2.37)	72.96 (+0.85)
	PKT [440]	53.20	64.01	72.68
	PKT+InDistill	54.54 (+1.34)	64.78 (+0.77)	72.77 (+0.09)
	CRD [438]	51.34	63.19	73.70
	CRD+InDistill	58.40 (+7.06)	67.98 (+4.79)	73.97 (+0.27)

Apart from the standard KD evaluation setups, we consider an additional teacher-student pair for CIFAR-100 and ImageNet datasets – specifically, ResNet50-ResNet18 and ResNet32x4-ResNet32, respectively – to assess the contribution of pruning in our method. The corresponding results are presented in Table 21, concerning both classification and metric learning tasks. In terms of accuracy, the improvements mirror those seen in the previous experiments (i.e., Table 19 and Table 20)





with gains ranging from +0.09% to +0.87%. Notably, for the metric learning task, we observe significantly higher improvements in terms of mAP and P@100, such as +7.06% mAP⁵ and +4.79% P@100 for CRD+InDistill on CIFAR100. These gains can be attributed to the preservation of information flow paths, enabling better feature learning.

8.4.5. Conclusions

We have introduced the final version of InDistill, a method initially developed during the second year of the project. The finalization process involved conducting a wide range of additional experiments to assess its performance when used combined with recent state-of-the-art KD approaches. InDistill has already been leveraged in T6.2 for compressing synthetic image detection models for use in Android smartphones, as detailed in D6.3.

8.4.6. Relevant publications

- Sarridis, I., Koutlis, C., Kordopatis-Zilos, G., Kompatsiaris, I., & Papadopoulos, S. (2024). InDistill: Information flow-preserving knowledge distillation for model compression. arXiv preprint arXiv:2205.10003. (accepted in WACV 2025)

8.4.7. Relevant software

- The Pytorch implementation can be found in <https://github.com/garridis/indistill>.

8.4.8. Relevance to AI4Media use cases and media industry applications

The presented method supports UC1 (AI for Social Media and Against Disinformation), specifically, the Feature 1A (Detection/Verification of Synthetic Media). In this context, InDistill has been employed in UC1 to compress the synthetic image detection models, while preserving their high accuracy.

8.5. Towards Optimal Trade-offs in Knowledge Distillation for CNNs and Vision Transformers at the Edge

Contributing partner: CERTH

8.5.1. Introduction

The proliferation of IoT and smart devices has sparked a continuous generation of visual data, revolutionizing how we interact with technology, take decisions, and live in smart environments. From commodity cameras to specialized vision sensors, these devices capture a wealth of visual information, ranging from environmental observations to daily activities. By harnessing deep learning algorithms on the edge, we can process and analyze visual data in real time, enabling new capabilities for end users. Keeping data on the edge offers advantages, particularly on privacy, security, and communication, but also imposes computational constraints.

CNNs and Vision Transformers (ViTs) are the cornerstone architectures for various computer vision tasks, exhibiting state-of-the-art performance on numerous benchmarks. Nevertheless, their computational demands often render them impractical to train or deploy on resource-constrained

⁵The presented values are calculated based on cosine similarity. Euclidean distance exhibits almost identical ranking results with InDistill outperforming the rest of the methods and with only a few ranking differences among them. We opt for presenting the former as it consistently provides higher scores for all methods.





devices. On the other hand, there is currently wide availability of powerful pretrained vision models, which have been trained extensively on vast datasets. Such models can be used as *teachers* for transferring knowledge to smaller, compressed *student* models through *knowledge distillation* (KD), which can run on the edge.

While KD is a promising approach for model compression, the computational burden of KD itself has been relatively ignored. Typically, this process is executed at cloud infrastructures and data centers, which offer large amounts of computational resources. In contrast, the execution on edge devices with limited resources, especially for visual data, is relatively unexplored. Our research endeavors to fill this research gap by evaluating the applicability of KD to vision models deployed on computationally-constrained devices. In this work, edge computing refers to the paradigm of decentralized data processing where training, inference and even the compression of artificial neural networks are performed closer to the data capturing end-user devices. The edge computing environments we consider are confined spaces, such as smart homes, wherein computational tasks are executed utilizing one single commercially GPU.

8.5.2. Methodology

Instead of introducing a novel methodology for knowledge distillation, our objective is to address four key questions that are frequently encountered by machine learning practitioners when attempting to implement KD in an edge computing environment characterized by limited computational resources. By delving into these common queries, described in the following paragraphs, we aim to provide valuable insights and guidance for researchers and practitioners regarding the architecture of teacher and student, the size of the student, the impact of the image resolution and the improvement that brings the fine-tuning in the compressed student.

Different ANN Types for Student and Teacher. KD involves transferring knowledge from a teacher artificial neural network (ANN) model to a student ANN model by minimizing the discrepancy between the logits produced by each model. Unlike traditional compression methods, KD can effectively reduce the size of an ANN regardless of structural and typological differences between the teacher and student architecture. The structural dissimilarity between the teacher and student architectures significantly influences the student's performance and the efficacy of the distillation process. Opting for a complex ANN architecture, such as a ViT, the teacher model is anticipated to yield greater robustness. This happens because the knowledge (soft labels) imparted by an advanced teacher can more accurately capture class distributions. However, the use of complex ANNs also incurs longer inference times, thereby impeding the timely flow of information during the distillation process and elevating computational costs.

Similar considerations apply to the selection of the student model. While prior research [443] suggests that ViTs outperform CNNs, one should note that transformers necessitate substantially more time and computational resources for training. While this may not pose a significant challenge in training scenarios supported by powerful cloud infrastructures, it becomes a critical concern in more constrained environments. Given this consideration, the choice of ANN architecture for student and teacher in order for the student to deliver accurate outcomes with limited distillation time and resources warrants further experimental investigation.

Student Model Capacity. The gap of model capacity between the teacher and the student is a critical consideration in the KD process. An optimal balance must be struck to ensure effective distillation outcomes. When the student model's capacity is too low relative to the teacher, the student struggles to effectively incorporate the logits information provided by the teacher. This limitation impedes the student's ability to capture the distilled knowledge, thus hindering its performance.

Conversely, when the student model's capacity is excessively large, the expected improvements





in distillation efficacy may not materialize. Larger student models tend to exhibit slower learning rates and are more susceptible to overfitting, resulting in diminished generalization performance. Furthermore, the computational and memory resources required for training and deploying larger student models pose significant challenges, particularly in resource-constrained environments such as edge devices. The increased computational demands, higher memory footprint, and slower inference speeds associated with larger models render them less practical for deployment in real-world edge scenarios. Therefore, determining the optimal student size also requires experimental investigation.

Higher Resolution Images. The resolution of input images significantly influences the KD process, impacting both model performance and computational efficiency. When utilizing low-resolution images, computational demands are reduced, leading to expedited distillation and inference times. However, this comes at the cost of potential loss of fine-grained details and contextual information, which may hinder model performance, particularly in tasks requiring precise object recognition or classification. Furthermore, low-resolution images may limit the model's ability to generalize to diverse data distributions and discriminate between similar classes.

Models trained on high-resolution images exhibit improved generalization capabilities and enhanced discriminative power, leading to more robust performance. Nonetheless, the computational complexity associated with processing high-resolution images results in longer training and inference times, posing challenges in resource-constrained environments. Additionally, high-resolution images may be more sensitive to noise. This affects model training and performance, while also requiring higher memory resources during deployment.

Fine-tuning the Student Model. Fine-tuning is a widely established technique that is very often employed in compressed models with the aim of enhancing their performance. However, it is important to note that fine-tuning also entails significant time and computational resource consumption. Often likened to a second training stage, fine-tuning necessitates careful consideration from AI practitioners regarding the trade-offs involved. The computational overhead incurred by fine-tuning prompts a reevaluation of its utility and efficacy in optimizing model performance, particularly in resource-constrained environments.

Furthermore, the significance of fine-tuning is heightened in edge computing scenarios. Edge devices, situated at the periphery of the network, have the capability to capture images that are more relevant to the specific user context. Leveraging fine-tuning with these locally captured images enables the adaptation of ANNs to specific needs. This tailored approach not only enhances model performance but also justifies the additional computational resources required for fine-tuning.

8.5.3. Experimental setup

We conducted experiments utilizing various transformer architectures, including the basic ViT and more advanced architectures such as Data-efficient Image Transformer (DeiT), and Swin Transformer [444], alongside the classic VGG [445] CNN architecture. We experimented with various combinations of these models, ensuring that the teachers always had significantly more parameters than the students. For student models, in addition to the standard VGG variations with 11, 13, 16, and 19 layers described in [445], we also conducted experiments with models having 2, 3, 4, 5, and 8 layers.

We opted to utilize the VGG architecture over other state of the art architectures such as EfficientNet and ResNet due to its flexibility in accommodating various sizes, allowing us to investigate the impact of different student sizes in our research. These experiments were carried out using three popular datasets: Cifar10, Cifar100, and the large-scale ImageNet-1k dataset. We utilized the KD approach implemented in the Model Compression Research Package [446] by Intel Labs, which is based on the original paper by Geoffrey Hinton [447]. Utilizing a consumer GPU equipped with 8 GB of memory, we systematically evaluated the performance and efficiency of each





Table 22. Knowledge Distillation between ViTs & CNNs

Teacher	Student	Acc.	Learning (KD) Time
CNN	CNN	Best	Best
CNN	Transformer	Bad	Good
Transformer	CNN	Good	Bad
Transformer	Transformer	Worst	Worst

model architecture across these datasets. By employing a range of datasets, transformers and VGG, our experiments aim to shed light on the four facets of KD for computer vision tasks as described in previous sections.

Before embarking on experiments involving KD, we conducted investigations into training CNN and transformer architectures from scratch on a GPU for limited time periods that range from few up to 24 hours. Our observations revealed that training a CNN model from scratch on a GPU yielded superior performance compared to training a transformer. For the same training periods the transformers’ accuracy was lower compared to the one attained by CNN from 4% to 16%. Despite employing the same dataset across all experiments, the transformer’s training process exhibited a substantially slower learning rate, proving inefficient when constrained to the limited computational setting of a single GPU.

8.5.4. Results

Experiments on Different ANN Types for Student-Teacher. Through experiments with ViT, DeiT, Swin and VGG on Cifar10, Cifar100, and ImageNet-1k, we aimed to explore the efficacy and intricacies of distilling knowledge between established CNN models and transformer architectures. Specifically, we explored the effectiveness of KD: i) from CNNs to CNNs, ii) from CNNs to Transformers, iii) from Transformers to CNNs, and iv) from Transformers to Transformers examining how different models for teachers and students impact the performance (Acc) and efficiency (KD Time) of the distillation process and the compressed models. A qualitative summary of the obtained results is presented in Table 22.

The experimental outcomes underscore that the KD process exhibits differing dynamics when applied between different models for the teacher and student. We observed that the KD process is faster and leads to the best accuracy when implemented between CNNs due to the swift output of logits and the distillation loss calculations. Conversely, transformers require numerous epochs and iterations to effectively learn during the distillation process. This disparity arises primarily from the intricate architecture and larger capacity of transformers compared to CNNs. The transformers’ higher capacity results in longer inference times and makes slower the KD processes compared to CNNs. Consequently, the KD process with transformers is characterized by prolonged learning periods and slower inference times, contrasting with the swifter and more efficient KD process observed with CNNs. In further experiments on Transformers, we also allowed the distillation to run for several days. However, the accuracy improvements were very slow to the point that we considered them impractical when considering an edge environment as a target deployment setting for the KD process.

Experiments on the Student Model Capacity. This subsection delves into the experimental investigation of how varying the size of the student CNN model influences the accuracy and inference time. Through experimentation with student models of different capacities, ranging from compact to more expansive architectures, we aimed to elucidate the trade-offs between model size and performance. Table 23 presents a set of representative experimental outcomes when performing KD





Table 23. Exploring Knowledge Distillation with Various Sizes of CNN Students & Image Resolutions

		Extra Small	Small	Medium	Large	Extra Large
	Params (M)	0.004178	0.015978	0.062474	0.24705	0.982538
size: 32x32	Accuracy	0.591099	0.730499	0.824799	0.875299	0.906099
	Ops (M)	0.163754	0.511818	1.760906	6.470922	24.738314
size: 224x224	Accuracy	0.668199	0.767099	0.824999	0.861899	0.878799
	Ops (M)	8.015786	25.063242	86.253194	317.013258	1212.054026

from DeiT to VGG. The different sizes of VGG are estimated by the million (M) of their parameters (Params). Moreover, we examine the implications of model size on the efficiency of the KD process, including the number of operations (Ops) for student inference.

Through quantitative analysis of accuracy metrics in conjunction with the capacity of student models, it becomes evident that larger CNN models for students, characterized by a higher number of parameters, tend to exhibit superior accuracy. However, this is accompanied by a corresponding increase in computational complexity, as larger models entail a greater number of operations. Specifically, the Extra Large student model comprising approximately 98 million parameters, necessitates approximately 24 million operations for inference while achieving 90% accuracy. The observations in this and previous subsections highlight the value of allocating edge resources for a relatively large CNN model as a student, in contrast to the use of transformers.

Experiments on Higher Resolution Images. We investigated the impact of employing higher resolution images on the accuracy, workload and memory resources. In Table 23, we present a series of experiments conducted with image resolutions 32 x 32 and 224 x 224. In both cases we kept the duration of KD constant. The analysis of results reveals an interesting trend: For student models with smaller capacities (extra small and small), employing higher resolution images leads to improved accuracy. However, this pattern is reversed for larger student models (large and extra large), where lower resolution images yield higher accuracy. This phenomenon can be attributed to the slower learning rates observed in larger student models when processing higher resolution images. The intricacies of high-resolution data introduce greater complexity to the learning process, causing larger models to struggle to converge effectively and ultimately resulting in lower accuracy compared to their smaller counterparts.

In terms of memory footprint, a single RGB image with a resolution of 32 x 32 pixels and 24 bits per pixel (bpp) consumes 3072 bytes, while the same image at a resolution of 224 x 224 pixels requires 150,528 bytes. Similarly, grayscale images with 8 bpp exhibit notable differences in size, with a 32 x 32 resolution image occupying 1024 bytes and a 224 x 224 resolution image consuming 50,176 bytes. These numbers underscore that the memory requirements for both RGB and grayscale images are substantially larger, approximately 49 times, for the 224 x 224 resolution compared to the 32 x 32 resolution. A similar trend is observed in computational workload, where operations required for 224 x 224 images range from 49 to 52 times those required for their 32 x 32 counterparts. These findings lead us to conclude that utilizing relatively large CNN models for students while employing lower image resolutions achieves the optimal balance between performance, memory usage, and computational efficiency.

Experiments with Fine-tuning the Student Model. Fine-tuning in ViT and VGG-CNNs involves selectively training certain parts of the student model while keeping other parts frozen to preserve the knowledge captured during KD. In ViT the process starts by freezing the transformer layers to maintain the learned representations and only fine-tuning the task-specific layers namely the final classification head. Similarly, in VGG-CNNs, the convolutional layers are kept frozen to





Table 24. Enhancement in Accuracy with Fine-Tuning on CNNs & ViTs

	Params (M)	Accuracy	
		Prior Fine-Tuning	Post Fine-Tuning
VGG	24.73	0.9060	0.9342
Swin-T	26.60	0.8475	0.9616
ViT	85.27	0.8600	0.9814

retain their feature extraction capabilities, while the fully connected layers and fine-tuned. The learning rate, which controls the step size at each training iteration, is set to lower values during fine-tuning to ensure that the adjustments to the pre-trained weights are limited and do not disrupt existing knowledge gained by KD.

We present our findings on the impact of fine-tuning the student model derived from the KD process in Table 24. It is evident that fine-tuning transformers results in significant improvements compared with CNNs. Specifically fine-tuning the Swin-T model led to a substantial increase in accuracy of approximately 16%. Similarly, when fine-tuning the ViT model, we observed a notable improvement of almost 12% in accuracy. On the other hand, we observed only small improvements (approximately 3%) when fine-tuning the VGG model. These were consistently observed across all models when utilizing the CIFAR-10 dataset. We used different parts of the dataset for distillation, fine-tuning and evaluation. In all experiments, the fine-tuning process occurred over the course of 15 epochs.

8.5.5. Conclusions

In this work, we present insights for researchers and practitioners aiming to compress and deploy ANNs in edge environments under constrained computational resources and time limitations. Despite the prevailing perception of the superior accuracy of ViTs, our empirical evidence indicates that utilizing relatively large CNNs with low resolution images presents a more efficient and feasible approach to derive highly performing and efficient compressed student models. Only when fine-tuning is applied after KD, we observe that transformer students surpass CNNs, albeit with a significant increase in resource consumption and time delay.

8.5.6. Relevant publications

- Violos, J., Papadopoulos, S., & Kompatsiaris, I., Towards Optimal Trade-Offs in Knowledge Distillation for CNNs and Vision Transformers at the Edge. In Proceedings of the 32nd European Signal Processing Conference, EUSIPCO 2024, August 26-30, 2024 Lyon, France. Zenodo preprint: <https://zenodo.org/records/12510505> .

8.5.7. Relevance to AI4Media use cases and media industry applications

Our research work can be integrated into all AI4Media use cases that have a component where a convolutional model or Vision Transformer need to be transformed into a lightweight version through knowledge distillation, such as Use Case 2 (AI for News - The Smart News Assistant) and Use Case 3 (AI in Vision).

8.6. FPGM pruning for lightweight mobile face detection models

Contributing partner: CERTH





8.6.1. Introduction and methodology

Even though face detection is a mature technology and the backbone of numerous applications, the rise of edge computing has necessitated the usage of efficient and compact face detectors. One promising solution is network pruning [448], which has achieved considerable success in tasks like image classification with various techniques proposed over the years. Yet, the application of pruning in face detection and its potential benefits remain largely uncharted.

In our work, we have applied the Filter Pruning via Geometric Median (FPGM) [449] algorithm to prune two already compact and small, in terms of parameters, face detectors, namely, EXTD (Extremely Tiny Face Detector) [450] and EResFD (Efficient ResNet Face Detector) [451]. FPGM identifies and prunes the filters with the “most redundancy”, a principle that has shown improved performance over other pruning algorithms in the literature. This is compared against the widely used L1 Norm pruning criterion as baseline [452]. In both cases, we apply the Soft Filter Pruning (SFP) paradigm [453], an iterative procedure which updates the pruned parameters in subsequent training steps instead of freezing their values to zero. The main benefits of SFT are larger model capacity and less dependence on the pre-trained model [453]. For evaluation, we use the WIDER FACE face detection dataset [454], in particular, we split the validation set of WIDER FACE into three subsets based on difficulty (Easy, Medium and Hard), and report the Mean Average Precision (MAP) metric on each subset.

In our original experiments, we pruned the entire network using a uniform pruning rate. However, pruning some layers might affect the performance of the network more than others. Hence, in order to improve the performance of the original methodology, we split the network’s layers into groups, each one with its individual and independent pruning rate, and optimize those pruning rates using Bayesian optimization. The selection of the optimization objective function is critical. Its value must be inexpensive to evaluate, as it needs to be assessed in every iteration during the optimization process. The chosen objective function is the validation loss of the pruned network after one epoch of training, supplemented by an additional term to ensure that the network is pruned approximately at the target pruning rate. After the optimization, the original training pipeline is followed using the pruning rates suggested by the optimization, instead of a uniform pruning rate for the entire network. The experiments were conducted on the EResFD face detection model.

It is important to note that this methodology solely optimizes the network layer pruning rates and still requires an additional algorithm to define the pruning process. In our experiments, the FPGM pruning algorithm is utilized, however, the methodology is algorithm-agnostic and can be applied with any pruning algorithm.

8.6.2. Algorithm

Consider an individual convolutional layer in a face detector with weight parameters,

$$F = [F_1, \dots, F_n], \quad (11)$$

where, $F_j \in \mathbb{R}^{k \times k \times c}$ is the j th filter with spatial size $k \times k$ and depth c , and n is the total number of filters in the layer. Based on the above formulation, the goal of the proposed approach is: given a filter pruning rate θ , prune the $n\theta$ filters in each layer of the face detector.

The L1 Norm filter prunes the F_j filters with the lowest L1 norm, while FPGM prunes the filters that it deems redundant, by computing the geometric median of the filters and pruning the ones closest to it. The training process consists of three steps: pre-training, iterative training and soft pruning, and fine-tuning. The complete training algorithm is presented in Algorithm 4.





Algorithm 4 Face detection neural network training and pruning pipeline

```

Initialize neural network  $\theta$ 
train  $\theta$  for 300 epochs
for epoch in  $[0, 199]$  do
  if epoch  $\bmod 5 = 0$  then
    soft-prune  $\theta$ 
  end if
  train  $\theta$  for one epoch
end for
prune  $\theta$  and freeze pruned parameters
train  $\theta$  for 10 epochs
  
```

In Algorithm 4, the pruning can either be uniform across the entire network or non-uniform based on the Bayesian optimization result. If the pruning rates are optimized using Bayesian optimization, this process is conducted after the 200 epochs of pre-training, using Algorithm 5 with Algorithm 6 as the objective function to be optimized.

Algorithm 5 Bayesian optimization

```

Require: Initial number of points  $n_0$ 
Require: Total number of iterations  $N$ 
Require: Objective function  $f$ 
Evaluate  $f$  at  $n_0$  random initial points
 $n \leftarrow n_0$ 
while  $n \leq N$  do
  Update the posterior probability distribution on objective function  $f$  using all available data
  Sample point  $x_n$  using the acquisition function
  Evaluate  $f(x_n)$ 
end while
  
```

Algorithm 6 Bayesian optimization pruning objective function

```

Require: Target pruning rate percentage  $t$ 
Require: Sparsity threshold  $thres$ 
Require: Pre-trained neural network  $\theta$ 
Require: Loss function  $f_{loss}$ 
Require: Sparsity penalty function  $g(actual\_sparsity, target\_sparsity)$ 
 $\theta' = \text{copy}(\theta)$ 
Prune neural network  $\theta'$ 
Evaluate sparsity of  $\theta'$ 
if  $sparsity < t - thres$  or  $sparsity > t + thres$  then
  value = 100
else
  train  $\theta'$  for one epoch
  value =  $f_{loss}(\theta') + g(sparsity, t)$ 
end if
return value
  
```





Table 25. Comparative MAP results on EXT D between the proposed pruning approach (FPGM-based) and our baseline.

Method	Easy	Medium	Hard	# of Parameters	Real Sparsity
EXTD(orig.)	0.9210	0.9110	0.8560	162,352	0%
EXTD(rep.)	0.8961	0.8868	0.8268		
FPGM 10%	0.8988	0.8828	0.8026	149,472	7.93%
L1 10%	0.8950	0.8766	0.7961		
FPGM 20%	0.8931	0.8789	0.7992	136,296	16.05%
L1 20%	0.8921	0.8766	0.7923		
FPGM 30%	0.8885	0.8588	0.7168	122,034	24.83%
L1 30%	0.8806	0.8522	0.6655		
FPGM 40%	0.8539	0.8213	0.6915	108,858	32.95%
L1 40%	0.8427	0.8068	0.6544		
FPGM 50%	0.8485	0.8118	0.6565	94,448	41.83%
L1 50%	0.8422	0.7971	0.6267		

8.6.3. Experimental results

We compare the FPGM pruning algorithm with the widely used L1 Norm pruning criterion on two compact and small face detector models, EResFD and EXT D. These comparisons are reported in Tables 25 and 26. Additionally, in Table 27 we compare universal FPGM pruning with FPGM pruning that uses pruning rates optimized through Bayesian optimization. All the experiments were conducted with the WIDER FACE face detection dataset and various target pruning rates.

8.6.4. Conclusion

We have investigated the effects of network pruning on face detection models to make them more lightweight and suitable for deployment on mobile or other lightweight devices. The results indicate that FPGM pruning is an effective method for reducing the number of parameters without significantly affecting performance. Additionally, dividing the network’s layers into groups and optimizing the pruning rate for each group proved to be superior to universal FPGM pruning.

8.6.5. Relevant publications

- Gkrispanis, K., Gkalelis, N., Mezaris, V. Filter-Pruning of Lightweight Face Detectors Using a Geometric Median Criterion. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision 2024 (pp. 280-289). arxiv link: <https://zenodo.org/records/13143850>

8.6.6. Relevant software

The Python implementation of the FPGM based iterative training and pruning can be found on the github repository at <https://github.com/IDT-ITI/Lightweight-Face-Detector-Pruning>.





Table 26. Comparative MAP results on EResFD between the proposed pruning approach (FPGM-based) and our baseline.

Method	Easy	Medium	Hard	# of Parameters	Real Sparsity
EResFD(orig.)	0.8902	0.8796	0.8041	92,208	0%
EResFD(rep.)	0.8660	0.8555	0.7731		
FPGM 10%	0.8728	0.8582	0.7757	87,368	5.25%
L1 10%	0.8470	0.8345	0.7410		
FPGM 20%	0.8369	0.8201	0.7230	76,677	16.84%
L1 20%	0.8263	0.8038	0.6723		
FPGM 30%	0.8311	0.8160	0.7175	69,746	24.36%
L1 30%	0.8218	0.8001	0.6663		
FPGM 40%	0.8124	0.7952	0.6807	57,055	35.95%
L1 40%	0.7603	0.7349	0.5800		
FPGM 50%	0.7103	0.6830	0.5254	47,284	48.72%
L1 50%	0.6992	0.6704	0.4824		

Table 27. Comparative MAP results on EResFD between universal FPGM pruning with FPGM pruning that uses pruning rates optimized through Bayesian optimization.

Method	Easy	Medium	Hard	Sparsity
EResFD (orig.)	0.8902	0.8796	0.8041	0%
EResFD(rep.)	0.8660	0.8555	0.7731	0%
optimized FPGM 10%	0.8622	0.8506	0.7636	10.24%
universal FPGM 10%	0.8728	0.8582	0.7757	5.25%
optimized FPGM 20%	0.8518	0.8408	0.7554	21.68%
universal FPGM 20%	0.8369	0.8201	0.7230	16.84%
optimized FPGM 30%	0.8348	0.8266	0.7231	31.59%
universal FPGM 30%	0.8311	0.8160	0.7175	24.36%
optimized FPGM 40%	0.8227	0.8192	0.7205	40.02%
universal FPGM 40%	0.8124	0.7952	0.6807	35.95%
optimized FPGM 50%	0.7976	0.7910	0.6876	52.61%
universal FPGM 50%	0.7103	0.6830	0.5254	48.72%





8.6.7. Relevance to AI4Media use cases and media industry applications

The FPGM pruning algorithm is an effective method to reduce the number of parameters in face detection models and potentially any convolutional neural network. Therefore, it can be integrated into all AI4Media use cases which have a component where a convolutional model needs to be lightweight or be deployed in a mobile device.

8.7. Porting Large Language Models to Mobile Devices for Question Answering

Contributing partner: JR

8.7.1. Introduction

Large language models (LLMs) [455] on end-user devices offer sophisticated natural language processing and more intuitive interactions with the device. These models support applications such as advanced virtual assistants, language translation, text summarization and named entity or keyword extraction. Another important use case is *question answering*, which can provide accurate and contextually relevant answers to a wide array of user queries. For example, it can be used for fake news detection by querying the LLM about the validity of dubious claims made in a news article.

Because of the limited processing power of a typical smartphone, user queries are usually processed in the cloud and the answers are sent back to the device. This workflow is standard for ChatGPT and other LLM apps but not always possible or desirable, for example, for journalists operating in areas with limited connectivity or under strict monitoring and surveillance of Internet traffic (e.g. in authoritarian regimes). In this case, the processing has to be done on the device. In the following, we demonstrate how to port LLMs efficiently to mobile devices so that they run natively and in interactive speed.

8.7.2. LLM framework for on-device inference

LLM inference can run natively on a mobile device via *Tensorflow Lite (TFLite)*, which is the most popular framework for on-device inference. Since most publicly available fine-tuned models (e.g., from the *Hugging Face model hub*⁶) provide only PyTorch weights, a conversion to TFLite is necessary. This pipeline however is complex (PyTorch → ONNX → Tensorflow → TFLite) and not future-proof, as legacy TensorFlow 1.X versions have to be used along several unmaintained code repositories.

We therefore propose using *llama.cpp*⁷, a flexible and self-contained C++ framework for LLM inference on a variety of devices. This framework runs state of the art models like *Llama / Llama2* [456], *Vicuna* [457] or *Mistral* [458] either on CPU or GPU/CUDA, and offers multiple configuration options (e.g. for temperature, context size or sampling method). It also supports a variety of sub-8bit quantization methods, from 2 bits to 6 bits per parameter, which is crucial for running models with billions of parameters on smartphones with limited memory.

In order to build the C++ libraries and executables of the *llama.cpp* framework, a standard Linux build toolchain is required comprising a terminal (shell), command-line tools, CMake/Make, C/C++ compiler and linker and more. For Android, such a build toolchain is available via the *Termux* app⁸, which has already been used for deep learning tasks, e.g., for on-device training

⁶<https://huggingface.co/docs/hub/models-the-hub>

⁷<https://github.com/ggerganov/llama.cpp>

⁸<https://termux.dev/en/>





of neural networks [459]. Termux can be installed via the Android open-source software package manager *F-Droid* without root access requirements. After installation, further necessary tools like *wget*, *git*, *cmake*, *clang* compiler can be installed via *pkg*, the Termux package manager.

For building the *llama.cpp* binaries, we clone its latest sources from the respective github repo. We invoke *CMake* to generate the Makefile and build all binaries via the *make* command. We compile the binaries with model inference done on the CPU, as GPU inference relies on CUDA which is not available on Android devices. After compiling, several binaries are available on the device. The most important ones are an executable for direct interactive chatting with the LLM and a server application with an *REST-API* which is similar to the OpenAI API for ChatGPT. The server application allows for further integration, for example into a on-device GUI app for question answering.

8.7.3. Model selection and prompt format

On the Hugging Face model hub, there are many pretrained large language models available which differ in size, architecture, training / fine-tuning method and datasets, and task (e.g., base model for text completion versus instruct model for instruction following and chat). After some experiments, we have selected the Orca-Mini-3B model with 3 billion parameters. It runs in interactive speed on a recent smartphone and provides decent responses to a user query due to fine-tuning via imitation learning with the Orca method [6]. We employ a quantized model with approximately 5.6 bits per parameter, which takes roughly 2.2 GB of CPU RAM on the device. For an instruct model, it is important to maintain the same prompt format (system prompt, user prompt etc.) as the one used during fine-tuning to ensure good performance. For the Orca-Mini-3B model, the appropriate prompt format for the server application is as follows:

```
### System:\n You are an AI assistant that follows instruction extremely well.  
Help as much as you can.\n\n ### User:\n What is the smallest state in India ?\n\n ### Response:\n'
```

8.7.4. Experiments

We perform a subjective evaluation of the selected model's responses on questions from diverse subjects like politics, geography, history and more. Our tests have shown that the model provides accurate and faithful answers for most of the user queries, with occasional hallucinations (i.e., synthesis of false information) which plague all LLMs. In addition, the responses are generated fast enough for an interactive chat on a Samsung Galaxy S21 smartphone. An example output of the LLM application for direct chat can be seen in Fig. 27.

8.7.5. Conclusion

We have demonstrated how to port LLMs to run natively and efficiently on mobile devices. In particular, we have described our LLM framework, as well as the model and prompt format used for question answering. Our experiments have shown that the model runs in interactive speed on a Galaxy S21 smartphone and provides high-quality answers for the user queries. In the future, we will explore recently introduced LLMs like phi-2 [460] and GPU acceleration of the model via OpenCL or Vulkan on the device.

8.7.6. Relevant publications

- Fassold, H. Porting Large Language Models to Mobile Devices for Question Answering. In Proceedings of International Conference on Advances in Signal Processing and Artificial





```

17:14 4G 63%
== Running in interactive mode. ==
- Press Ctrl+C to interject at any time.
- Press Return to return control to LLaMa.
- To return control without starting a new line
, end your input with '/'.
- If you want to submit another line, end your
input with '\'.

> What is the smallest state in India ?
The smallest state in India is Goa.

> Can you explain quantum computing in simple te
rms ?
Sure! Quantum computing is a way of processing
information using quantum-mechanical phenomena,
such as superposition (the ability to exist in m
ultiple states) and entanglement (the phenomenon
where two particles become linked so that chang
ing one affects the other). This is different fr
om classical computing.
ESC / - HOME ↑ END PGUP
↵ CTRL ALT ← ↓ → PGDN

```

Figure 27. Interactive chat application on a smartphone

Intelligence (ASPAI), 2024. Zenodo link: <https://zenodo.org/records/11058352>

8.7.7. Relevance to AI4media use cases and media industry applications

The developed work is relevant for Use Case 1 (AI for Social Media and Against Disinformation). It was successfully integrated by ATC into the prototype mobile application for fact checking (see D8.5).

8.8. AdaFamily: A family of adaptive gradient methods for training

Contributing partner: JR

8.8.1. Introduction

Adaptive gradient methods, especially the *Adam* algorithm [461], are nowadays the standard for training deep neural networks, both on a single machine and distributed training. This is because they are less sensitive to weight initialization and hyperparameters compared with mini-batch stochastic gradient descent (SGD) [462]. This can be attributed to two key properties of the Adam algorithm: *adaptive learning rate* and *momentum term*. Via the *adaptive learning rate*, the algorithm computes an individual learning rate for each parameter of the neural network model, in contrast to SGD which keeps a fixed learning rate for all parameters of the model. This is especially helpful for neural networks with many different types of layers, with the parameters





in each layer typically having a different representative value range. On the other side, via the *momentum term* it can keep progress even when encountering regions of the loss function landscape with high curvature, which resemble long and narrow ravines (mathematically speaking, these are regions where the Hessian matrix has a high condition number).

Since the introduction of the Adam algorithm, many variations of the algorithm have been proposed (see Table 2 in the survey paper [463]). Although many of them might have marginal benefits in practice [463], a few of them seem to be able to really provide an improvement of the training process when compared with Adam. Specifically, [464] demonstrates that L_2 regularization and weight decay are *not* equivalent for adaptive gradient methods, and proposes the *AdamW* algorithm, which decouples the weight decay from the gradient-based update. The *AdaBelief* algorithm [465] modifies the calculation of the scaling term v_t (corresponding to the denominator in the weight update step of the Adam algorithm) in a way which takes into account the "belief" in the gradient. Experiments with a variety of models (CNN, LSTM, GAN) show that this modification improves the performance of the training. Recently, the *AdaMomentum* algorithm [466] has been proposed that also modifies the calculation of the scaling term v_t by replacing the gradient with its exponential moving average (EMA). They also propose to move the addition of the constant ϵ , which prevents division by zero, to a different term and justify why this is advantageous.

AdaBelief and AdaMomentum differ from Adam in only a few places, most prominently in the term corresponding for the scaling term v_t . The scaling term v_t can be interpreted also as a diagonal preconditioner [467] which is applied (left-multiplied) to the gradient prior to the weight update, so these algorithms differ mainly by the way how the preconditioner is calculated. Inspired by this observation and by the question whether we can "blend" these algorithms together in a useful way, we propose *AdaFamily*, a family of Adam-like algorithms parametrized by a hyperparameter μ lying in range $[0, 1]$. All variants of the AdaFamily algorithm can be seen in a certain way as a *blend* of the Adam/AdamW, AdaBelief and AdaMomentum algorithm.

8.8.2. Algorithm

In the following, we formally define the AdaFamily optimization algorithm for training a deep neural network. We first describe the used notation and provide the pseudo code of the AdaFamily algorithm. After that, we elaborate on how AdaFamily is related to Adam/AdamW, AdaBelief and AdaMomentum and how the hyperparameter μ can be interpreted.

Regarding *notation*, we denote by t the current iteration (current step) in the training process. $\theta \in \mathbb{R}^d$ denotes the model parameter and $f(\theta) \in \mathbb{R}$ denotes the loss function. We further use θ_t to denote the parameter at step t and f_t to denote the noisy realization of f at time t due to the stochastic mini-batch mechanism. The gradient of f_t is denoted by g_t , and α is the step size (learning rate). m_t represents the exponential moving average of the gradient, whereas v_t corresponds to the scaling term (preconditioning term). ϵ is a small constant number added in adaptive gradient methods to refrain the denominator from being too close to zero. β_1, β_2 are the decaying parameter in the EMA formulation of m_t and v_t correspondingly. For any vectors $a, b \in \mathbb{R}^d$, we employ $\sqrt{a}, a^2, |a|, a/b$ for *elementwise* square root, square, absolute value or division, respectively.

The overall workflow of the AdaFamily algorithm is given as pseudo code in Algorithm 7. One can see that the main difference to Adam and its variations is the term (we will denote it by S) which is element-wise squared in the computation of the preconditioner v_t . The variable c can be seen as a normalization factor, where $c(\mu)$ is a slightly modified triangle function which returns 1.0 for $\mu = 0.0$ or $\mu = 1.0$ and 2.0 for $\mu = 0.5$.

We will now calculate S for different values of μ , which allows us to see the relation between AdaFamily and Adam/AdamW, AdaBelief and AdaMomentum. We furthermore will denote with





Algorithm 7 AdaFamily algorithm.

The main differences to the Adam algorithm are marked in blue.

Require: μ : hyperparameter in range $[0, 1]$

Require: $f(\theta)$: objective function (loss function) with model parameters θ

Require: α : stepsize (learning rate)

Require: β_1, β_2 : exponential decay rates

Require: θ_0 : initial parameters

$m_0 \leftarrow 0, v_0 \leftarrow 0, t \leftarrow 0$

$c \leftarrow 2 \cdot (1 - |\mu - 0.5|)$

while not converged **do**

$t \leftarrow t + 1$

$g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$

$m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$

$v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) (c \cdot ((1 - \mu) g_t - \mu m_t))^2 + \epsilon$

$\hat{m}_t = m_t / (1 - \beta_1^t), \hat{v}_t = v_t / (1 - \beta_2^t)$

$\theta_t \leftarrow \theta_{t-1} - \alpha (\hat{m}_t / \sqrt{\hat{v}_t})$

end while

$AdaFamily_{(\mu)}$ the respective variant of the AdaFamily algorithm for a specific μ .

- For $\mu = 0.0$, S is equal to g_t . So the variant $AdaFamily_{(0.0)}$ is similar (but not identical) to the *Adam/AdamW* algorithm.
- For $\mu = 0.5$, S is equal to $g_t - m_t$. So the variant $AdaFamily_{(0.5)}$ is similar (but not identical) to the *AdaBelief* algorithm.
- For $\mu = 1.0$, S is equal to $-m_t$. So the variant $AdaFamily_{(1.0)}$ is similar (but not identical) to the *AdaMomentum* algorithm.
- For all other values of μ , the variant $AdaFamily_{(\mu)}$ can be seen as a "blend" (mixture) of different Adam-variants. For example, $AdaFamily_{(0.25)}$ can be interpreted as a blend of Adam and AdaBelief, with both contributing "equally" in some way to the blend. In the same way, $AdaFamily_{(0.75)}$ can be seen as a blend of AdaBelief and AdaMomentum.

The reason why a specific AdaFamily variant is similar but not exactly identical to Adam and its variants is because the constant ϵ is added in different places in the respective algorithms. The placement of ϵ has an influence [466] on the algorithm performance. For AdaFamily, we follow the procedure proposed (and justified) in AdaMomentum and add ϵ in the preconditioning term v_k .

One can see that with AdaFamily we get an infinite amount of variants of Adam-like algorithms, parametrized via the hyperparameter μ . The hyperparameter μ determines how "close" the respective AdaFamily variant is to either Adam/AdamW, AdaBelief or AdaMomentum.

8.8.3. Experimental results

We evaluate our proposed AdaFamily algorithm on the task of image classification. We compare the AdaFamily variants for $\mu \in \{0.0, 0.25, 0.5, 0.75, 1.0\}$ against the Adam [461], AdamW [464], AdaBelief [465] and AdaMomentum [466] algorithm. We employ three standard datasets for image classification: SVHN [468], CIFAR-10 and CIFAR-100 [469]. The datasets consist of 32x32 pixel RGB images, which belong to either 10 classes (SVHN, CIFAR-10) or 100 classes (CIFAR-100). For all algorithms, learning rate α is set to the default value 10^{-3} and weight decay is set to 10^{-4} . For





all algorithms except for Adam, decoupled weight decay is employed (as proposed in AdamW). The exponential decay parameters are set to their defaults $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for all algorithms. The constant ϵ is set to its default value 10^{-8} . The mini-batch size is 128 and training is done for 150 epochs, with the learning rate decayed by a factor of 0.5 at epochs 50 and 100.

We perform the experiments with four popular neural network models for computer vision, taking into account both standard networks (with medium complexity) as well as light-weight networks designed for mobile devices. We employ ResNet-50 [470] and DenseNet-121 [471] as standard network models, whereas MobileNetV2 [472] and EfficientNet-B0 [473] are taken as representatives of light-weight networks. To measure the performance of a certain algorithm, we utilize the top-1 classification error of the final trained model on the test set (which of course has not been seen during training). We do 10 different runs with random seeds and take the average of these 10 runs.

The results of the experiments are shown in Table 28 for CIFAR-10 and Table 29 for CIFAR-100. One can observe that the variants of the AdaFamily algorithm are outperforming Adam/AdamW, AdaBelief and AdaMomentum in most cases. For all datasets and all network models, a specific variant of AdaFamily is either the best-performing or second-best algorithm.

Regarding how to choose the hyperparameter μ for the AdaFamily algorithm, one can draw the conclusion that smaller values of μ perform better for standard models (ResNet-50, DenseNet-121), whereas larger values of μ perform better for light-weight models (MobileNetV2, EfficientNet-B0). Based on this, a rough guideline could be to employ the variant $AdaFamily_{(0.25)}$ for standard models (with medium complexity) and $AdaFamily_{(0.75)}$ for light-weight models. As mentioned in the description of the AdaFamily algorithm, $AdaFamily_{(0.25)}$ can be interpreted as a blend of Adam/AdaBelief, whereas $AdaFamily_{(0.75)}$ can be seen as a blend of AdaBelief/AdaMomentum.

8.8.4. Conclusion

We have presented *AdaFamily*, a family of adaptive gradient methods parameterized by the hyperparameter μ , which can be interpreted as sort of a *blend* of the optimization algorithms Adam, AdaBelief and AdaMomentum. Experiments on the task of image classification have demonstrated that our proposed method outperforms these algorithms in most cases.

In the future, we will port the AdaFamily algorithm to a distributed training framework like *DeepSpeed* or *Colossal-AI*. The combined formulation allows for a faster porting of the algorithm to a distributed training framework, as only one Algorithm (AdaFamily) has to be ported instead of multiple ones.

Table 28. Results (measured as test error, in percent) on the CIFAR-10 dataset for different models. The best and second-best result for each model is marked in orange and blue.

Algorithm	ResNet-50	DenseNet-121	MobileNetV2	EfficientNet-B0
Adam	12.89	10.31	14.33	21.18
AdamW	13.27	9.32	15.18	21.41
AdaBelief	12.70	8.93	14.97	21.45
AdaMomentum	14.11	9.48	14.15	19.56
AdaFamily _(0.0)	12.69	8.93	15.07	21.61
AdaFamily _(0.25)	12.71	8.89	15.34	22.29
AdaFamily _(0.5)	12.65	8.92	14.85	21.55
AdaFamily _(0.75)	13.79	9.21	14.19	19.36
AdaFamily _(1.0)	14.56	9.50	14.18	19.67





Table 29. Results (measured as test error, in percent) on the CIFAR-100 dataset for different models. The best and second-best result for each model is marked in *orange* and *blue*, respectively.

Algorithm	ResNet-50	DenseNet-121	MobileNetV2	EfficientNet-B0
<i>Adam</i>	41.11	32.91	42.84	52.55
<i>AdamW</i>	39.78	31.55	44.88	53.13
<i>AdaBelief</i>	40.15	29.40	42.66	51.85
<i>AdaMomentum</i>	43.72	30.37	40.20	50.58
<i>AdaFamily</i> _(0.0)	39.05	29.39	43.09	52.75
<i>AdaFamily</i> _(0.25)	38.81	29.21	43.37	53.25
<i>AdaFamily</i> _(0.5)	39.43	29.46	42.71	52.05
<i>AdaFamily</i> _(0.75)	42.05	30.15	40.47	50.17
<i>AdaFamily</i> _(1.0)	43.35	30.32	40.06	50.31

8.8.5. Relevant publications

- Fassold, H. AdaFamily: A family of Adam-like adaptive gradient methods. In Proceedings of International Conference on Intelligent Systems and Patterns Recognition (ISPR), 2022. arxiv link: <https://arxiv.org/abs/2203.01603>

8.8.6. Relevant software

The Python implementation of the AdaFamily algorithm (as well as the other algorithms used in the evaluation like AdaMomentum, AdaBelief etc.) can be found in the github repository at https://github.com/hfassold/omni_optimizer

8.8.7. Relevance to AI4Media use cases and media industry applications

The AdaFamily algorithm is a better alternative to commonly used adaptive gradient optimization algorithms like Adam, both for training on a single machine as well as for distributed training. Therefore, it can be integrated into all AI4Media use cases which have a component where a model is either trained or fine-tuned. For example, it can be used in Use Case 6 (AI for Human Co-creation) to train models for composing music and for audio analysis.





9. Deep quality diversity (Task 3.6)

Contributing partners: UM

QD algorithms have been recently introduced to the Evolutionary Computation EC literature as a way of handling deceptive search spaces. The goal of these algorithms is “to find a maximally diverse collection of individuals (with respect to a space of possible behaviours) in which each member is as high performing as possible” [25]. The inspiration for such approaches is natural evolution, which is primarily open-ended—unlike the objective-based optimization tasks to which EC is often applied. While the rationale of open-ended evolution has been previously used as an argument for genetic search for pure behavioural novelty, QD algorithms re-introduce a notion of (localized) quality among individuals with the same behavioural characteristics. QD algorithms attempt to balance between their individuals’ quality and their population’s diversity, and thus media content which have strict quality requirements, such as games that are playable from start to finish, are the ideal arena for advancing quality-diversity.

The aim of Task 3.6 is to couple DNN architectures with divergent search for transforming exploration, aiming for both diverse and high quality outcomes. Experiments in this deep-learning-based QD search (*deepQD*) approach during the reported period are aligned on two main directions:

D1 improve the definition of diversity based on learnt representations.

D2 promote diversity and quality in existing deep learning generative architectures for media.

9.1. Multimodal Quality Diversity in Creative Domains

Contributing partner: UM

9.1.1. Introduction and methodology

Evolutionary search in creative domains such as visual, audio, or text generation has traditionally struggled to evaluate the artefacts it produces. This is mostly because there is no universal metric to assess the quality of media content [474, 475]. Early approaches relied on ad-hoc metrics such as timing intervals in music generation [476] and compression-based indices for image generation [477], or tasked humans to evaluate the evolving population [478–481]. As more refined deep learning algorithms became available, models trained for a specific task have been employed more frequently as a fitness measure of evolved artefacts [482, 483].

For generative media, the most interesting development in the field of deep learning is the training and release of multimodal models. These models map multiple modalities to the same latent space, thereby enabling the direct comparison of different types of media. Contrastive Language-Image Pre-Training (CLIP) [484] was one such model which demonstrated excellent zero-shot image classification to any input set of semantic labels. Similar models map text with other modalities, such as audio [485], which in unison with CLIP (or similar models) may compare images to another modality via an intermediary text modality. Alternatively, models such as Meta’s ImageBind [486] directly combine multiple input and output modalities into a single embedding space, which facilitates *multimodal generation* and assessment but also opens up new possibilities for cross-modal learning and transfer learning.

One of the prominent QD algorithms is Multi-dimensional Archive of Phenotypic Elites (MAP-Elites) [487]. MAP-Elites partitions the solution space into a multi-dimensional grid (the *feature map*), where each axis represents varying properties within a specific behavioural characteristic (BC) or phenotypic trait of the solutions. Each cell stores the optimal individual (elite) according



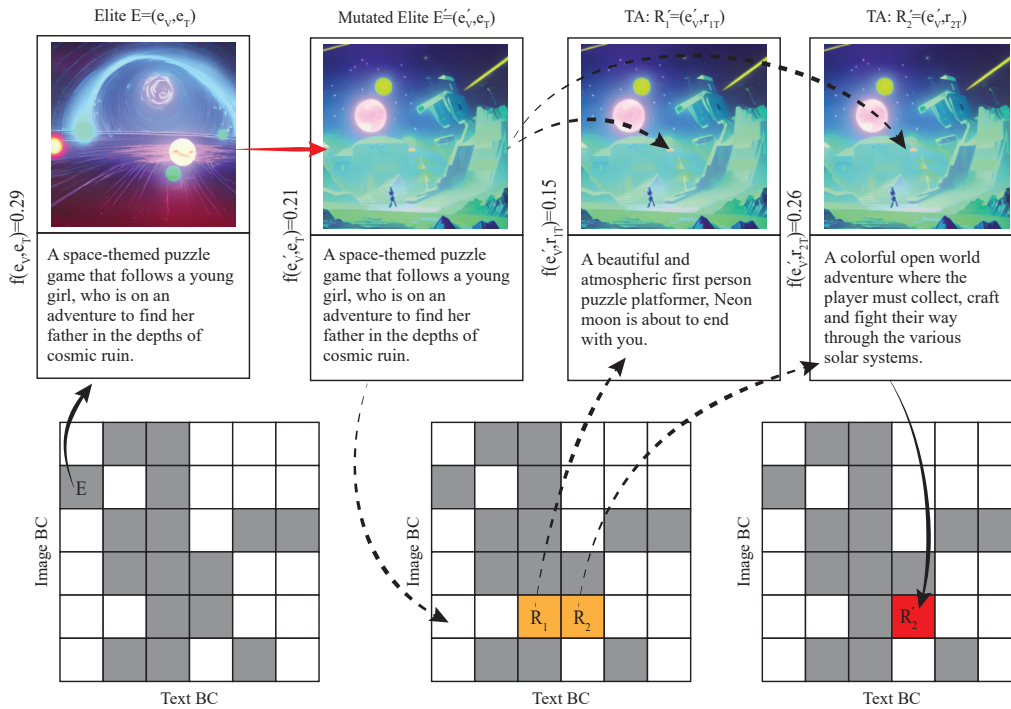


Figure 28. The MELiTA process in a simplified feature map for this use case, with grey cells occupied by elites. From one selected elite E , the changed image (e'_v) produces three candidate solutions from elites E , R_1 , R_2 . Based on their CLIP score, the ordered list of candidates is $\vec{L} = \{R'_2, E', R'_1\}$. Since $q(R'_2) > q(R_2)$ the candidate R'_2 (that merges the image from E' and text from R_2) replaces R_2 . If $q(R'_2) \leq q(R_2)$ then E' would occupy the empty cell at (5,0). Dotted lines denote temporary individuals that are lost after this parent selection.

to the global fitness function, promoting only competition within the phenotypic niche. The most popular implementation of MAP-Elites operates in a steady-state fashion, selecting a parent among the elites (at random) and mutating it to produce an offspring. The offspring is then mapped to a cell of the feature map according to its BCs and may replace the elite in that cell if it has a higher fitness. As MAP-Elites illuminates a problem space, it is particularly apt for creative domains where it has already shown successes [488–493].

This work extends the MAP-Elites algorithm to a multimodal creative domain, specifically generating text descriptions and cover images for hypothetical video games. To address this challenge, we propose an algorithmic improvement on QD search: MAP-Elites with Transverse Assessment (MELiTA). MELiTA introduces an inter-modal evaluation process that shares partial artefacts (e.g., image or text) among phenotypically similar elites in order to find more coherent pairings (Figure 28). We test MELiTA on a bimodal test case for generating novel text and artworks for fictional game titles. For the text modality, two separate GPT-2 models were finetuned on a dataset of 72,000 Steam game titles and descriptions. The text can be mutated either partially or fully, and is characterized for the archive using topic modelling via the Latent Dirichlet Allocation (LDA) algorithm. For the image modality, MELiTA leverages Stable Diffusion (SD) to generate game cover arts which closely align with the title and description. Images are mutated using augmentation functions from the TorchVision software library, as well as using an image-to-image SD model, and are characterized by their colorfulness and complexity. This innovative approach enhances the creative co-evolutionary process, resulting in the discovery of fitter and more diverse

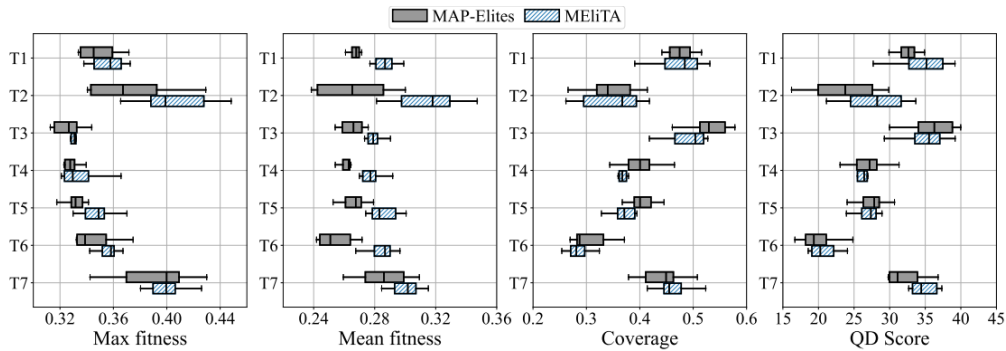


Figure 29. Metrics of the archives after 2000 selections in MAP-Elites and MELiTA. Box plots summarize values from 10 runs per title.

outcomes (i.e., higher in both quality and diversity).

9.1.2. Experimental results

Our experimental protocol revolves around three hypotheses for comparing MELiTA against vanilla MAP-Elites. We hypothesize that MELiTA will (H1) produce a higher quality and more diverse archive, (H2) produce said results more efficiently, and (H3) produce solutions which are more diverse to human consumers. This involved selecting 7 game titles of varying themes for comprehensive testing and conducting 10 evolutionary trials per title. Each trial generated 100 descriptions per title, and 4 images per description, with each run starting on the same seed set of 100 individuals. The performance was quantified using metrics such as mean and maximum fitness, coverage of the feature map, and the Quality-Diversity (QD) score, with the speed of discovery measured through the area under the curve (AUC). Additionally, orthogonal diversity metrics based on SBERT embeddings were implemented to assess the alignment of algorithmic outputs with human perceptions of diversity.

The experimental results demonstrated that MELiTA generally outperformed MAP-Elites in producing fitter individuals across most of the game titles, with statistically higher mean fitness in six out of seven cases and higher maximum fitness in three (Fig. 29). However, MAP-Elites achieved greater coverage in two of the game titles. Notably, MELiTA also displayed enhanced efficiency, achieving optimal solutions more rapidly than MAP-Elites as evidenced by the area under the curve (AUC) metrics. Despite this, the QD scores were comparable across both algorithms, indicating that while MELiTA often led to better-performing elites, it did not always lead to a broader exploration of the feature map. The findings from the orthogonal diversity metrics indicated that although MELiTA enhances the quality of solutions, it does not consistently exceed MAP-Elites in promoting diversity. These results are a partial validation of the hypotheses, confirming improvements in quality and efficiency but not fully supporting the expected increase in diversity as perceived by humans.

9.1.3. Conclusion

This work introduces MELiTA, an adaptation of the MAP-Elites algorithm to multimodal creative domains. Results show that MELiTA significantly enhances the quality of generated solutions compared to MAP-Elites, achieving higher mean fitness in six out of seven game titles and higher





maximum fitness in three titles. This indicates that MELiTA produces more coherent and optimized outputs, highlighting its potential for improving solution quality in various applications. We plan to explore using MELiTA in more complex generation tasks and across in tasks with more than two modalities to highlight the efficacy of our approach.

9.1.4. Relevant publications

- Zammit, M., Liapis, A., & Yannakakis, G. N. (2024, March). MAP-elites with transverse assessment for multimodal problems in creative domains. In Proceedings of the International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar) (pp. 401-417). Cham: Springer Nature Switzerland. Zenodo Link: <https://zenodo.org/records/11145009>

9.1.5. Relevance to AI4Media use cases and media industry applications

The MELiTA algorithm is a promising approach for enhancing the quality of generated solutions in any creative and multimodal domain, not only games (Use Case 5 - AI for Games). MELiTA can be integrated into use cases that involve creative content generation, such as (but not limited to) Use Case 6 (AI for Human Co-creation) where it can be used as a design assistant to generate music and associated artworks.

9.2. Large Language Models for Automated Game Generation

Contributing partner: UM

9.2.1. Introduction and methodology

This work represents the first steps taken to expand upon the MELiTA algorithm described in the previous section. In the ever-evolving landscape of video games, developers are frequently faced with the challenge of creating highly immersive gaming experiences that resonate with a wide range of players with diverse interests and personalities. Traditional game development processes often adhere to predefined visual assets and storylines, limiting the capacity for dynamic adaptation and player-driven customisation. Content generators for individual game facets are abundant, but the orchestration of multiple facets required for a cohesive outcome is a difficult and as yet unsolved problem [494], despite some notable work in the field [495]. The recent advances in large language models (LLMs) have endowed them with the ability to comprehend and interpret complex game concepts, narratives, and themes [496]. Their extensive training on vast textual corpora equips them with a rich understanding of language, enabling them to grasp the nuances and intricacies necessary to generate adaptations, whilst maintaining thematic coherence. Combined with the capabilities of text-to-image generative models, such as Stable Diffusion (SD) [497], LLMs can orchestrate a sophisticated content generation pipeline, enabling the dynamic re-theming of games by changing their narrative and visual assets. We hereby explore how LLMs may be used as a top-down pipeline for orchestration [494] to adapt and personalise gaming experiences.

To test the versatility of our proposed system, we developed CrawLLM, a dungeon crawler game using the Unity game engine⁹. The player navigates top-down mazes, battling enemies and collecting keys to unlock doors. The player's goal is to find the exit. Doors require varying numbers of keys, may be one-way, or hidden. Combat is card-based, where players select action cards using mana. Winning a combat round grants new cards for future encounters. Due to the time-intensive

⁹<https://unity.com/>



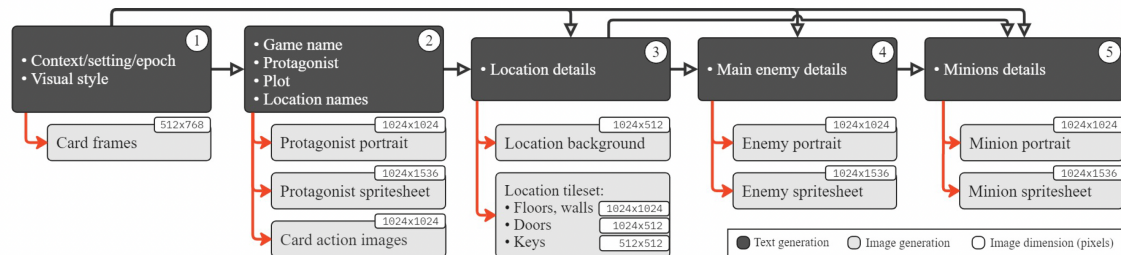


Figure 30. The text and image assets generation sequence. Pixel dimensions for images are also shown. Arrows delineate the inputs used to generate the respective assets. Outputs from block #1 were also used in all image prompts (arrows not shown).

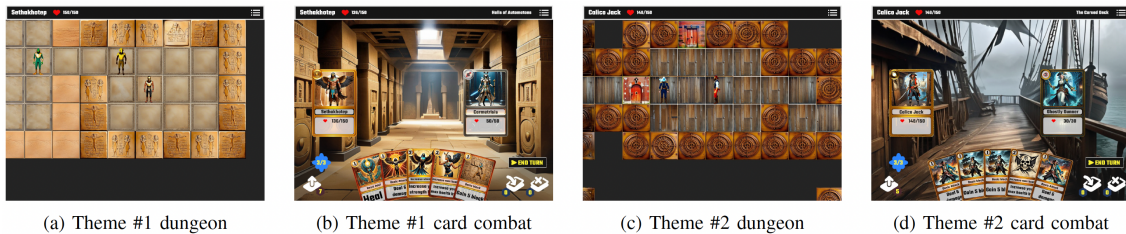


Figure 31. Screenshots showing the dungeon crawling (a and c) and card combat sections (b and d) of CrawlLLM with an ancient Egypt (Theme #1) and a pirate theme (Theme #2).

nature of sequentially generating multiple assets, we present a playable demonstration featuring 20 pre-generated themes constituting a complete generative run, rather than a curated selection. The dungeon layout and solvable door/key puzzles are dynamically generated using grammars [498] built around the unpublished cyclic dungeon generation method from Unexplored (Ludomotion, 2017). The dungeon is divided into cycles of rooms, each representing a narrative location with its own door/key puzzle. Rooms may be replaced by sub-cycles, to increase complexity while retaining solvability.

The Mixtral 8x7B LLM [499] was used to generate new themes. Figure 30 shows the iterative sequence of narrative elements, names, and descriptions which were generated. In addition to the theme, main plot, and protagonist name and descriptions, the names of locations for the dungeon cycles are generated. In each location, a main antagonist and subsequently two minions are generated to embellish the combat encounters and add difficulty. Furthermore, the LLM produces the prompts required to create visual assets via a Stable Diffusion XL (SDXL) [500] text-to-image generative model. Figure 31 shows examples of the game’s visuals with different themes applied.

The game structure posed several challenges for content generation. Tile set images needed visual coherence yet distinction across dungeon locations. Creating subtle secret door tiles from regular walls was difficult. This was accomplished by supplementing SDXL with ControlNET [501] models, providing rough outline sketches alongside text prompts to guide image generation. Card frames were similarly constrained by demarcating areas for text and images. To animate the characters’ movement, ControlNET was provided with manually generated templates of poses and depth maps for each required frame. This ensured a precise positioning of the characters within the spritesheets, which were generated as one image for consistency. Card images, player portraits, and background images used a plain SDXL text-to-image pipeline.





9.2.2. Experimental results and discussion

The pipeline presented in this early proof of concept features no manual post-processing or curation of the generated assets, highlighting that these are fit for use despite some imperfections. However, there are a number of issues with the current process. Firstly, the pipeline and the LLM prompts have to be meticulously crafted and tuned to match the game mechanics and individual assets required. Character consistency between different SD generations is also challenging, so multiple images of the same characters often result in notable visual differences. Furthermore, enemies were limited to a humanoid form since bipedal pose templates were used to generate spritesheets. Another limitation is that pretrained open source SD models lack a transparency channel, requiring automated background removal which often omits peripheral details.

Following this work, we intend to quantify coherence and diversity across the range of generated assets, and develop an automated game asset generation pipeline which may be approached from a quality-diversity angle [502]. This study is thus intended as a first step towards LLMs acting as a central coordination layer for automated game generation, streamlining the game development process while maintaining cohesive aesthetics.

9.2.3. Conclusion

The main contributions of this work are to:

- Introduce a novel approach to automated game generation using LLMs as a central coordination layer.
- Demonstrate a demo of CrawLLM, a dungeon crawler game with card-based combat, along with 20 pre-generated themes showcasing the feasibility and adaptability of the system.

9.2.4. Relevant publications

- Zammit, M., Liapis, A., & Yannakakis, G. N. (2024). CrawLLM: Theming Games with Large Language Models. Proceedings of the IEEE Conference on Games (IEEE CoG) 2024. Zenodo Link: <https://zenodo.org/records/13330059>.

9.2.5. Relevance to AI4media use cases and media industry applications

Whilst this work directly relates to Use Case 5 (AI for Games), it can also be applied to other use cases and segments of the media industry which involve the generation of content, such as Use Case 6 (AI for Human Co-creation) where it can assist in creative workflows and provide alternative themes and ideas to the designer.

9.3. Quality Diversity in Dynamic Environments

Contributing partner: UM

9.3.1. Introduction and methodology

Moving away from automated game generation using LLMs, in this work we focus on solving optimisation problems which are not fixed, but rather change in different ways over time. These Dynamic Optimisation Problems (DOPs) [503,504] are of high interest to the research community, as they mirror real-world problems where time affects performance of solutions [505,506], constraints [507–509], or even the bounds of the solution space available to the solving algorithm [510,511]. DOPs are





distinct from noisy optimisation problems, where the problem itself is static over time but there is noise in either the evaluation of a solution or in its fitness assignment. While this latter family of optimisation problems also resembles real-world problems, we are more interested in the former group, as time-dependent environments can also be further extended to handle noisy evaluations.

When searching for a solution in a dynamic environment, changes often occur in the fitness landscape. Thus, it is important to be able to detect and adapt to such changes quickly, otherwise previously high-performing solutions may become outdated and hinder the search process [503]. Applying to DOPs evolutionary algorithms (EAs) [512] and particle swarm optimisation (PSO) [513] helps diversify the population, which improves detection of changes in the environment and adaptation. However, to our knowledge, prior applications of such algorithms relied on specific diversification techniques, and did not consider changes in the environment that would alter the measure of diversity itself. On the other hand, quality-diversity (QD) algorithms are a family of EAs that explicitly takes into account the behaviour of solutions to keep a diverse population [25]. Additionally, prior work on QD applied to noisy domains [514] and with changing featuremaps [515, 516] showed promising results. Therefore, we believe QD algorithms provide a solid foundation to solve DOPs, with some critical adaptations.

In this work, we propose a framework to adapt existing QD algorithms to solve DOPs, improving their performance over their static counterpart. Our Dynamic Quality-Diversity (D-QD) can search in environments that change at an unknown frequency and with unknown severity, on either the fitness landscape or the behavioural mapping. To address the challenges of DOPs, our approach involves adapting QD algorithms to better handle environmental shifts that impact solution effectiveness over time. Our D-QD algorithm utilizes a dual strategy of re-evaluation and real-time adaptation: it periodically re-evaluates stored solutions to ensure their fitness remains optimal under the new conditions and adapts its search strategy to focus on areas of the solution space that are likely to be affected by the changes. By doing so, D-QD maintains a diversified portfolio of high-quality solutions that are robust to changes in the problem landscape. This methodology leverages the intrinsic diversity-maintaining properties of QD algorithms while enhancing their flexibility and responsiveness, making them well-suited for the dynamic nature of real-world problems where conditions evolve unpredictably.

9.3.2. Experimental results

To evaluate the effectiveness of the D-QD framework, we conducted a series of experiments using two well-known QD algorithms, MAP-Elites and Covariance Matrix Adaptation MAP-Elites (CMA-ME), across two dynamic environments: a modified sphere function and a dynamic lunar lander scenario. These environments were chosen to represent both abstract mathematical challenges and more complex, real-world dynamic tasks. In each environment, the algorithms were tasked with maintaining an archive of diverse, high-quality solutions under conditions where environmental variables were changed unpredictably at regular intervals. The experiments measured the algorithms' ability to detect environmental shifts (an example of which can be seen in Fig. 32), adapt the population of solutions, and maintain or improve the overall quality and diversity of the archive over time.

The results demonstrate that the D-QD framework significantly outperforms traditional QD approaches in dynamic environments. In the dynamic sphere environment, D-QD adapts quickly to changes, maintaining a high diversity score and effectively updating solutions in response to environmental shifts. In the more complex dynamic lunar lander environment, the framework shows a robust ability to handle real-world complexities, with improved performance in adapting to significant shifts in environmental conditions. These results were quantified using metrics such as the percentage of updated solutions in the archive, the mean squared error between the predicted and



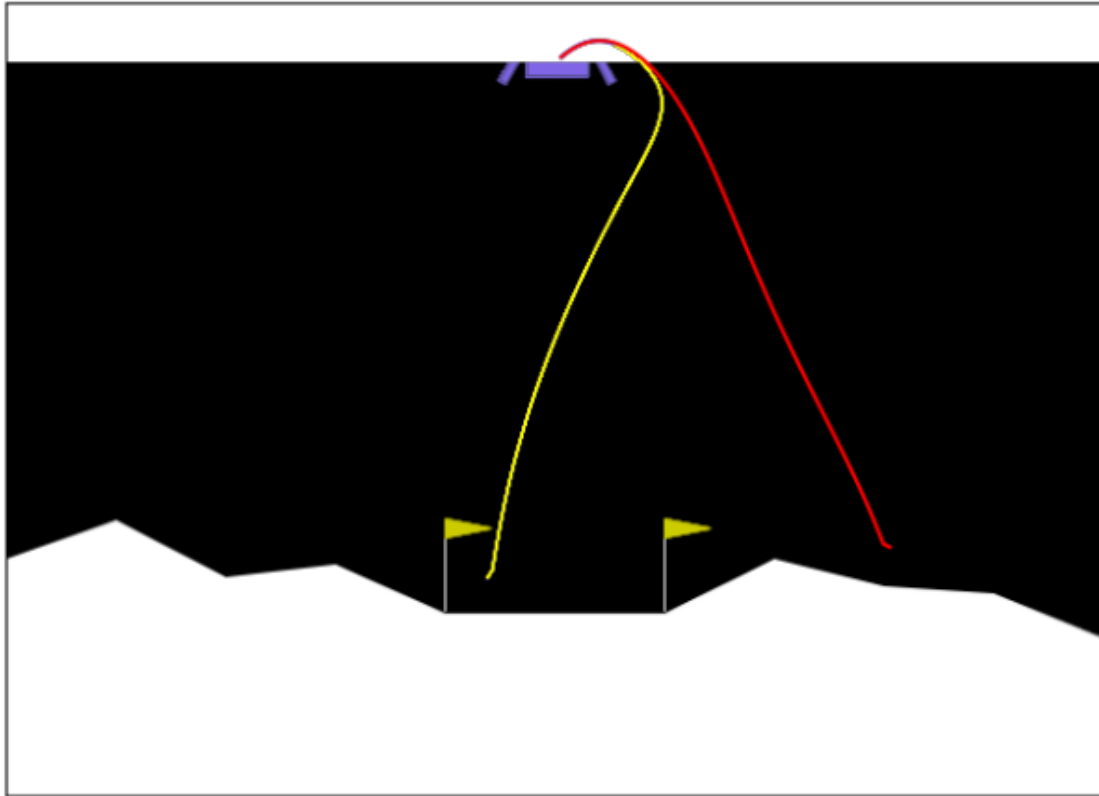


Figure 32. The complete trajectory of the same lander in the dynamic Lunar Lander environment, before an environment shift (yellow) and after (red).

actual performance of solutions, and the overall diversity of the solution set. The D-QD algorithms not only preserved high-quality solutions but also demonstrated superior adaptability, making them particularly suitable for dynamic optimization problems where environmental conditions are continually evolving.

9.3.3. Conclusion

The main contributions of this work are to:

- Introduce a novel D-QD framework adapted for DOPs, enhancing traditional Quality-Diversity algorithms to handle environmental variability effectively. This adaptation allows for continuous adaptation to changing conditions, maintaining the relevance and efficacy of solutions over time.
- Demonstrate through our experiments that D-QD algorithms can significantly outperform static QD algorithms in two separate dynamic environments, showcasing their applicability and robustness across different types of dynamic challenges.
- Establish a set of performance metrics specifically designed to evaluate the effectiveness of QD algorithms in dynamic environments, providing a comprehensive assessment framework





for future research in dynamic optimization using QD.

These contributions mark a significant step forward in the field of evolutionary computation, particularly in addressing real-world problems where conditions are not static but evolve unpredictably.

9.3.4. Relevant publications

- Gallotta, R., Liapis, A., & Yannakakis, G. N. (2024). Dynamic quality-diversity search. Zenodo Link: <https://zenodo.org/records/13330034>

9.3.5. Relevant software/datasets/other outcomes

- The code for this paper is available at <https://github.com/gallorob/dynamic-quality-diversity>.

9.3.6. Relevance to AI4media use cases and media industry applications

This work is related to any media application where content or behaviors need to be generated/evaluated in dynamic environments. As with previous sections, this work is directly relevant to Use Case 5 (AI for Games), and can also be applicable to Use Case 6 (AI for Human Co-creation) where it can be used to generate content that adapts to user preferences or changing conditions, as well as potentially to Use Case 2 (AI for Media Production) where it can be used to optimize media production workflows in response to changing requirements or constraints.

9.4. Quality-Diversity Search for Constrained Structure Design

Contributing partner: UM

9.4.1. Introduction and methodology

Our final contribution focuses on the use of QD search for optimization of outputs following a set of designer-specified constraints. A frequently encountered problem within the Architecture, Engineering and Construction (AEC) industry is the design of efficient shell structures. Such structures – in which architectural form is inextricably linked to structural behaviour and performance – require a close collaboration between architect, structural engineer and other AEC consultants to find a shape that is acceptable to all parties and their separate demands. Historically, many of these shells were conceived as form-active structures, and thus their final shape was ascertained through form-finding techniques—first physically [517], [518], then digitally [519]. These digital simulations are remarkably flexible, and able to incorporate a range of physical input constraints, such as ensuring structural efficiency (e.g. axial loading only) or geometric rationalisation (e.g. avoiding panel warp). However, for a single set of inputs (materials, loading and support conditions) form-finding algorithms only generate a singular solution, rather than a range of potential solutions. As such, it can be difficult for the designer to evaluate potential trade-offs between the desired criteria and propose new weightings of them. We see similar limitations in evolutionary computation (EC) algorithms, such as Divide-and-Conquer or Hill-Climbing. These algorithms are Single Objective Optimisation (SOO) approaches, and as such they too suffer from a difficulty in weighting and combining output values to achieve a well-balanced fitness evaluation for the variants. More sophisticated EC approaches include Multi Objective Optimisation (MOO) or Pareto optimisation algorithms, such as SPEA-2 and HypE. These MOO algorithms have been implemented in AEC digital design toolkits for several years now [520], [521], and allow substantially more detailed exploration of the solution space.



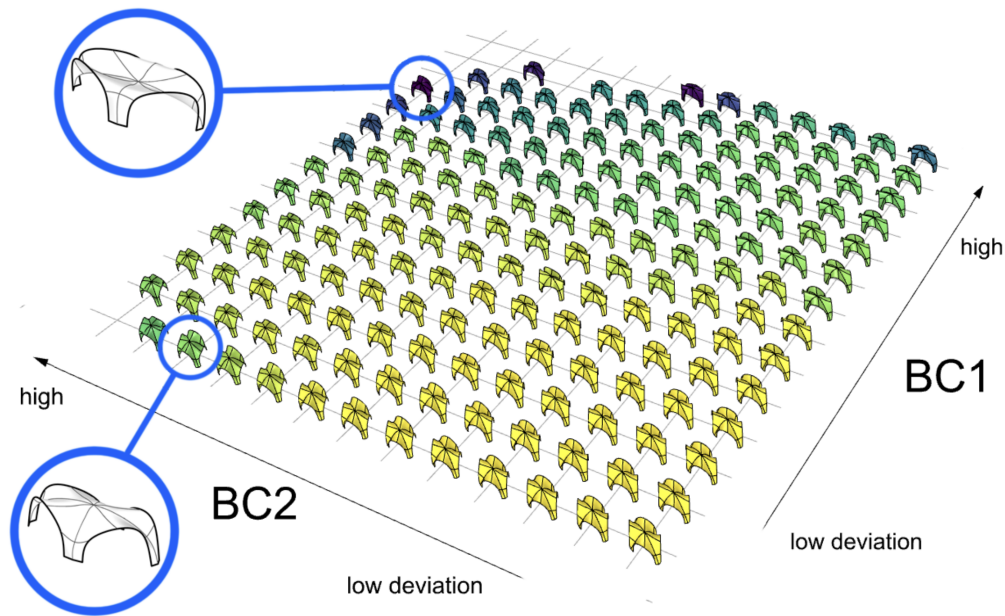


Figure 33. Visualisation of all feasible solutions found in a single run of the FI-MAP-Elites algorithm on the Complex Test Case. Meshes in yellow have a higher fitness (i.e. lower elastic energy). All feasible designs are found in the first 13 by 14 cells (BC1 by BC2) of the feature map; empty cells beyond this range are omitted.

This work presents a constrained variant of the MAP-Elites algorithm (described in more detail in Section 9.1) for QD search called the Feasible-Infeasible Map-Elites (FI-Map-Elites) algorithm, and tests it in a set of shell structure optimization tasks ranging from simple to complex. While shell structure optimization has received attention in the past through evolutionary approaches [522], here we propose a QD method that explicitly produces a large archive of potential solutions and visualizes their trade-offs. The FI-MAP-Elites algorithm, leverages two archives evolving in parallel: one archive of solutions that fail a constraint on maximum allowed displacement, and another archive of solutions that satisfy this constraint and search for optimal elastic energy of the structure. The algorithm has been integrated into Grasshopper¹⁰ and allows engineers to specify the structure of possible solutions and the dimensions towards which FI-MAP-Elites should explore. Comparisons against a baseline SOO algorithm available in Grasshopper show that the FI-Map-Elites algorithm produces far more diverse solutions, while SOO typically converges towards a single type of solution. Therefore, this approach argues that QD search algorithms that illuminate the solution space and show trade-offs between different designs will help engineers interface with the subjectivity of the AEC design process.

9.4.2. Experimental results

To evaluate the performance of the proposed FI-MAP-Elites algorithm, we conduct a series of experiments using parametric models of shell structures defined in Rhino’s visual programming interface, Grasshopper. These models are tested under various structural constraints and loading conditions to simulate real-world engineering scenarios. We use a population of 100 randomly initialized individuals, applying mutation operations to explore the design space, and iterating

¹⁰Tool can be downloaded from: <https://www.grasshopper3d.com/>





through 50,000 evaluations to evolve the population towards higher quality and diversity. We then test them across two simple test cases and one complex test case. In the simple test cases, a square SubD surface with 9 patches and 16 control points must be optimized. The algorithm varies the height of each control point to find optimal shapes under two different support constraints. The first simple test constrained all translations at the four corner points, while the second allowed free translation in the x direction. For the complex test case, a more detailed four-arch vault shell must be optimized. This model, comprising 16 patches and 29 control points, requires optimizing the structure under self-weight conditions with a displacement threshold of 6 cm.

The results in our simple test case demonstrated that the FI-MAP-Elites algorithm achieved higher feasible coverage and QD-score compared to the SOO approach, with the fittest solutions exhibiting lower elastic energy and displacement. For the complex test case, the results show that FI-MAP-Elites outperformed the SOO approach by generating a diverse set of feasible solutions, with the best solutions having significantly lower elastic energy and displacement. The feasible archive (visualized in Figure 33) indicates that FI-MAP-Elites effectively explored the design space, uncovering a wide range of structurally sound and geometrically varied shell structures. These results indicate that the FI-MAP-Elites algorithm provides a robust framework for exploring the design space of shell structures, offering engineers a comprehensive view of possible design alternatives and their trade-offs. Future work will focus on further refining the algorithm, expanding its application to other types of structural engineering problems, and integrating it into professional workflows through the development of a user-friendly plugin for Rhino.

9.4.3. Conclusion

The main contributions of this work are to:

- Introduce a constrained variant of the MAP-Elites algorithm, called FI-MAP-Elites, tailored for quality-diversity search with specific feasibility requirements.
- Implement the proposed algorithm within the Rhino/Grasshopper environment, enabling seamless integration with existing parametric modeling and structural analysis tools.
- Demonstrate the effectiveness of FI-MAP-Elites through extensive experiments on both simple and complex shell structure test cases, showing that the algorithm can generate a diverse set of high-quality solutions that satisfy designer-specified structural constraints.

9.4.4. Relevant publications

- Sfikas, K., Liapis, A., Hilmersson, J., Dudley, J., Tibuzzi, E., & Yannakakis, G. N. (2023, October). Design space exploration of shell structures using quality diversity algorithms. In Proceedings of IASS Annual Symposia (Vol. 2023, No. 8, pp. 1-12). International Association for Shell and Spatial Structures (IASS). Zenodo Link: <https://zenodo.org/records/7974309>.

9.4.5. Relevance to AI4media use cases and media industry applications

This work is directly relevant to any sector of the media industry or use case where high quality and diverse content needs to be generated within a specific feasibility space. This is beneficial to Use Case 5 (AI for Games) for generating game content (e.g. in-game structures), as well as any other use case that may require constrained media generation.





10. Learning to count (Task 3.7)

Contributing partners: CNR

“Learning to Count” is a task having to do with supervised learning approaches for training estimators of quantities. There are two classes of problems that are being addressed in this task, and that may be usefully viewed as forming two different subtasks, i.e.,

- “Learning to quantify” (LQ – a.k.a. *quantification*). This subtask is concerned with training unbiased estimators of class prevalence via supervised learning, i.e., learning to estimate, given a sample of objects, the percentage of objects that belong to a given class. This task originates with the observation that “CC”, the trivial method of obtaining class prevalence estimates, is often a biased estimator, and thus delivers suboptimal quantification accuracy. This bias is particularly strong when the data exhibits *dataset shift*, i.e., when the joint distribution of the dependent and the independent variables is not the same in the training data and in the unlabelled data for which predictions must be issued. Quantification is important for several applications, e.g., gauging the collective satisfaction for a certain product from textual comments, establishing the popularity of a given political candidate from blog posts, predicting the amount of consensus for a given governmental policy from tweets, or predicting the amount of readers who will find a product review helpful.
- “Learning to count objects”. This subtask has to do with using machine learning approaches in order to train estimators of the number of objects (which may be inanimate objects, such as cars, but may also be animate objects, such as people or animals) in visual media, such as still images or video frames. Example applications of these techniques are, e.g., counting the number of cars in a video frame (in order to estimate traffic volume or car park occupancy), or counting the number of people in a still image (say, in order to estimate the amount of people taking part in a rally).

10.1. Measuring fairness under unawareness of sensitive attributes: A quantification-based approach

Contributing partner: CNR

10.1.1. Introduction

Algorithms and models are increasingly deployed to inform decisions about people, inevitably affecting their lives. As a consequence, those in charge of developing these models must carefully evaluate their impact on different groups of people and favour *group fairness*, that is, ensure that groups determined by sensitive demographic attributes, such as race or sex, are not treated unjustly. To achieve this goal, the availability (*awareness*) of these demographic attributes to those evaluating the impact of these models is fundamental. Unfortunately, collecting and storing these attributes is often in conflict with industry practices and legislation on data minimisation and privacy. For this reason, it can be hard to measure the group fairness of trained models, even from within the companies developing them. In this work, we tackle the problem of measuring *group fairness under unawareness* of sensitive attributes, by using techniques from *quantification*, a supervised learning task concerned with directly providing group-level prevalence estimates (rather than individual-level class labels). We show that quantification approaches are particularly suited to tackle the fairness-under-unawareness problem, as they are robust to inevitable distribution shifts while at the same time decoupling the (desirable) objective of measuring group fairness from





the (undesirable) side effect of allowing the inference of sensitive attributes of individuals. More in detail, we show that fairness under unawareness can be cast as a quantification problem and solved with proven methods from the quantification literature. We show that these methods outperform previous approaches to measure demographic parity in five experimental protocols, corresponding to important challenges that complicate the estimation of classifier fairness under unawareness.

10.1.2. Methodology

We assume the existence, in the operational setup, of three separate sets of data points:

- A *training set* \mathcal{D}_1 for h , $\mathcal{D}_1 = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$, typically of large size, where h is the classifier whose fairness we want to measure. Given the difficulties inherent in demographic data procurement mentioned in the introduction, we assume that the sensitive attribute S is not part of the vectorial representation X .
- A small *auxiliary set* $\mathcal{D}_2 = \{(\mathbf{x}_i, s_i) \mid \mathbf{x}_i \in \mathcal{X}, s_i \in \mathcal{S}\}$, containing demographic data, employed to train quantifiers for the sensitive attribute.
- A set $\mathcal{D}_3 = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathcal{X}\}$ of unlabelled data points, which are the data to which classifier h is to be applied, representing the deployment conditions. Alternatively, \mathcal{D}_3 could also be a labelled held-out test set available at a company, if it has acted proactively rather than reactively, for pre-deployment audits. In our experiments we use labelled data and call \mathcal{D}_3 the *test set*, on which the fairness of the classifier h should be measured.

In other words, this simulates a scenario in which we have training data for training the classifier whose fairness needs to be evaluated, test data on which the classifier should behave fairly, and auxiliary labelled data that we can use for training a model to be used for evaluating the fairness of the previously mentioned classifier.

It is worth re-emphasizing that, from the perspective of the estimation task at hand, i.e., estimating the fairness of the classifier h , \mathcal{D}_2 represents the quantifier's training set, while \mathcal{D}_3 is its test set.

Proposition 1. *Observational measures of algorithmic fairness can be computed, under unawareness of sensitive attributes, by estimating the prevalence of the sensitive attribute in specific subsets of the test set.*

Proof. We prove this statement for True Positive Rate Disparity (TPRD), defined as:

$$\text{True Positive Rate Disparity: } \delta_h^{S, \text{TPRD}} = \Pr(\hat{Y} = \oplus \mid S = 1, Y = \oplus) - \Pr(\hat{Y} = \oplus \mid S = 0, Y = \oplus)$$

Both terms in the above equation can be written as

$$\begin{aligned} \Pr(\hat{Y} = \oplus \mid S = s, Y = \oplus) &= \frac{\Pr(Y = \oplus, \hat{Y} = \oplus, S = s)}{\Pr(Y = \oplus, S = s)} \\ &= \underbrace{\frac{\Pr(S = s \mid Y = \oplus, \hat{Y} = \oplus)}{\Pr(S = s \mid Y = \oplus)}}_{\text{obtained from prevalence estimator}} \cdot \underbrace{\frac{\Pr(Y = \oplus, \hat{Y} = \oplus)}{\Pr(Y = \oplus)}}_{\text{known quantity}} \end{aligned}$$

In other words, TPRD can be calculated by estimating the prevalence of the sensitive attribute among the positives and the true positives in \mathcal{D}_3 . Analogous results can be proven for other measures of observational fairness, under the assumption that Y and \hat{Y} are known. \square





Remark 1. *This proposition is important for two reasons. First, it shows that inference of sensitive attributes at the individual level is not necessary to measure fairness under unawareness; rather, prevalence estimates in given subsets are sufficient. Second, it suggests that methods directly targeting prevalence estimates (i.e., quantifiers) are especially suited in this setting.*

Notice that, for the purposes of a fairness audit, it is common to assume that the ground truth variable Y is available in \mathcal{D}_3 . In the banking scenario, this is only partially realistic, as the outcomes for the accepted applicants are eventually observed, but the outcomes for the rejected applicants remain unknown, leaving us with a problem of sample selection bias. This is an instance of a general estimation problem, common to all fairness criteria that require knowledge of the ground truth variable Y , such as TPRD. This represents an open research problem which is beyond the scope of this work and demands additional caution in the estimation and interpretation of these fairness measures.

We focus on a detailed study of demographic disparity (DD). This allows us to thoroughly characterize and discuss DD estimators while avoiding the pitfalls and complexity of uncertain ground truth information. We leave additional measures of observational fairness for future work.

We write DD as

$$\delta_h^S = \Pr(\hat{Y} = \oplus | S = 1) - \Pr(\hat{Y} = \oplus | S = 0) = \mu(1) - \mu(0), \quad (12)$$

where

$$\mu(s) = \Pr(\hat{Y} = \oplus | S = s) \quad (13)$$

is the acceptance rate of individuals in the group $S = s$. To estimate the demographic disparity of a classifier $h(\mathbf{x})$ in the test set \mathcal{D}_3 , we can use any quantification approach. Applying Bayes' theorem to Equation 13, we obtain

$$\begin{aligned} \mu(s) &= p_{\mathcal{D}_3}(\oplus | s) \\ &= p_{\mathcal{D}_3^\oplus}(s) \frac{p_{\mathcal{D}_3}(\oplus)}{p_{\mathcal{D}_3}(s)}, \end{aligned} \quad (14)$$

where we use $p_{\mathcal{D}_3}(\oplus)$ as a shorthand of $p_{\mathcal{D}_3}(h(\mathbf{x}) = \oplus)$, and where we have defined

$$\begin{aligned} \mathcal{D}_3^\oplus &= \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \oplus\} \\ \mathcal{D}_3^\ominus &= \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \ominus\}. \end{aligned}$$

Since $p_{\mathcal{D}_3}(\oplus)$ is known (it is the fraction of items in \mathcal{D}_3 that have been assigned class \oplus by the classifier h), in order to compute $\mu(s)$ through Equation 14, for $s \in \{0, 1\}$, we only need to estimate the prevalence values $\hat{p}_{\mathcal{D}_3^\oplus}(s)$ and $\hat{p}_{\mathcal{D}_3^\ominus}(s)$; the latter is needed to estimate the denominator of Equation 14, i.e., the prevalence $p_{\mathcal{D}_3}(s)$ of the sensitive attribute value s in the entire test set \mathcal{D}_3 , since

$$p_{\mathcal{D}_3}(s) = p_{\mathcal{D}_3^\oplus}(s) \cdot p_{\mathcal{D}_3}(\oplus) + p_{\mathcal{D}_3^\ominus}(s) \cdot p_{\mathcal{D}_3}(\ominus). \quad (15)$$

In order to compute $p_{\mathcal{D}_3^\oplus}(s)$ and $p_{\mathcal{D}_3^\ominus}(s)$ we can use a quantification-based approach, which can be easily integrated into existing machine learning workflows, as summarized by the method below.

Method. Quantification-Based Estimate of Demographic Disparity.

1. The classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$ is trained on \mathcal{D}_1 and ready for deployment, e.g., to estimate the creditworthiness of individuals. The assumption that, at this training stage, we are unaware of the sensitive attribute S is due to the inherent difficulties in demographic data procurement.





2. We use the classifier h to classify the auxiliary set \mathcal{D}_2 , thus inducing a partition of \mathcal{D}_2 into $\mathcal{D}_2^{\oplus} = \{(\mathbf{x}_i, s_i) \in \mathcal{D}_2 \mid h(\mathbf{x}) = \oplus\}$ and $\mathcal{D}_2^{\ominus} = \{(\mathbf{x}_i, s_i) \in \mathcal{D}_2 \mid h(\mathbf{x}) = \ominus\}$.
3. We use \mathcal{D}_2^{\oplus} as the training set for the quantifier $q_{\oplus}(s)$, whose task will be to estimate the prevalence of value s (e.g., African-American applicants) on sets of data points labelled with class \oplus (e.g., creditworthy applicants). Likewise, we use \mathcal{D}_2^{\ominus} as the training set for a quantifier $q_{\ominus}(s)$ whose task will be to estimate the prevalence of s on sets of data points labelled with \ominus . Intuitively, separate quantifiers specialized on different subpopulations (of positively and negatively classified individuals) should perform better than a single quantifier. Our ablation study supports this hypothesis.
4. The classifier h is deployed, classifying the test set \mathcal{D}_3 , thus inducing a partition of \mathcal{D}_3 into positive $\mathcal{D}_3^{\oplus} = \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \oplus\}$ and negative $\mathcal{D}_3^{\ominus} = \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \ominus\}$.
5. We apply the quantifier q_{\oplus} to \mathcal{D}_3^{\oplus} to obtain an estimate $\hat{p}_{\mathcal{D}_3^{\oplus}}^{q_{\oplus}}(s)$ of the prevalence of s in \mathcal{D}_3^{\oplus} , and we apply q_{\ominus} to \mathcal{D}_3^{\ominus} to obtain an estimate $\hat{p}_{\mathcal{D}_3^{\ominus}}^{q_{\ominus}}(s)$ of the prevalence of s in \mathcal{D}_3^{\ominus} . Recall that $\hat{p}_{\sigma}^q(s)$ denotes the prevalence of an attribute value s in a set σ as estimated via quantification method q .
6. To avoid numerical instability in the denominator of Equation 17 below, we apply Laplace smoothing to the estimated prevalence values $\hat{p}_{\mathcal{D}_3^{\oplus}}^{q_{\oplus}}(s)$ and $\hat{p}_{\mathcal{D}_3^{\ominus}}^{q_{\ominus}}(s)$. We use the variant that uses known incidence rates, using \mathcal{D}_2^{\ominus} and \mathcal{D}_2^{\oplus} as the control populations, and assume a pseudocount $\alpha = 1/2$. We thus compute the smoothed estimator

$$\begin{aligned} \tilde{p}_{\mathcal{D}_3^{\oplus}}^{q_{\oplus}}(s) &= \frac{\hat{p}_{\mathcal{D}_3^{\oplus}}^{q_{\oplus}}(s) \cdot |\mathcal{D}_3^{\oplus}| + p_{\mathcal{D}_2^{\oplus}}(s) \cdot \alpha \cdot |\mathcal{Y}|}{|\mathcal{D}_3^{\oplus}| + \alpha \cdot |\mathcal{Y}|} \\ &= \frac{\hat{p}_{\mathcal{D}_3^{\oplus}}^{q_{\oplus}}(s) \cdot |\mathcal{D}_3^{\oplus}| + p_{\mathcal{D}_2^{\oplus}}(s)}{|\mathcal{D}_3^{\oplus}| + 1} \end{aligned}$$

and analogously for $\tilde{p}_{\mathcal{D}_3^{\ominus}}^{q_{\ominus}}(s)$.

7. Finally, we estimate the demographic disparity of h , defined in Equation 12, as

$$\hat{\delta}_h^S = \hat{\mu}(1) - \hat{\mu}(0) \quad (16)$$

where, as from Equations 14 and 15,

$$\hat{\mu}(s) = \tilde{p}_{\mathcal{D}_3^{\oplus}}^{q_{\oplus}}(s) \cdot \frac{p_{\mathcal{D}_3}(\oplus)}{\tilde{p}_{\mathcal{D}_3^{\oplus}}^{q_{\oplus}}(s) \cdot p_{\mathcal{D}_3}(\oplus) + \tilde{p}_{\mathcal{D}_3^{\ominus}}^{q_{\ominus}}(s) \cdot p_{\mathcal{D}_3}(\ominus)} \quad (17)$$

Thorough experiments that validate the described method are presented in [523].

10.1.3. Conclusion

Measuring the differential impact of models on groups of individuals is important to understand their effects in the real world and their tendency to encode and reinforce divisions and privilege across sensitive attributes. Unfortunately, in practice, demographic attributes are often not available. In this work, we have taken the perspective of responsible practitioners, interested in internal fairness audits of production models. We have proposed a novel approach to measure group fairness under





unawareness of sensitive attributes, utilizing methods from the quantification literature. These methods are specifically designed for group-level prevalence estimation rather than individual-level classification. Since practitioners who try to measure fairness under unawareness are precisely interested in prevalence estimates of sensitive attributes (Proposition 1), it is useful for the fairness and quantification communities to exchange lessons.

We have studied the problem of estimating a classifier’s fairness under unawareness of sensitive attributes, with access to a disjoint auxiliary set of data for which demographic information is available. We have shown how this can be cast as a quantification problem, and solved with established approaches of proven consistency. We have conducted a detailed empirical evaluation of different methods and their properties focused on demographic parity. Drawing from the algorithmic fairness literature, we have identified five important factors for this problem, associating each of them with a formal evaluation protocol. We have tested several quantification-based approaches, which, under realistic assumptions for an internal fairness audit, outperform previously proposed estimators in the fairness literature. We have discussed their benefits and limitations, including the unbiasedness guarantees of some methods, and the potential for misuse at an individual level.

Future work may require a deeper study of the relation between classification and quantification performance and the extent to which these two objectives can be decoupled. It would be interesting to explicitly target decoupling through learners aimed at maximizing quantification performance subject to a low classification performance constraint. Ideally, decoupling should provide precise privacy guarantees to individuals while allowing for precise group-level estimates. Another important avenue for future work is the study of confidence intervals for fairness estimates provided by quantification methods. A reliable indication of confidence for estimates of group fairness may be invaluable for a practitioner arguing for resources and attention to the disparate effects of a model on different populations. Finally, the estimators presented in this work may be plugged into optimization procedures aimed at improving, rather than measuring, algorithmic fairness. Mixed loss functions, jointly optimizing accuracy and fairness can be optimized, even under unawareness of sensitive attributes, with our methods providing fairness estimates at each iteration. It will be interesting to evaluate fairness estimators in this broader context and extend them, e.g., to ranking problems and counterfactual settings.

10.1.4. Relevant publications

- Alessandro Fabris, Andrea Esuli, Alejandro Moreo, and Fabrizio Sebastiani. Measuring fairness under unawareness of sensitive attributes: A quantification-based approach. **Journal of Artificial Intelligence Research** 76:1117–1180, 2023. [523] <https://zenodo.org/records/7090075>

10.1.5. Relevant software/datasets/other outcomes

- The code needed to reproduce all our experiments is publicly available in open-source mode at <https://github.com/alessandro-fabris/ql4facct>.

10.2. Binary quantification and dataset shift: An experimental investigation

Contributing partner: CNR





10.2.1. Introduction

Quantification (variously called *learning to quantify*, or *class prior estimation*, or *class distribution estimation* – see [524, 525] for overviews) is a supervised learning task concerned with estimating the *prevalence values* (or *relative frequencies*, or *prior probabilities*) of the classes in a sample of unlabelled datapoints, using a predictive model (the *quantifier*) trained on labelled datapoints. A common trait of all the applications of quantification is that all of them emerge from the need to monitor evolving class distributions, i.e., situations in which the class distribution of the unlabelled data may differ from the one of the training data. In other words, these situations are characterised by a type of *dataset shift* [526, 527], i.e., the phenomenon according to which, in a supervised learning context, the training data and the unlabelled data are not IID. Dataset shift comes in different flavours; the ones that have mostly been discussed in the literature are (i) *prior probability shift*, which has to do with changes in the class prevalence values; (ii) *covariate shift*, which concerns changes in the distribution of the covariates (i.e., features); and (iii) *concept shift*, which has to do with changes in the functional relationship between covariates and classes. We provide more formal definitions of dataset shift and its subtypes in the sections to come.

Since quantification aims at estimating class prevalence, most experimental evaluations of quantification systems (see, e.g., [528–536]) have focused on situations characterised by prior probability shift, while the other two types of shift mentioned above have not received comparable attention. A question then naturally arises: *How do existing quantification methods fare when confronted with types of dataset shift other than prior probability shift?*

This work offers a systematic exploration of the performance of existing quantification methods under different types of dataset shift. To this aim we first propose a fine-grained taxonomy of dataset shift types; in particular, we pay special attention to the case of covariate shift, and identify variants of it (mostly having to do with additional changes in the priors) that we contend to be of special relevance in quantification endeavours, and that are understudied. We then follow an empirical approach, devising specific experimental protocols for simulating all the types of dataset shift that we have identified, at various degrees of intensity and in a tightly controlled manner. Using the experimental setups generated by means of these protocols, we then test a number of existing quantification methods; here, the ultimate goal we pursue is to better understand the relative merits and limitations of existing quantification algorithms, to understand the conditions under which they tend to perform well, and to identify the situations in which they instead tend to generate unreliable predictions.

10.2.2. Methodology

One of the main reasons why we study quantification is the fact that most scenarios in which estimating class prevalence values via supervised learning is of interest, *violate the IID assumption*, i.e., the fundamental assumption (that most machine learning endeavours are based on) according to which the labelled datapoints used for training and the unlabelled datapoints we want to issue predictions for, are assumed to be drawn independently and identically from the same (unknown) distribution.¹¹ If the IID assumption were not violated, the supervised class prevalence estimation problem would admit a trivial solution, consisting of returning, as the estimated prevalence \hat{p}_σ^g for any sample σ of unlabelled datapoints, the true prevalence p_L that characterises the training set

¹¹For example, we might be interested in monitoring through time the degree of support for a certain politician by estimating the prevalence values of classes “Positive” and “Negative” in tweets that express opinions about this politician (this is an instance of *sentiment quantification* [534]). The very fact that we want to monitor these prevalence values through time is an implicit assumption that these prevalence values may vary, i.e., may take values different from the prevalence values that these classes had in the training data. In other words, it is an implicit assumption that we may be in the presence of some form of dataset shift.





	$P(X)$	$P(Y)$	$P(X Y)$	$P(Y X)$
Prior probability shift		\neq	$=$	
Covariate shift	\neq			$=$
Concept shift			\neq	\neq

Table 30. Main types of dataset shift discussed in the literature. For the type of dataset shift on the row, symbol “ \neq ” indicates that the distribution on the column is assumed to change across L and U , while symbol $=$ indicates that the distribution is assumed to remain invariant. The last column indicates the section of the present work where this type of shift is discussed in detail.

L , since both L and σ would be expected to display the same prevalence values. This “method” is called, in the quantification literature, the *maximum likelihood prevalence estimator* (MLPE), and is considered a trivial baseline that any genuine quantification system is expected to beat in situations characterised by dataset shift.

We thus assume the existence of two unknown joint probability distributions $P_L(X, Y)$ and $P_U(X, Y)$ (where U represents the unlabelled data to which a quantifier must be applied) such that $P_L(X, Y) \neq P_U(X, Y)$ (the *dataset shift assumption*). The ways in which the training distribution and the test distribution may differ, and the effect these differences can have on the performance of quantification systems, is the main subject of the present work.

Any joint probability distribution $P(X, Y)$ can be factorised, alternatively and equivalently, as:

- $P(X, Y) = P(X|Y)P(Y)$, in which the marginal distribution $P(Y)$ is the distribution of the class labels, and the conditional distribution $P(X|Y)$ is the class-conditional distribution of the covariates. This factorization is convenient in *anti-causal learning* (i.e., when predicting causes from effects) [537], i.e., in *problems of type $Y \rightarrow X$* [538].
- $P(X, Y) = P(Y|X)P(X)$, in which the marginal distribution $P(X)$ is the distribution of the covariates and the conditional distribution $P(Y|X)$ is the distribution of the labels conditional on the covariates. This factorization is convenient in *causal learning* (i.e., when predicting effects from causes) [537], i.e., in *problems of type $X \rightarrow Y$* [538].

Which of these four ingredients (i.e., $P(X)$, $P(Y)$, $P(X|Y)$, $P(Y|X)$) change or remain the same across L and U , gives rise to different types of shift, as discussed in [526, 539]. In this section we turn to describing the types of shift that we consider in this study. To this aim, also recalling that the related terminology is sometimes confusing in this respect (as also noticed by [526]), we clearly define each type of shift that we consider.

When training a model, using our labelled data, to issue predictions about unlabelled data, we expect some relevant general conditions to be invariant across the training distribution and the unlabelled distribution, since otherwise the problem would be unlearnable. In Table 30, we list the three main types of dataset shift that have been discussed in the literature. For each such type, we indicate which distributions are assumed (according to general consensus in the field) to vary across L and U , and which others are assumed to remain constant. In [540] we thoroughly discuss the relationships between these three types of shift and quantification.

It is immediate to note from Table 30 that, for any given type of shift, there are some distributions (corresponding to the blank cells in the table – e.g., $P(X)$ for prior probability shift) for which it is not specified if they change or not across L and U ; indeed, concerning what happens in these cases, the literature is often silent. In [540] we try to fill these gaps. We identify applicatively interesting subtypes of dataset shift based on different ways to fill the blank cells of Table 30, and propose





experimental protocols that recreate them in order for quantification systems to be tested under those conditions.

Thorough experiments that validate the described method are presented in [540].

10.2.3. Conclusion

Since the goal of quantification is estimating class prevalence, most previous efforts in the field have focused on assessing the performance of quantification systems in situations characterised by a shift in class prevalence values, i.e., by prior probability shift; in the quantification literature other types of dataset shift have received less attention, if any. In this work, we have proposed new evaluation protocols for simulating different types of dataset shift in a controlled manner, and we have used them to test the robustness to these types of shift of several representative methods from the quantification literature. The experimental evaluation we have carried out has brought about some interesting findings.

The first such finding is that many quantification methods are robust to prior probability shift but not to other types of dataset shift. When the simplifying assumptions that characterise prior probability shift (e.g., that the class-conditional densities remain unaltered) are not satisfied, all the tested methods (including the Saerens-Latinne-Decaestecker method (SLD), a top performer under prior probability shift) experience a marked degradation in performance.

A second observation is that, while previous theoretical studies indicate that the Probabilistic Classify and Count (PCC) method should be the best quantification method for dealing with covariate shift, our experiments reveal that its use should only be recommended when the class label proportions are expected not to change substantially (a setting that we refer to as *pure* covariate shift).

Such a setting, though, is fairly uninteresting in real-life applications, and our experiments show that other methods (particularly: the Saerens-Latinne-Decaestecker method (SLD) and Probabilistic Adjusted Classify and Count (PACC)) are preferable to PCC when covariate shift is accompanied by a change in the priors. However, even SLD becomes unstable under certain conditions in which both covariates and labels change. We argue that such a setting, which we have called *local* covariate shift, shows up in many applications of interest (e.g., prevalence estimation of plankton subspecies in sea water samples [541], or seabed cover mapping [542], in which finer-grained unobserved classes are grouped into coarser-grained observed classes).

Finally, our results highlight the limitations that all quantification methods exhibit when coping with concept shift. This was to be expected since no method can adapt to arbitrary changes in the functional relationship between covariates and classes without the aid of external information. The same batch of experiments also shows that concept shift may induce a change in the priors that can partially compensate the bias of a quantifier; however, such an improvement is illusory and accidental, and it is difficult to envision clever ways for taking advantage of this phenomenon.

Possible directions for future work include extending the protocols we have devised to other specific types of shift that may be application-dependent (e.g., shifts due to transductive active learning [543], to oversampling of positive training examples in imbalanced data scenarios [544], to concept shifts in cross-lingual applications), and to types of quantification other than binary (e.g., multiclass, ordinal, multi-label). The goal of such research, as well of the research presented in this work, is to allow a correct evaluation of the potential of different quantification methods when confronted with the different ways in which the unlabelled data we want to quantify on differs from the training data, and to stimulate research in new quantification methods capable of tackling the types of shift that current methods are insufficiently equipped for.





10.2.4. Relevant publications

- Pablo González, Alejandro Moreo, Fabrizio Sebastiani. Binary quantification and dataset shift: An experimental investigation. **Data Mining and Knowledge Discovery**. Forthcoming. [540] <https://arxiv.org/abs/2310.04565>

10.2.5. Relevant software/datasets/other outcomes

- The code needed to reproduce all our experiments is publicly available in open-source mode at https://github.com/pglez82/quant_datasetshift.

10.3. Kernel density estimation for multiclass quantification

Contributing partner: CNR

10.3.1. Introduction

Despite the fact that binary quantification (i.e., the setting in which the classes of interest are *positive* vs. *negative*) has been, by far, the most studied scenario in the quantification literature [529, 531, 532, 545–547], the truth is that many of the applications of quantification naturally arise in the multiclass regime, i.e., in cases in which there are more than two mutually exclusive classes.

The multiclass extension of an originally conceived binary method is sometimes trivial. However, not all methods readily lend themselves to such a seamless multiclass adaptation. An example of this can be found in the distribution-matching (DM) methods, first proposed by [548], which nowadays represent a very important family of methods in the quantification literature [549–552].

In a nutshell, DM approaches aim at reconstructing the distribution of the test datapoints by seeking for the closest mixture of the class-conditional distributions of the training datapoints. The mixture parameter yielding the best such matching is an estimate of the sought class prevalence values. While in principle this intuition seems directly applicable to the multiclass case, there are technical impediments that make it difficult in practice. Devising better ways to overcome these impediments is the central topic of this work.

Modelling the distribution of high-dimensional data is known to be extremely difficult. For this reason, most DM approaches first transform the datapoints into posterior probabilities, by means of a probabilistic classifier, and then try to model the distribution of posteriors in place of attempting to model the distribution of covariates (which seems hopeless especially in high dimensional spaces). Current DM methods model the distribution of posterior probabilities by means of *histograms*. In the binary case [553], this comes down to generating one histogram for the distribution of posteriors from the positive training instances, and another histogram for the distribution of posteriors from the negative training instances; at inference time, another such histogram is created from the test instances, and a mixture parameter combining the training histograms is sought so that the resulting mixture of histograms best matches the test one.

In this work, we propose an alternative representation mechanism for modelling the distribution of posterior probabilities that preserves information about inter-class interactions. In particular, we propose to replace the n class-dependent histograms with a single kernel density estimation (KDE) model for the distribution of n -dimensional vectors (posterior probabilities) lying in the unit $(n - 1)$ -simplex. We present compelling empirical evidence that our proposed KDE-based model, which we dub KDEy, does indeed improve the performance of previous (histogram-based) DM approaches. We also test the KDE-based representation within the maximum likelihood (ML) framework, and show our model often shows superior performance with respect to current state-of-the-art ML models for quantification as well.





10.3.2. Methodology

In this section, we turn to describe our KDE-based solution to the multiclass quantification problem. In a nutshell, the main idea of our method consists of switching the problem representation from discrete, univariate PDFs (histograms) of the posterior probabilities to continuous, multivariate PDFs on the unit $(n - 1)$ -simplex. Our PDFs will be represented by means of Gaussian Mixture Models (GMMs) obtained via KDE (Figure 34).

Below, we describe how to represent bags as GMMs using KDE. Then, we show two different ways of framing the optimization problem: one based on distribution matching and another based on maximum likelihood.

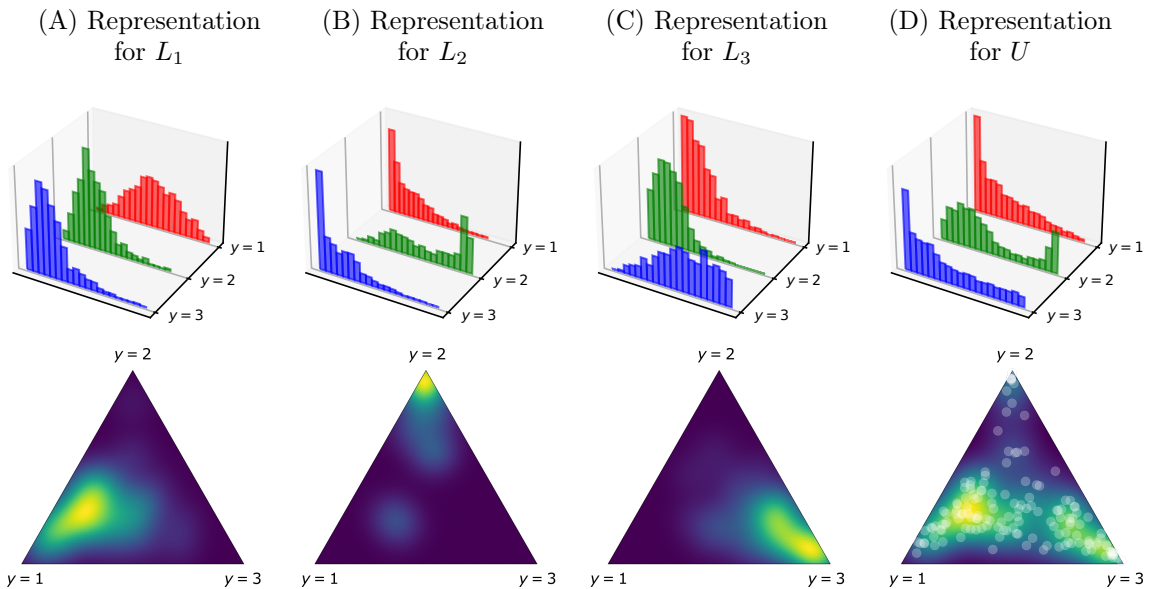


Figure 34. Problem representations obtained with traditional class-wise histograms (first row) and with our proposed mechanism based on GMMs (second row) on a 3-class sentiment problem (the dataset is called “wb” and belongs to the “Tweets” group described). The first three columns (A,B,C) show the model representations for the training sets L_1 , L_2 , and L_3 , while the last column (D) shows the representation for the test set U . The quantification problem is framed as the task of reconstructing (D) as a convex linear combination of (A,B,C).

Let p_θ be the kernel density estimator defined by

$$p_\theta(\mathbf{x}) = \frac{1}{|X|} \sum_{\mathbf{x}_i \in X} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right), \quad (18)$$

where $\theta = \{K, h, X\}$ are parameters of the model, with K the kernel function controlled by the bandwidth h , and X the set of reference points. The different density functions we will deal with all share the components K and h (that we treat as model hyperparameters) and differ only on the set of reference points. We will thus alleviate notation by simply writing p_X , instead of p_θ or $p_{\{K,h,X\}}$.

We will concentrate our attention on Gaussian kernels, i.e., the case in which the kernel $K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) = K(\mathbf{x}; \mathbf{x}_i, h)$ corresponds to the multivariate normal distribution $\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \Lambda^{-1})$ with mean $\boldsymbol{\mu} = \mathbf{x}_i$ and covariance matrix $\Lambda^{-1} = h^2 I_D$, with I_D the identity matrix of order D , that is

$$K(\mathbf{x}; \mathbf{x}_i, h) = \frac{1}{h^D (2\pi)^{\frac{D}{2}}} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|_2^2}{2h^2}\right).$$





Note also that D is the dimensionality of the input space, i.e., that $\mathbf{x} \in \mathbb{R}^D$. We here wrote D on purpose, since we are not assuming to be modelling the input space $\mathcal{X} = \mathbb{R}^d$ in which our covariates live, but rather the probability simplex $\tilde{\mathcal{X}} = \Delta^{n-1}$. For this reason, we first map every datapoint into a posterior probability by means of a soft classifier s . Note, thus, that in our case this translates into $D = n$ since our posterior probabilities have n dimensions (although only $n - 1$ degrees of freedom). To ease notation, we simply overline the symbol denoting a set of examples to indicate that its members have been converted into posterior probabilities; e.g., by \tilde{L}_i we indicate the set $\{s(\mathbf{x}) : \mathbf{x} \in L_i\}$ where $L_i = \{\mathbf{x} : (\mathbf{x}, y) \in L \wedge y = i\}$ as before. This mapping is carried out via k -fold cross-validation in the training set.

At training time, the quantifier needs to model a dedicated density function for each of the class-conditional distributions of posterior probabilities. Given a set of reference points for each class ($\tilde{L}_1, \tilde{L}_2, \dots, \tilde{L}_n$) and a mixture vector $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_n) \in \Delta^{n-1}$, we thus define $\mathbf{p}_\alpha : \Delta^{n-1} \rightarrow \mathbb{R}_{\geq 0}$ as

$$\mathbf{p}_\alpha(\tilde{\mathbf{x}}) = \sum_{i=1}^n \alpha_i p_{\tilde{L}_i}(\tilde{\mathbf{x}}), \quad (19)$$

where $\tilde{\mathbf{x}}$ is a vector of posterior probabilities. For a fixed $\boldsymbol{\alpha}$ and given a soft classifier s , the density of a datapoint $\mathbf{x} \in \mathcal{X}$ (i.e., a vector of covariates) is estimated by the KDE mixture as $\mathbf{p}_\alpha(s(\mathbf{x}))$. Note that, since we chose our kernel to be Gaussian, \mathbf{p}_α is also a GMM.

GMMs are less sensitive to the choice of the bandwidth than histograms are to the choice of binning. The latter follows from the fact that the density-mass “blocks” become smooth when using Gaussian kernels, and from the fact that the centers of these are not data-agnostic, as in histograms. In other words, a GMM in which the Gaussians are centered at the datapoints is akin to a model of soft assignments (the location of a datapoint influences the density estimation in any other location of the space) conversely to the hard assignments of histograms (the location of a datapoint affects one and only one bin in the model).

Finally, note that replacing histograms with GMMs brings about the following advantages:

1. GMMs scale smoothly with the number of classes.
2. GMMs rely on soft assignments, which effectively retain more information.
3. Inter-class correlations are preserved.

10.3.3. Conclusion

Many disciplines exist for which the interest lies in knowing the distribution of the classes in unlabelled data samples, and in which we are not interested in individual label predictions. A myriad of quantification methods have been proposed so far, among which, distribution matching methods represent a fairly important family of approaches. While the distribution matching methods are natively binary, some extensions have been proposed to the multiclass case in the literature. In this article, we argue that such extensions are suboptimal, since they fail to capture the possible inter-class interactions that might exist in the data.

We have investigated possible ways for bringing these class-class correlations into the model; to this aim, we propose to switch the representation mechanism from independent class-wise histograms to multivariate GMMs obtained through KDE. We have presented different instances of our KDE-based solution, depending on whether the solution is framed as a distribution matching setting, or whether it is framed under the maximum likelihood framework. In all cases, our proposed methods performed very well, beating the previously proposed histogram-based models, and also





setting a new state-of-the-art result in the multiclass task T1B of the LeQua competition [554]. The method has demonstrated competitive performance also in binary problems, hence proving itself a versatile approach for quantification problems.

The methods we present introduce a new hyperparameter: the bandwidth of the kernel. The experiments we have carried out indicate that our method behaves stably with respect to the hyperparameter (small variations produce small effects in performance). This seems to suggest more sophisticated alternatives might be explored that aim at finding the optimal value avoiding a brute-force optimization. In future work, we plan to pursue this idea.

10.3.4. Relevant publications

- Alejandro Moreo, Pablo González, and Juan José del Coz. Kernel density estimation for multiclass quantification. **Machine Learning**. Submitted for publication. [555]

10.3.5. Relevant software/datasets/other outcomes

- The code needed to reproduce all our experiments is publicly available in open-source mode at <https://github.com/HLT-ISTI/QuaPy/tree/kdey>.

10.4. Quantification using permutation-invariant networks based on histograms

Contributing partner: CNR

10.4.1. Introduction

One of the main advantages of adopting deep neural network architectures (DNNs) for quantification is that DNNs allow the learning process to handle bags of examples (labeled by their class prevalence values) instead of individual examples (labeled by class). Following this intuition, a change in the learning paradigm with respect to the traditional one was first proposed in [556]. In this work, we offer an in-depth exploration of the implications of this change of paradigm, by analyzing the main advantages and limitations with respect to traditional approaches to quantification. Conversely, traditional quantification methods adopt an *asymmetric* approach in which a classifier is trained to infer the class of the individual examples and in which the label predictions are used to estimate the prevalence of the classes in the bag. This way, the training labels (class labels attached to the example) and the labels to be predicted (class prevalence values attached to the bag) are not *homologous*. In contrast, following the approach proposed in [556], we can reframe the quantification problem as a *symmetric* supervised learning task in which the training set consists of a collection of bags containing examples labeled at the aggregate level (i.e., without individual class labels). This formulation posits the quantification problem as a multivariate regression task, in which the labels provided for training and the labels we need to predict become homologous. Throughout this work, we will demonstrate further advantages of this formulation. Among them, and in contrast to traditional quantification methods, the quantifier becomes capable of optimizing any specific loss function.

With this aim, our work investigates the application of DNNs to the symmetric quantification problem. The work begins by addressing a central issue that arises when making predictions for entire bags rather than for individual examples, namely, how to represent bags in a permutation-invariant manner.

Two influential DNN architectures have been proposed for set processing: DeepSets [557] and SetTransformers [558]. The former employs a pooling layer like max, average, or median, to





summarize each bag, while the latter uses a transformer architecture without positional encoding. These approaches were designed as universal approximation functions for set-based problems. Here, we propose a new architecture, called HistNetQ, relying on histogram-based layers. The rationale why histograms seem promising is two-fold: histograms are naturally geared towards representing densities and convey more information than plain statistics (like the mean, or median). We will show that histogram-based layers can be seen as a generalization of the pooling layers proposed in [556, 557].

The contributions of this work are three-fold. First, we analyze the symmetric approach of [556] for quantification, discussing its strengths and limitations. Secondly, we empirically assess the suitability of previously proposed permutation-invariant layers to the quantification problem. Finally, we propose HistNetQ, a new permutation-invariant architecture based on differentiable histograms, specifically useful for quantification tasks.

Our experiments show two main results: i) HistNetQ outperforms not only traditional quantification methods and previous general-purpose DNN architectures for set processing but also state-of-the-art quantification-specific DNN methods [530, 556] in the LeQua [559] competition, the only competition entirely devoted to quantification held to date, ii) HistNetQ proves competitive also under the asymmetric approach too, that is, when a set of training bags is not available and must be generated from D via sampling.

10.4.2. Methodology

In this work, we propose a permutation-invariant layer for quantification that gains inspiration from histograms. Histograms represent powerful tools for describing sets of values: they are directly aligned with the concept of counting, and they disregard the order in which the values are presented. However, histograms are not differentiable operators and hence cannot be directly employed as building blocks in a deep learning model. In order to overcome this impediment, histograms can be approximated by using common differentiable operations such as convolutions and pooling layers. Different realizations of this intuition have been reported in the literature of computer vision [560–562] but, to the best of our knowledge, no one before has investigated differentiable histograms in quantification.

Previous attempts for devising differentiable histograms differ in how these are implemented. On the one hand, [561, 562] proposed soft variants in which every value can potentially contribute to more than one bin, based on the distance of the value to the center of the bin and the width thereof. On the other hand, in [563] the authors propose a hard variant, that is, every value only contributes to the bin in which the value falls. Throughout preliminary experiments we carried out using all variants, we found that the differences in performance were rather small. The hard variant proved slightly better in such experiments (in terms of validation loss) and is our variant of choice. Other architectures and their results are discussed in the supplementary material of our paper [564].

More formally, given a bag of n data examples $B = \{\mathbf{x}_i\}_{i=1}^n$, with $\mathbf{x}_i \in \mathcal{X}$, our goal is to compute a histogram for every feature vector $\{\mathbf{f}_k\}_{k=1}^z$, where $\mathbf{f}_k \in \mathbb{R}^n$ represents the values of the k -th feature across the n instances in the bag B , and where z is the number of features extracted (i.e., every histogram is computed along a different column from a $n \times z$ matrix representing B). The hard differentiable histogram layer proposed in [563] takes a user-defined hyperparameter N determining the (fixed) number of bins (we use the same number of bins for all feature vectors), and defines $\{(\mu_b^{(k)}, w_b^{(k)})\}_{b=1}^N$, the bin centers and widths, as independent learnable parameters for each feature vector \mathbf{f}_k . The value in the b -th bin of the k -th histogram is defined by:

$$H_b^{(k)}(B) = \frac{1}{n} \sum_{i=1}^n \phi(\mathbf{f}_k[i]; \mu_b^{(k)}, w_b^{(k)}), \quad (20)$$



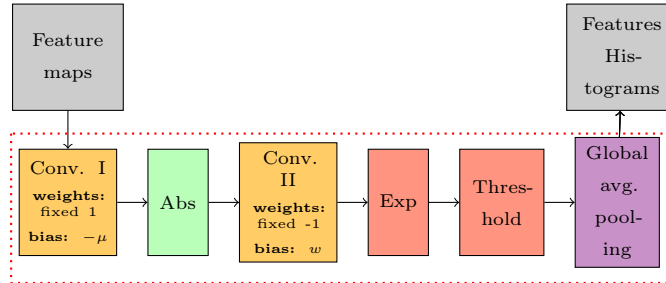


Figure 35. Learnable histogram layer with hard binning and learnable bin centers and widths. The individual components are common operations used in DL frameworks that we use to compute Equation 21.

where ϕ is defined by:

$$\phi(v; \mu, w) = \begin{cases} 0, & \text{if } 1.01^{w-|v-\mu|} \leq 1 \\ 1, & \text{otherwise.} \end{cases} \quad (21)$$

The value 1.01 in Equation 21 is justified in [563] simply as a value that yields slightly smaller values than 1 when the exponent is < 0 and slightly bigger values than 1 if the exponent is > 0 . This, in combination with a threshold operation, results in a (differentiable) mechanism to detect which values fall into which bin (see Figure 35 for a graphical representation of the layer).

Note that we compute densities (by dividing the counts by n) and not plain counts, in order to factor out the effect of the bag size in the final representation. Note also that the total number of parameters of a differentiable histogram layer is $2Nz$. Since the bin centers and widths are learnable, the output can contain interval “gaps” (i.e., intervals in which values are not taken into account), interval overlaps (thus allowing one value to contribute to more than one overlapping bin at the same time), or even zero-width bins. This means that the output of the layer is not strictly a histogram, but this allows the model to control the complexity of the representation (should N be too high, the model can well learn to overlap bins or create zero-width ones).

It is worth noting that the quantification method HDy [553] also relies on histograms. However, there are significant differences between HDy and HistNetQ. To begin with, HistNetQ models histograms on the latent representations of the (potentially high-dimensional) data, whereas HDy models histograms on the posterior probabilities returned by a soft classifier. Also, as HistNetQ uses a symmetric approach and learns directly from bags, it does not need to impose any learning assumption, while HDy instead relies on the prior probability shift assumptions. Lastly, HistNetQ enables the optimization of a specific loss function during the learning process, while this is not possible in HDy.

Lemma 1. *Hard differentiable histogram layers are permutation-invariant.*

Proof. The proof is straightforward. The value $H_b^{(k)}(B)$ is computed by summing over the values returned by the ϕ function. Although $\pi(B)$, with π any permutation function, alters the order of the values within the feature vectors \mathbf{f}_k , this ordering does not affect the final counts since:

$$\frac{1}{n} \sum_{i=1}^n \phi(\mathbf{f}_k[i]; \mu_b^{(k)}, w_b^{(k)}) = \frac{1}{n} \sum_{i=1}^n \phi(\pi(\mathbf{f}_k)[i]; \mu_b^{(k)}, w_b^{(k)}),$$

and hence $H_b^{(k)}(B) = H_b^{(k)}(\pi(B))$. □





One of the claims of the work is that polling layers like average, median, or max proposed for set operations [556, 557] can be seen as simplified models (or ablations) of our proposal of using histogram layers (in other words, that a histogram subsumes the information conveyed by these statistics). In order to verify this, we designed a toy experiment where a small neural network is trained to learn each of the aggregation functions (average, median, and max). To this aim, we equip our network with a single histogram layer of 64 bins, followed by just two fully connected layers (sizes 32 and 16). The network is then trained on randomly generated vectors of 100 real values between $[0, max]$, where max is a random number in the range $[0, 1]$. The absolute errors are pretty low: 0.0055 (average), 0.0090 (median), and 0.0219 (max) suggesting that histograms are richer representations than the average, median, or max. As the histogram layer can capture the distribution of the data, it provides a more comprehensive view of the data beyond single summary statistics, something that makes them a promising approach for machine learning tasks that require a density estimation method over sets.

10.4.3. Conclusion

This work introduces HistNetQ, a DNN for quantification that relies on a permutation-invariant layer based on differentiable histograms. We carried out experiments using two different quantification problems (from computer vision and text analysis) in which we compared the performance of HistNetQ against previously proposed networks for set processing and also against the most important algorithms from the quantification literature. The results show that HistNetQ achieved state-of-the-art performance in both problems. From a qualitative point of view, HistNetQ also displays interesting properties like i) the ability to directly learn from bags labeled by prevalence, which allows HistNetQ to be applied to scenarios in which traditional methods cannot; and ii) the possibility to directly optimize for specific loss functions.

This research may hopefully offer a new viewpoint in quantification learning, since our results suggest that exploiting data labeled at the aggregate level might be preferable, in terms of quantification performance, than exploiting data labeled at the individual level. Overall, this study seems to suggest that HistNetQ is a promising alternative for implementing the symmetric approach in real applications, obtaining state-of-the-art results that surpass previous approaches.

Future work may include i) studying the capabilities of HistNetQ when confronted with types of dataset shift other than prior probability shift [565, 566] and ii) exploring potential applications of this architecture to other problems that, like quantification, require learning a model from density estimates over sets of examples.

10.4.4. Relevant publications

- Olaya Pérez-Mon, Alejandro Moreo, Juan José del Coz, and Pablo González. Quantification using permutation-invariant networks based on histograms. **Neural Computing and Applications**, 2023. Submitted for publication. [564]

10.4.5. Relevant software/datasets/other outcomes

- The code needed to reproduce all our experiments is publicly available in open-source mode at <https://github.com/a2032/a2032>.

10.5. Quantifying query fairness under unawareness

Contributing partner: CNR





10.5.1. Introduction

Traditionally, the main purpose of an information retrieval (IR) system is to retrieve documents that are most relevant to a user’s query. However, IR systems that focus solely on optimising the search results’ relevance might inadvertently introduce biases and unfairness against certain groups of documents. Typically, to measure the unfairness of an IR system, existing metrics compare the distribution of groups in a ranking with a target distribution, e.g., the distribution of the groups in the entire collection. For the assessment of the group distributions, these metrics rely on the true group labels associated with each individual document in the ranking. However, when the groups are defined using demographic or sensitive information, these labels are often unknown. Such a setting is typically known as “fairness under unawareness”.

To obtain the group membership of individual documents, the labels can be inferred using machine-learned classifiers, and the prevalence of the different groups can be estimated by simply “counting” the predicted labels. However, such approaches are known to yield suboptimal performance in the presence of dataset shifts, thereby leading to an inaccurate assessment of fairness. In this work, we thus propose to use quantification methods (i.e., techniques specifically devised for the estimation of class proportions under dataset shift) for reliably measuring fairness under unawareness in IR systems. Our experiments on the TREC 2022 Fair Ranking Track collection show that quantification techniques significantly enhance the accuracy of determining group prevalence in rankings, which leads to more accurate fairness measurements. We find that our method (which extends beyond binary sensitive groups) significantly outperforms existing baselines for multiple sensitive attributes. To the best of our knowledge, this is the first work to establish a robust protocol for reliably measuring fairness under unawareness across multiple groups of documents.

10.5.2. Methodology

Given that many of the proposed evaluation measures for assessing fairness in rankings rely on estimating group distributions, one might reasonably expect that simply incorporating some of the most sophisticated quantification methods would enhance the accuracy of metric prediction. However, contrary to this expectation, our preliminary experiments have shown otherwise. Indeed, we observed that the application of quantification techniques almost always led to a deterioration in the ranking fairness prediction with respect to Classify and Count (CC).

The reason is that, while it makes sense to think that the prior distribution of the groups might change over time, or on a per-query basis, the truth is that the case we are facing here is not an example of prior probability shift (PPS). To see why, consider the random variable Q that takes on values 1 (“the document is relevant”) and 0 (“the document is irrelevant”) with respect to a specific query. Note that the class-conditional distribution is a mixture of relevant and irrelevant documents, i.e., $P(X, Q|Y) = P(X|Y, Q = 1)P(Q = 1) + P(X|Y, Q = 0)P(Q = 0)$; however, we might expect $P_{tr}(Q = 1) \ll P_{te}(Q = 1)$, since it is likely that the vast majority of the training data used for learning our quantifier is irrelevant to a specific query, while the majority of the documents retrieved for the query are indeed relevant to it. The class-conditional distributions are thus different, and this clashes with the PPS assumptions.

Note that the random variable Q might be regarded as the “selection variable”, which is representative of a type of dataset shift known as *sample selection bias* (SSB).¹²

While different quantification methods have shown varying degrees of effectiveness in addressing different types of shift [540], we are unaware of the existence of any quantification method robust to

¹²Sample selection bias is often defined differently, since the selection variable has an effect on the way the training instances (and not the test instances, as in our case) are selected [539].





SSB. In the following, we propose a first approach to make quantification methods robust against SSB.

The reason why traditional quantifiers failed in the presence of SSB in our preliminary experiments has to do, as previously hinted, with the violation of the assumption $P_{tr}(X|Y) = P_{te}(X|Y)$, which characterizes PPS.

Let us turn back to the ACC method to illustrate the problem (a similar rationale applies to other quantification algorithms as well). Recall that ACC replaces $P_{te}(\hat{Y} = 1|Y)$ with $P_{tr}(\hat{Y} = 1|Y)$ on the grounds that the class-conditional distributions of training and test datapoints are the same. That $P_{tr}(X|Y) = P_{te}(X|Y)$ implies $P_{tr}(\hat{Y}|Y) = P_{te}(\hat{Y}|Y)$ follows from the fact that \hat{Y} depends uniquely on X by means of a function (the classifier); inasmuch as the classifier is a measurable function this equivalence holds [567].

In general, $P_{te}(\hat{Y} = i|Y = j)$ represents the *classification rates* of the classifier in the *test set*, and is given by

$$P_{te}(\hat{Y} = i|Y = j) = \mathbb{E} [\mathbf{1}[h(\mathbf{x}) = i]]_{\mathbf{x} \sim P_{te}(X|Y=j)} \quad (22)$$

Of course, we do not have access to the true distribution $P_{te}(X|Y = j)$ of the expectation, but if we could assume $P_{te}(X|Y) = P_{tr}(X|Y)$, then this expectation could be estimated by means of an empirical distribution $\mathbf{x}_1, \dots, \mathbf{x}_m \sim P_{tr}(X|Y = j)$, as

$$P_{te}(\hat{Y} = i|Y = j) \approx \frac{1}{m} \sum_{k=1}^m \mathbf{1}[h(\mathbf{x}_k) = i] \quad (23)$$

Although we know that this assumption is flawed in the presence of SSB, one fundamental observation arises: the pitfall stems from the choice of the empirical distribution used to characterize the classifier h , rather than from the classifier itself.

This is important since most quantifiers use a training set $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$, $\mathbf{x}_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$ to both learn a classifier h and estimate, via cross-validation,¹³ its classification rates (in our binary example, this reduces to estimating tpr and fpr). Note also that we could assume that training a classifier is costly, whereas learning the classification rates is rather inexpensive (it only involves issuing predictions and rearranging counts).

We propose to disentangle the classifier-training phase from the correction learned by the quantifier. We therefore assume to have access to two sets of labelled data, $L_{cls} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ that we use to train our classifier (offline since it is costly), and $L_q = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m'}$ that we use to learn the correction (at query time since it is inexpensive). What remains ahead in order to make quantification robust to SSB is to counter the selection bias in L_q .

The main idea underlying this work is the following. Reducing the sampling bias from the test data is impossible; the test documents are retrieved by the search engine precisely to guarantee the documents are relevant to a specific query. The main idea we explore in this work is *to use the same search engine with the same query* to retrieve, from an *auxiliary* pool of training documents (different from those used to train the classifier), a list of training documents that are biased towards the query. This way, the empirical distribution L_q that we retrieve from the auxiliary pool can be regarded as a sample from a query-biased distribution $P_q(X, Q|Y)$ and, since we can now assume $P_q(Q) \approx P_{te}(Q)$ (i.e., both distributions are biased towards the query) then we can also assume $P_q(X, Q|Y) \approx P_{te}(X, Q|Y)$ and restore the fundamental PPS assumption. Since the SSB in L_q mimics the SSB that affects the test data, *the sampling bias shift vanishes*.

We thus consider an auxiliary set of labelled documents \mathcal{L} containing pairs (\mathbf{x}_i, y_i) labelled by sensitive attributes (hereafter called the “training pool”). From this training pool (\mathcal{L}), we select,

¹³This is in order to avoid the same datapoint being classified to take part in the training of the classifier.





using the same retrieval model and query that we issue on the test pool (\mathcal{U}), a ranked list of (labelled) documents L_q that we use to learn a per-query quantification correction to estimate group prevalence in the top- k prefixes of (unlabelled) rankings U_q .

We have previously explained the intuitions behind our method with respect to (binary) ACC, a relatively simple quantifier. In this section, we generalize the rationale to more sophisticated multiclass quantification methods.

In the modern perspective of multiclass quantification [568], most quantifiers can be framed as the problem of solving for $\mathbf{p} \in \Delta^{n-1}$ the system of linear equations

$$\mathbf{t} = \mathbf{M}\mathbf{p} \quad (24)$$

where $\mathbf{t} = \Phi(U)$ is the representation of the test bag U , and $\mathbf{M} = [\Phi(L_1), \dots, \Phi(L_n)]$ is the matrix containing the class-wise representations of the training sets $L_i = \{\mathbf{x}_k : (\mathbf{x}_k, y_k) \in L, y_k = i\}$, for a given representation function $\Phi : \mathbb{N}^{\mathcal{X}} \rightarrow \mathbb{R}^z$ that embeds bags into z -dimensional vectors, for some z .

Most quantifiers rely on a representation function of the form:

$$\Phi(\mathbf{X}) = \frac{1}{|\mathbf{X}|} \sum_{\mathbf{x} \in \mathbf{X}} \phi(\mathbf{x}) \quad (25)$$

in which a surrogate instance-wise representation function $\phi : \mathcal{X} \rightarrow \mathbb{R}^z$ is invoked, thus effectively computing a mean embedding.

Different choices for Φ and ϕ give rise to different instances of quantification methods. For example, ACC comes down to choosing, as our representation function ϕ , the output of a crisp classifier h encoded as a one-hot vector. The columns of \mathbf{M} thus represent the classification rates of the classes (as estimated on training data), and the problem comes down to reconstructing the class counts of the test examples as a linear combination of the training ones, by solving $\mathbf{p} = \mathbf{M}^{-1}\mathbf{t}$.

More sophisticated methods exist, though. For example, *Probabilistic Adjusted Classify and Count* (PACC) [529] (that we use in our experiments), defines ϕ as a probabilistic classifier returning the posterior probabilities for each class. When \mathbf{M} is not invertible [569], the variant we employ frames Equation 24 as the minimization problem

$$\mathbf{p}' = \arg \min_{\mathbf{p} \in \Delta^{n-1}} \|\mathbf{t} - \mathbf{M}\mathbf{p}\|^2 \quad (26)$$

We also use KDEy [555], a state-of-the-art multiclass quantification method that defines Φ as a Gaussian mixture model, obtained via kernel density estimation, of the posterior probabilities returned by a probabilistic classifier. We use the maximum-likelihood variant that solves Equation 24 as the minimization problem

$$\mathbf{p}' = \arg \min_{\mathbf{p} \in \Delta^{n-1}} \mathcal{D}_{\text{KL}}(\mathbf{t} \parallel \mathbf{M}\mathbf{p}) \quad (27)$$

where \mathcal{D}_{KL} is the well-known Kullback-Leibler divergence.

Note that PACC and KDEy both rely on a probabilistic classifier that is trained in advance. Equations 26 and 27 do only require converting the documents in L_q and the test documents U_q (both retrieved for the same query but from different pools) into posterior probabilities and solving the optimization problem. Both operations are rather fast given modern optimization routines and given the fact that the number of retrieved documents (either for training and test) is typically small (in our experiments, this number is bounded by 1,000 documents).

Thorough experiments that validate the described method are presented in [570].





10.5.3. Conclusion

In this work, we have investigated how to reliably assess the fairness of search results in rankings for multiclass groups of documents under unawareness of document labels. We have demonstrated that simply counting over the predictions of a classifier often leads to unreliable fairness assessments. To address this limitation, we have proposed the use of quantification to accurately estimate the prevalence of different groups in a ranking. While most quantification techniques are designed to counter prior probability shift, the problem we face here is instead affected by a different type of shift: sample selection bias. To the best of our knowledge, our approach is the first attempt towards making quantification robust to this type of shift. Our extensive evaluation on multiclass fairness groups of a publicly available fairness ranking benchmark has shown that our approach can successfully predict query fairness, and do so more accurately than existing methods from the literature.

In future work, we aim to investigate the suitability of a normalised-discounted variant of RAE for fairness evaluation. We are also interested in exploring different collections where the group prevalence may have naturally varied across training and test conditions.

10.5.4. Relevant publications

- Thomas Jaenich, Alejandro Moreo, Alessandro Fabris, Graham McDonald, Andrea Esuli, Iadh Ounis, and Fabrizio Sebastiani. Quantifying query fairness under unawareness. **Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM 2024)**, Boise, US. Submitted for publication. [570]

10.5.5. Relevant software/datasets/other outcomes

- The code needed to reproduce all our experiments will be made publicly available in open-source mode once the paper is accepted for publication.

10.5.6. Relevance to AI4Media use cases and media industry applications

While this development does not play a role in any specific AI4Media use case (this being a very recent development), the solution proposed has a lot of potential for use in the media sector. One of the possible applications has to do with implementing news search engines that return “fair” rankings, i.e., rankings that give fair exposure to members of protected groups.

10.6. Learning to quantify graph nodes

Contributing partner: CNR

10.6.1. Introduction

This work deals with the related but different task of *network quantification* (NQ), i.e., performing quantification under PPS on interlinked datapoints. In other words, the labeled and the unlabeled datapoints are nodes of a graph (actually: of the *same* graph) and therefore do not meet the standard i.i.d. assumption. NQ shares common traits with the better-known task of node classification [571]. However, the end goals of the two tasks are quite distinct: while node classification is focused on predicting the individual classes of the unlabeled nodes, NQ is about predicting their aggregate class distribution. Figure 36 summarizes the main differences between these tasks. As noted by [572], scenarios in which NQ is applicable abound, since in many settings (e.g., social network



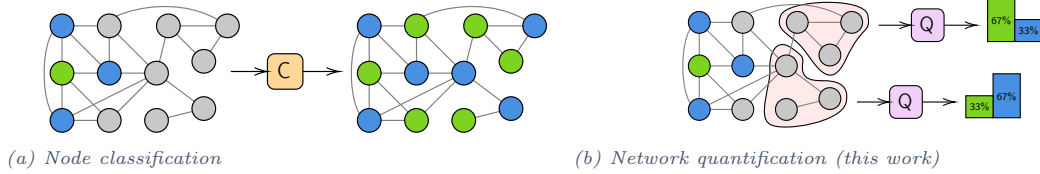


Figure 36. We show the differences between node classification and network quantification on a partially unlabeled graph where nodes may belong to the “blue” or “green” classes. unlabeled nodes are shown in gray. Node classification (a) is performed by a node classifier C , which takes as input the partially unlabeled graph and returns as output an isomorphic graph where the class of the unlabeled nodes has been predicted. In contrast, network quantification (b) uses a quantifier Q , which takes as input subsets of unlabeled nodes (in light pink shades) and returns as output their class distribution. In this work, we study network quantification under prior probability shift.

analysis, epidemiology, market research) it is desirable to estimate how a population of interrelated individuals is distributed according to the indicators (i.e., classes) of interest.

Moving the focus of quantification onto graph nodes adds an additional layer of complexity, since real-world graphs where NQ could be applied are usually large-scale and characterized by complex properties such as non-linear connectivity patterns and heterophily (i.e., prevalence of inter-class edges). Ideally, to be proficient in NQ, a learning method should (i) exploit the network connectivity to emit coherent predictions; (ii) be powerful enough to capture the complex properties of real-world networks; (iii) be flexible enough to adapt to their possibly heterophilic nature; and (iv) be efficient and resource-friendly to make operating at scale feasible. However, existing methods for NQ hardly satisfy all these requirements.

To satisfy these *desiderata* we propose XNQ, a model that integrates the representational capabilities of randomized recursive Graph Neural Networks with a powerful Expectation-Maximization approach for quantification. XNQ is purposely designed to be resource-efficient, scalable, and resilient to the challenges of heterophily. Through extensive experimental evaluation, we demonstrate that XNQ significantly outperforms the best methods proposed so far for NQ, setting a new state of the art. Additionally, we validate our design through comprehensive ablation studies, and we show that XNQ achieves the best trade-off in terms of performance and computational efficiency.

10.6.2. Methodology

We start this section by restating the objective of NQ for clarity. Given a partially labeled graph G with $V = L \cup U$, our goal is to produce an estimate $\hat{p}_U(\oplus)$ of the proportion of positive nodes in the unlabeled subset. To satisfy the stringent requirements of NQ, we developed the *eXtreme Network Quantifier* (XNQ) model, referring to its enhanced efficiency and effectiveness. At a high level, XNQ is composed of three modules applied sequentially:

1. An unsupervised *node embedder* which computes node embeddings leveraging the node features and the graph structure;
2. An intermediate *readout classifier* which takes the node embeddings as input and computes the node class posterior predictions as output;
3. A downstream *aggregative quantifier* which aggregates the classifier’s posterior predictions and estimates the class prevalence values.

Below, we describe each component in detail.

Differently from previous literature NQ methods, XNQ leverages the node representations computed by a GNN model in order to exploit information on input node features as well as the





comprehensive graph topology. To satisfy the requirement of efficiency, XNQ exploits a reservoir computing GNN such as GESN to embed nodes in an *unsupervised* and *untrained* fashion, allowing it to scale to larger networks without requiring a too large fraction of annotated nodes: as opposed to end-to-end trained GNNs, target nodes are not used for learning node representations, but only for training the classifier readout. GESN-based models have proven effective in solving node classification tasks, reaching state-of-the-art accuracy on several heterophilic graph benchmarks, while also reducing computation time compared to fully-trained graph neural networks [573]. Specifically, in XNQ node embeddings $\mathbf{h}_v^{(T)}$ are recursively computed by the following dynamical system, called the *reservoir*, which implements an untrained GNN layer:

$$\mathbf{h}_v^{(0)} \leftarrow \mathbf{0}, \quad \mathbf{h}_v^{(t)} \leftarrow \tanh \left(\mathbf{W}_{\text{in}} \mathbf{x}_v + \sum_{u \in \mathcal{N}(v)} \hat{\mathbf{W}} \mathbf{h}_u^{(t-1)} + \mathbf{b}_{\text{in}} \right), \quad (28)$$

where $\mathbf{W}_{\text{in}} \in \mathbb{R}^{d' \times d}$ and $\hat{\mathbf{W}} \in \mathbb{R}^{d' \times d'}$ are the input-to-reservoir and the reservoir recurrent weights, respectively ($\mathbf{b}_{\text{in}} \in \mathbb{R}^{d'}$ is the input bias). The embedding dimension $d' \in \mathbb{N}$ is a hyperparameter chosen by model selection, and its value is typically much larger than the input dimension d . Reservoir weights are randomly initialized from a uniform distribution, and then rescaled to the desired input range and reservoir spectral radius (also chosen via model selection), without requiring any training. The number of iterations $1 \leq t < T$ is set to be larger than the graph diameter, so as to have a comprehensive receptive field. Of crucial importance is the Lipschitz constant of the recursive map defined in Eq. 28, which is controlled by setting the spectral radius of $\hat{\mathbf{W}}$, i.e., the largest eigenvalue modulus $\rho(\hat{\mathbf{W}})$. Initializing the recurrent matrix with $\rho(\hat{\mathbf{W}}) < 1/\alpha$, where $\alpha = \rho(\mathbf{A})$ is the graph spectral radius, implies that the map is contractive and that the sensitivity of the node embeddings to long-range interactions is exponentially vanishing [573]. Since this setting leads to over-smoothing, node-level tasks frequently benefit from recurrent matrix initializations with $\rho(\hat{\mathbf{W}}) > 1/\alpha$. This holds particularly true for heterophilic graphs, where sensitivity only to the immediate neighbors may lead to misleading representations.

After being iteratively computed by Eq. 28 on the whole graph G , the node embeddings $\mathbf{h}_v^{(T)}, \forall v \in V$ are passed to the readout classifier.

XNQ uses a trained logistic regression readout module to compute the node predictions. The use of logistic regression is tightly coupled with choosing a GESN-based embedder, since the high-dimensional expansion performed by the reservoir usually results in a linear separation of the embeddings. Consequently, a linear model can be used to learn the classification rule, further contributing to making the approach extremely efficient. More precisely, our readout module takes the node embeddings $\mathbf{h}_v^{(T)}$ as input, and computes a raw posterior probability $\bar{y}_v \in [0, 1]$ as:

$$\bar{y}_v \leftarrow \text{sigmoid} \left(\mathbf{w}_{\text{out}} \mathbf{h}_v^{(T)} + b \right), \quad (29)$$

where $\mathbf{w}_{\text{out}} \in \mathbb{R}^{d'}$ are learnable weights and $b \in \mathbb{R}$ is a learnable bias. Once the readout has been learned, the output probabilities are calibrated and passed to the downstream quantifier. Calibration entails adjusting the output probabilities such that $\Pr(Y = \oplus | \bar{Y} = \bar{y}_v) \approx \bar{y}_v$, where \bar{Y} is a random variable that ranges over $[0, 1]$. In other words, by calibrating the output of the readout we are adjusting the predicted probabilities to approximately match the observed class frequencies. Calibration is achieved with by transforming the raw posterior probabilities as follows:

$$\hat{y}_v \leftarrow \frac{1}{1 + \exp(a \bar{y}_v + b)}, \quad (30)$$

where $a, b \in \mathbb{R}$ are parameters learned with maximum likelihood [574]. We choose this calibration method instead of isotonic regression as the latter requires more samples.





The readout classifier is trained and calibrated using the labeled node embeddings $\mathbf{h}_v^{(T)}$, $\forall v \in L$. Then, we use it to predict the missing labels for the unlabeled nodes in U , obtaining a set of calibrated posterior probabilities $\Pr(\oplus | \mathbf{h}_v^{(T)})$, $\forall v \in U$. These, together with the positive class proportion observed in the training set $p_L(\oplus)$, become the inputs for the next step.

The goal of XNQ's downstream quantifier is to output the desired estimate $\hat{p}_U(\oplus)$ using the observed training proportion $p_L(\oplus)$ and the posterior probabilities $\Pr(\oplus | \mathbf{h}_v^{(T)})$, $\forall v \in U$ as inputs. We do so by adapting the *Saerens-Latinne-Decaestecker* method [575] (**SLD**) to our setting. The rationale behind adapting SLD are its strong performance (it was the best performer in a recent data challenge devoted to quantification on data affected by PPS [554]) and its desirable theoretical guarantees. Indeed, SLD is proven to be *Fisher-consistent* under PPS [576], i.e., its class prevalence estimates are guaranteed correct under PPS if computed on the whole populations of interest (instead of the limited samples L and U). Basically, we use an instance of the expectation-maximization algorithm [577]. First, the algorithm is initialized by setting $\hat{p}_U^{(0)}(\oplus) \leftarrow p_L(\oplus)$ as starting prevalence estimate. Then, two mutually recursive steps are iterated (for $s \geq 1$):

E-step: The posterior probability $\Pr(\oplus | \mathbf{h}_v^{(T)})$ is scaled by the ratio between the previous estimate $\hat{p}_U^{(s-1)}(\oplus)$ and the initial estimate $\hat{p}_U^{(0)}(\oplus)$, and re-normalized. This has the effect of tuning the posterior probabilities towards the current class prevalence estimate.

M-step: The current estimate $\hat{p}_U^{(s)}(\oplus)$ is produced by predicting U with the updated posterior and setting the average prediction as the new estimate. This has the effect of tuning the class prevalence estimate towards the rescaled posterior probabilities.

The process is repeated until convergence (which occurs when the estimate remains stable through successive iterations), at which point the final estimate $\hat{p}_U(\oplus)$ is returned.

Thorough experiments that validate the described method are presented in [578].

10.6.3. Conclusion

We have presented XNQ, a novel model tailored to the many challenges of the NQ task, which integrates randomized recursive graph neural networks, a customized calibrated readout for quantification, and a downstream powerful quantifier based on the Expectation-Maximization algorithm. Our extensive and thorough evaluation shows that it is possible to improve at this task by effectively exploiting the graph structure of the data and by scaling seamlessly to hundreds of thousands of nodes without being impaired by common issues of large-scale networks such as heterophily. These results establish XNQ at the forefront of NQ research and pave the way for its application to further real-world case studies.

Two limitations of our approach are that it only studies binary NQ, and that it relies on aggregative quantification methods, which can only be applied on top of trained (calibrated) classifiers. In our future research, we would like to extend our approach to the multi-class case, and to investigate *non-aggregative* network quantification methods, i.e., methods that estimate class priors without assigning labels to (or computing posterior probabilities for) individual nodes. Unlike the aggregative methods, non-aggregative ones have the additional advantage that no inference at the individual level is performed; this is desirable in some applications of quantification, as in measuring the fairness (i.e., absence of bias) of a model with respect to sensitive attributes [523].

10.6.4. Relevant publications

- Alessio Micheli, Alejandro Moreo, Marco Podda, Fabrizio Sebastiani, William Simoni, Domenico Tortorella. Learning to quantify graph nodes. **Proceedings of the 38th Annual**





Conference on Neural Information Processing System (NeurIPS 2024), Vancouver, US. Submitted for publication. [578]

10.6.5. Relevant software/datasets/other outcomes

- The code needed to reproduce all our experiments will be made publicly available in open-source mode once the paper is accepted for publication.

10.6.6. Relevance to AI4Media use cases and media industry applications

While this development does not play a role in any specific AI4Media use case (this being a very recent development), the solution proposed has potential for use in the media sector. One of the possible applications has to do with estimating the distribution of popular support for a given political candidate (or policy, or parliamentary bill) from blog or microblog posts issued by interconnected people, such as Facebook users.

10.7. Using quantification to predict classifier accuracy under prior probability shift

Contributing partner: CNR

10.7.1. Introduction

The standard way of predicting the accuracy of a classifier on unseen data is using k -fold cross-validation (k FCV). However, the accuracy estimates that k FCV returns are accurate only when the training data T and the unseen data U are IID, i.e., when no *dataset shift* [527] is present in the data. Unfortunately, dataset shift is ubiquitous in real-world applications, for a variety of reasons. One such reason is the possible non-stationarity of the environment across time and/or space and/or other variables, in which case the deployment conditions are irreproducible at training time. Another reason is the possible presence of sample selection bias in the training data, as when the process of labelling these data has introduced bias in them intentionally (e.g., when oversampling the minority class) or unintentionally (e.g., when using active learning). In all these cases, predicting the accuracy of a classifier on unseen data is problematic.

In this work we tackle the problem of predicting the accuracy of a classifier on unseen data affected by *prior probability shift* (PPS), an important type of dataset shift in which $P(X|Y)$, the class-conditional distribution of the covariates, is invariant across the training data and the test data, but $P(Y)$, the distribution of the class labels, is not. We propose a novel classifier accuracy prediction (CAP) method built on top of *quantification* algorithms [524] robust to PPS, i.e., algorithms whose task is predicting the prevalence values of the classes (i.e., the class priors) in samples of unseen data affected by PPS; we name this method QuAcc (“Quantification for Accuracy Prediction”). The key idea that underlies QuAcc is that of viewing the cells of the contingency table (on which classifier accuracy is computed) as classes, and estimating the prevalence of these classes via quantification algorithms. QuAcc can thus deal with any classifier accuracy measure defined on a contingency table.

We have run experiments in which we test QuAcc against state-of-the-art CAP methods on data generated by a robust experimental protocol, i.e., on settings characterised by different amounts of training data imbalance, test data imbalance, and PPS. In order to ensure the reproducibility of our experiments, we make available the code and the data on which they are based. While in this work we only discuss (and experiment on) the binary case, QuAcc straightforwardly extends to the multiclass case.





10.7.2. Methodology

QuAcc, the method we propose, comes in three different variants, that we describe in the next three subsections.

10.7.2.1. The 1×4 method. In the general multiclass case, most classifier accuracy measures can be computed from the values c_{ij}^U of a contingency table, where c_{ij}^U is the number of datapoints in test set U that belong to class y_i and that the classifier has assigned to class y_j . Of course, in operational situations the values c_{ij}^U are unknown, since the true labels of the datapoints in U are unknown. Our method for estimating $A(h, U)$ under PPS is based on the following idea:

1. View the cells of the contingency table $C = \{c_{11}, \dots, c_{nn}\}$ as a codeframe, i.e., consider each cell $c_{ij} \in C$ as a class. Note that $C \equiv \mathcal{Y} \times \mathcal{Y}$.
2. Train, on a labelled set V , a model that estimates the values c_{ij}^U .
3. Use the estimates \hat{c}_{ij}^U to compute $\hat{A}(h, U)$.

Step 2 can obviously be recast as training on V a model that estimates the prevalence values $p_U(c_{ij})$ of classes c_{ij} in U , since $c_{ij}^U = p_U(c_{ij}) \cdot |U|$. A key aspect of QuAcc is that it uses, for tackling this step, techniques from quantification, since most such techniques are indeed concerned with training estimators of the class prevalence values under PPS [524].

In order to carry out Steps 2 and 3, we represent the datapoints as pairs $(\tilde{\mathbf{x}}, \tilde{y})$. Here $\tilde{\mathbf{x}}$ is a vector

$$\tilde{\mathbf{x}} = (\mathbf{x}, \Pr(y_1|\mathbf{x}), \dots, \Pr(y_n|\mathbf{x}))$$

which incorporates (i) the original representation \mathbf{x} that classifier h has used, and (ii) the posterior probabilities $\Pr(y_i|\mathbf{x})$ that h has returned for \mathbf{x} . In other words, we train a quantifier q by providing it with all the information we have about \mathbf{x} that q might use to figure out which cell c_{ij} datapoint \mathbf{x} is likely to belong to. (Of course, the quantifier is not interested in individual datapoints *per se*, but is interested in them only insofar as they contribute to the distribution of U across the classes.)

In our pairs $(\tilde{\mathbf{x}}, \tilde{y})$, \tilde{y} is instead a label that ranges not on \mathcal{Y} but on C . When \tilde{y} is the label of a datapoint in V , this indicates that \tilde{y} is its true label. When \tilde{y} is the label of a datapoint in U , instead, \tilde{y} is unknown. While it is not our goal to guess \tilde{y} for this datapoint in particular, it is our goal to estimate, for each $y_i, y_j \in \mathcal{Y}$, the prevalence $p_U(c_{ij}) \equiv c_{ij}^U/|U|$ of datapoints $(\mathbf{x}, y_i) \in U$ such that $h(\mathbf{x}) = y_j$. It is for reaching this goal that we use a quantifier. Once we have estimated the prevalence in U of all $c_{ij} \in C$, by multiplying all these estimates by $|U|$ we obtain estimates of all counts c_{ij}^U , and we can thus estimate $A(h, U)$.

An important aspect of this method is that it works for any classifier accuracy measure A defined in terms of a contingency table, since it does not estimate measure A directly but estimates the values of the cells of the contingency table C on which measure A is based. Another important aspect of this method is that it is classifier-independent and quantifier-independent, i.e., it does not make any assumption on which method has been used for training h and on which method is to be used for training q .

In the binary case (to which we restrict our analysis in this work) in which $\mathcal{Y} = \{\oplus, \ominus\}$, the contingency table is $C = \{\text{TP}, \text{TN}, \text{FP}, \text{FN}\}$, and we need to train a single multiclass quantifier that operates on these four classes; we thus call this the 1×4 method.

10.7.2.2. The 2×2 method. A variant of the 1×4 method can be obtained by observing that, once we have applied classifier h to U , we already know the value of $|\text{TP} \cup \text{FP}|$ (resp., the value of $|\text{TN} \cup \text{FN}|$), since this is the number of datapoints in U which h has assigned to class \oplus (resp., to





class \ominus). This means that we can leverage this information and solve instead our CAP problem by training, instead of one 4-class quantifier, two binary quantifiers, i.e., one that estimates how the datapoints in $TP \cup FP$ are distributed across TP and FP, and one that estimates how the datapoints in $TN \cup FN$ are distributed across TN and FN. Since this method involves two binary quantifiers, we call it the *2×2 method*.

In the multiclass case, switching from the analogue of the 1×4 method to the analogue of the 2×2 method means switching from a single quantifier that operates on n^2 classes to n quantifiers that operate on n classes each.

10.7.2.3. The 1×3 method. A further variant of the 1×4 method can be obtained by observing that class FP (resp., FN) is likely to be largely composed by negative (resp., positive) examples that, while lying in the region that the classifier has assigned to class \oplus (resp., class \ominus), lie just across the separating surface. In other words, most of the false positives and most of the false negatives are likely to lie in two regions of space that both flank (from two different sides) the separating surface, i.e., are likely to lie in two *contiguous* regions of space. It may thus make sense to merge FP and FN into a single class $FP \cup FN$. This solution thus involves a single three-class quantifier (we thus call it the *1×3 method*) that needs to estimate how the datapoints are distributed across classes TP, TN, and $FP \cup FN$.

One potential disadvantage of this solution with respect to the two previous ones is that it does *not* work for all classifier accuracy measures; in particular, it does not work for measures A such that $|FP|$ and $|FN|$ contribute differently to A (one example are cost-sensitive accuracy measures, in which the cost of a false positive may be stipulated to be different from the cost of a false negative). However, this solution does work for important classifier accuracy measures such as, e.g., vanilla accuracy and F_1 , whose mathematical forms are such that knowing the value of $|FP \cup FN|$ suffices, and knowing the individual values of $|FP|$ and $|FN|$ is not strictly required.

10.7.2.4. Adding covariates. We also explore the impact of enriching the vectorial representations $\tilde{\mathbf{x}}$ with information potentially useful for the quantification process. Given a datapoint \mathbf{x} and the vector $\mathbf{p} = (p_1, \dots, p_n)$, where $p_i = \Pr(y_i|\mathbf{x})$ is the posterior probability generated by classifier h , we explore three additional covariates that make explicit to the quantifier some information derived from \mathbf{p} .

The *max conf* covariate is defined as

$$MC(\mathbf{p}) = \max_{i \in \{1, \dots, n\}} p_i$$

MC provides a measure of how confident h is in predicting the class label of \mathbf{x} : the higher the value, the greater the confidence. The confidence level is thus decoupled from the actual class y_i it is associated with, making it potentially easier for the quantifier to exploit this information.

By drawing inspiration from [579], we consider the *negative entropy* given by

$$NE(\mathbf{p}) = \sum_{i=1}^n p_i \log p_i$$

Both MC and NE reach their maximum values when $p_i = 1$ for some i , and attain their minimum values for the uniform distribution. However, the type of information each measure contributes is distinct; MC only accounts for the maximum value, whereas NE instead processes all values.

The additional covariate we explore is based on the softmax function s , that maps any real-valued vector $\mathbf{z} = (z_1, \dots, z_n)$ into a probability distribution $s(\mathbf{z}) \equiv \mathbf{p} = (p_1, \dots, p_n)$ where





$p_i = e^{z_i} \cdot (\sum_{j=1}^n e^{z_j})^{-1}$. We consider, as the basis for a new covariate, the values returned by the *inverse softmax* $s^{-1}(\mathbf{p}) = (z'_1, \dots, z'_n)$, where $z'_i = \log p_i + c$ and c is the logarithm of the normalisation factor $c = \log(\sum_{j=1}^n e^{z_j})$. Since c is undetermined (the actual values z_j are unknown), we set $c = -\frac{1}{n} \sum_{j=1}^n \log p_j$, thus centering the resulting values z'_i around zero. The rationale behind inverted softmax to \mathbf{p} is that of amplifying, in a non-linear way, the difference between low-confidence and high-confidence values. As with MC, we focus on the maximum value. The resulting *max inverse softmax* (MIS) covariate is thus given by

$$\text{MIS}(\mathbf{p}) = \max_{i \in \{1, \dots, n\}} \left(\log p_i - \frac{1}{n} \sum_{j=1}^n \log p_j \right)$$

Thorough experiments that validate the described method are presented in [580].

10.7.3. Conclusion

We have presented QuAcc, a new method for predicting classifier accuracy under prior probability shift (PPS), an important type of dataset shift. QuAcc is built on top of “quantification” methods robust to PPS, i.e., methods devised for estimating the class prevalence values in samples of unlabelled datapoints affected by PPS. QuAcc is based on the key intuition of viewing the cells of the contingency table, which is used for computing classifier accuracy, as classes, and of training a quantifier that estimates the values of these cells. The experiments we run on four large datasets simulate a wide variety (a) of amounts of training and/or test data imbalance, and (b) of amounts of PPS. The results of these experiments show that QuAcc systematically outperforms all four state-of-the-art baselines we have employed, exhibiting average levels of error reduction (with respect to the best such baseline) ranging from +5.52% to 75.93%. QuAcc is independent of the algorithm used for training the classifier, of the algorithm used for training the quantifier, and of the metric used for measuring classifier accuracy. While our experiments have concentrated on the binary case, we note that two of the QuAcc variants we have presented (the 1×4 method and the 2×2 method) extend straightforwardly to the multiclass case.

We plan to extend this research by carrying out systematic experiments in the multiclass case too, and by testing our algorithms on the more challenging scenario in which the data V on which the quantifier is trained and the data T on which the classifier was trained are related by PPS. Additional future research we intend to carry out concerns the problem of classifier accuracy prediction under types of dataset shift different from PPS. This might not necessarily mean devising methods alternative to QuAcc, but might mean instantiating QuAcc with quantification methods which have proven robust to types of dataset shift different from PPS. This may prove nontrivial, though, since most of the quantification literature has focused on PPS [540], somehow neglecting other types of dataset shift.

10.7.4. Relevant publications

- Lorenzo Volpi, Andrea Esuli, Alejandro Moreo, Fabrizio Sebastiani. Using quantification to predict classifier accuracy under prior probability shift. **Proceedings of the 41st International Conference on Machine Learning (ICML 2024)**, Vienna, AT. Submitted for publication. [580]





10.7.5. Relevant software/datasets/other outcomes

- The code needed to reproduce all our experiments will be made publicly available in open-source mode once the paper is accepted for publication.

10.7.6. Relevance to AI4media use cases and media industry applications

While this development does not play a role in any specific AI4Media use case, the solution proposed has a lot of potential for use in the media sector. One of the possible applications has to do with estimating the distribution of popular support for a given political candidate (or policy, or parliamentary bill) from blog or microblog posts, where support is specified on an ordinal scale (say, from 1 to 5 “stars”).

10.8. Other contributions related to Learning to Quantify (Task 3.7)

Contributing partner: CNR

10.8.1. Summary

Other contributions related to LQ made by CNR in the reporting period are the following:

- The proceedings of LQ 2023, the 3rd edition of the international workshop on LQ [581], have been published in open-access form (see <https://zenodo.org/records/8374012>). Two out of four co-editors (A. Moreo and F. Sebastiani) are with CNR. A report on the workshop [582] (see <https://zenodo.org/records/11277232>) has been published on the *SIGKDD Explorations* magazine by the four co-editors.
- LQ 2024 (<https://lq-2024.github.io/>), the 4th edition of the international workshop on LQ, has been organized, and will be co-located with ECML/PKDD 2024 (Vilnius, LT, September 2024 – <https://ecmlpkdd.org/2024/>). The LQ 2024 proceedings will be published in open-access form. Two out of four co-organizers (A. Moreo and F. Sebastiani) are with CNR.
- A half-day tutorial on LQ will be offered by two CNR researchers (A. Moreo and F. Sebastiani) and a researcher from University of Dortmund at ECML/PKDD 2024 (see above).
- LeQua 2024 (<https://lequa2024.github.io/>), the 2nd edition of the biennial LeQua data challenge centred on LQ, has been organized and is currently ongoing; its results will be presented at the LQ 2024 workshop (see above) and documented in the LQ 2024 proceedings. All of the LeQua 2024 organizers are with CNR.
- A 7 1/2-hour course on LQ was given by two CNR researchers (A. Moreo and F. Sebastiani) at the 2nd European Summer School on Artificial Intelligence (ESSAI 2024 – <https://essai2024.di.uoa.gr/>), Athens, GR, July 2024.
- Fabrizio Sebastiani (CNR) has given a talk titled “Predicting classifier accuracy in the wild” in the AI4Media Artificial Intelligence Doctoral Academy (AIDA) “AI Excellence Lectures” series, March 26, 2023.
- A practitioner-oriented survey on quantification has been submitted to the *Communications of the ACM* by two CNR researchers (A. Moreo and F. Sebastiani).
- An extended version of a previously published conference paper [583] on ordinal quantification has been accepted for publication in the *Data Mining and Knowledge Discovery* journal [584]. Two of the four authors (A. Moreo and F. Sebastiani) are with CNR.





10.8.2. Relevant publications

- Mirko Bunse, Pablo González, Alejandro Moreo, and Fabrizio Sebastiani. Report on the 3rd International Workshop on Learning to Quantify (LQ 2023). **SIGKDD Explorations** 25(2), 2023. [582] This paper appears on Zenodo at <https://zenodo.org/records/11277232>.
- Mirko Bunse, Pablo González, Alejandro Moreo, and Fabrizio Sebastiani (eds.). **Proceedings of the 3rd International Workshop on Learning to Quantify (LQ 2023)**, Torino, IT, 2023. [585] These proceedings appear on Zenodo at <https://zenodo.org/records/8374012>.

10.8.3. Relevant software/datasets/other outcomes

- The code needed to reproduce all the experiments in [584] is available from <https://github.com/mirkobunse/regularized-oq>.





11. Quantum Reinforcement Learning (Task 3.8)

Contributing partners: BSC

In this section we introduce the results obtained with a combination of techniques using resources from both conventional and Quantum computation. On the Quantum side, we are restricted to a collection of small devices, with a low qubit count and shallow circuits. This is a realistic scenario considering today's Quantum devices. On the classical side, we do not impose any restrictions.

11.1. Quantum circuit depth optimization using RL

Contributing partner: BSC

11.1.1. Introduction and methodology

Current quantum devices face significant challenges, such as the presence of noise and decoherence in the physical systems that implement quantum circuits. These challenges limit the scalability and the reliability of quantum computation, and pose a major obstacle for achieving quantum advantage over classical computation. Therefore, it is essential to design and optimize quantum circuits in a way that minimizes the number of gates and the resources required, while preserving the functionality and the fidelity of the computation.

One of the common approaches for optimizing quantum circuits is to apply algebraic identities to perform gate permutations and gate cancellations in the original circuit. Using reinforcement learning (RL) in combination with this approach is currently being explored with promising results. However, in the previous context, the action space for the RL agent grows quickly, as there are several types of gate identities that need to be identified and each of those may involve multiple gates. This makes it harder for reinforcement learning agents to explore and exploit the optimal actions, as they have to deal with a large and diverse set of possible gate permutations and cancellations.

In this work, we incorporate RL and, more specifically, the Proximal Policy Optimization algorithm to guide the optimization of quantum circuits through the ZX formalism. We define a reward function that reflects the quality of the circuit optimization and explore the space of possible transformation rules. Before this work, convolutional neural networks were used to learn the ZX-Calculus rewrite rules [586], and the method was shown to improve the existing ZX Calculus based optimization algorithms implemented in the PyZX Python package for small circuits. However, some limitations concerning the use of convolutional neural networks were identified, such as the difficulty of handling variable-sized inputs and outputs, heavily limiting the scalability of the approach.

11.1.2. Methodology

ZX-diagram allows a graphical representation of a linear map between the basic components of a Quantum circuit –namely its qubits– by means of an undirected graph [587, 588]. The basic elements of a ZX-diagram are a set of operators named spiders (nodes) and wires (edges). Spiders can be of two types: Z and X, and they can be interpreted as tensors composed of Pauli-Z and Pauli-X operators eigenstates, respectively. The wires in the diagram can also be of two types, typically referred to as Simple and Hadamard wires. A ZX-diagram can be simplified by applying a set of rules that preserve the underlying tensor representation of the diagram.

Quantum circuit optimization via ZX-Calculus involves the following steps: First, one should transform the quantum circuit into its equivalent ZX-diagram. This diagram is then converted into





its *graph-like* form and simplified using graph-theoretic rules. After the simplification process is finished, one needs to transform the diagram back into an equivalent quantum circuit. This last step can be very inefficient or even unfeasible in some cases [589]. Even in the cases where this step is efficient, it can output circuits that are more computationally-expensive than the initial ones.

In this work, we use the *Proximal Policy Optimization* Reinforcement Learning algorithm (PPO), which has a positive track record in similar circuit optimization settings. The PPO algorithm is a policy-gradient method which relies on the optimization of the parameters of a policy function $\pi(a|s)$. This function returns the optimal action to be performed given an observation. Additionally, the agent guides its learning process by interpolating a value function that estimates the expected value of the returns for a given state, $V(s)$. Both $\pi(a|s)$ and $V(s)$ functions are typically approximated using Deep Neural Networks (DNNs). The DNN used for approximating the policy function is often referred to as the *actor*, as it determines the optimal action in a given state. On the other hand, the DNN used for approximating the value function is known as the *critic*, as it evaluates the expected returns for a given state. This actor-critic architecture is a common approach in RL algorithms.

11.1.3. Experimental results

We analyze the impact of this optimization strategy studying the reduction in size of a collection of Quantum circuits. We focus on the reduction of the depth of the Quantum circuit, a scarce resource in today's Quantum computers. We test the obtained agent policies on 1000 random Quantum circuits of increasing number of qubits (5, 10, 20, 40 and 80) and initial increasing average depth (d, 2d, 3d, 4d). For clarity, the largest instances are generated for 80 qubit circuits of 2100 random gates (which result on an average depth of 80). The probabilities of inclusion of each gate remain unchanged with respect to the training phase.

When optimizing the total amount of gates in the circuit, the agent consistently increases the amount of simplified gates across both circuit depth and number of qubits (Figure 37a). Naively, this signals that the agent is able to generalize correctly. On the other hand, the agent's performance seems to be mostly dependent on circuit depth when trying to reduce the amount of two-qubit gates (Figure 37b).

11.1.4. Conclusion

This work is an exploratory work to develop a novel Reinforcement Learning approach for quantum circuit optimization that exploits the advantages of ZX-Calculus. Here, we improve on our initial methodology by using a more sophisticated scheme based on Graph NNs instead of convolutional layers. To assess the validity of the method, we present results across three relevant criteria: quality of the optimization, computational efficiency and scalability. These results are benchmarked for universal circuits against the best-performing ZX-Calculus based circuit optimization algorithm across two differentiated optimization objectives, the total amount of gates in the circuit and two-qubit gates alone. We demonstrate that the agent is able to generalize the learn strategies and outperform the competition for circuits of up to 80 qubits and 2,100 gates for both tasks. This versatility is particularly useful for current experimental Quantum platforms, as the reward function can be shaped taking into account the specific properties of the quantum hardware in which the circuit will be executed, e.g., by weighting each type of gate depending on its fidelity, or taking into account qubit coherence times and the depth of the circuit.

11.1.5. Relevant publications

- "Reinforcement Learning Based Quantum Circuit Optimization via ZX-Calculus", Jordi Riu, Jan Nogué, Gerard Vilaplana, Artur Garcia-Saez, Marta P. Estarellas, arXiv preprint





<https://arxiv.org/abs/2312.11597>.

11.1.6. Relevance to AI4Media use cases and media industry applications

Current applications of Quantum circuits for media processing include image manipulation and classification, among other uses. Algorithms using image processing primitives use a low level representation using Quantum circuits, the basic components of Quantum algorithms. A paradigmatic example is image classification for the detection of anomalies in images (i.e. used to detect image manipulation).

The techniques introduced here allow an efficient implementation of these algorithms. As current Quantum devices are very limited in the depth and size of the Quantum circuits that can be implemented, a reduction in their size means a great advantage in the realistic implementation of these methods, with a potential impact in image processing applications. With these reduction in the resource count, these tools could be integrated sooner in the form of products or applications, using the same Quantum devices.



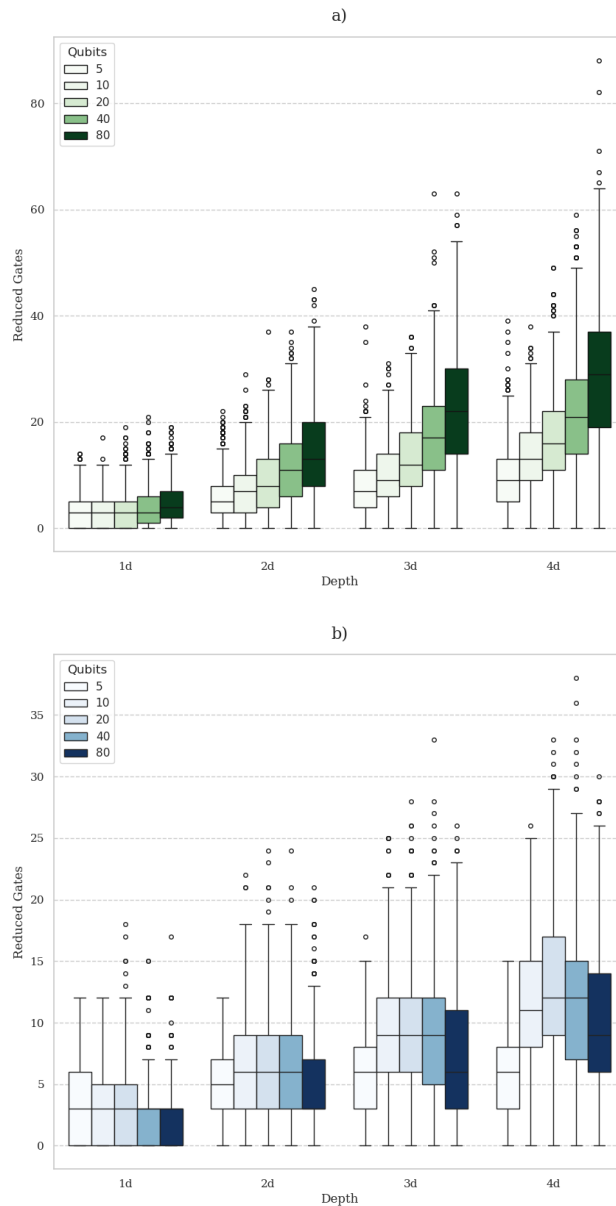


Figure 37. Box plot of the number of gates reduced by the agent once trained. Tests are performed for two-different tasks and across several circuit sizes, both in terms of number of qubits and average circuit depth. Statistics are drawn from one-thousand executions of random circuits for each circuit size and task. (a) Reduced single-qubit and two-qubit gates. Different shades of green depict different amount of qubits. (b) Number of two-qubit gates reduced. Different shades of blue depict different amount of qubits.





12. Ongoing Work and Conclusions

12.1. Ongoing work

Below, we briefly summarize the ongoing work associated to each task.

12.1.1. Lifelong and on-line learning (Task 3.1)

CEA will be focusing on multimodal continual learning based on large multimedia models. One stream of work will focus on their continual update to integrate new data swiftly. Another research direction will investigate adapted large model compression for continual learning.

To meet the recent decentralization trend in the community, **UNITN** is investigating a practical yet challenging task, Federated GCD (Fed-GCD), where the training data are distributed among local clients and cannot be shared among clients. Fed-GCD aims to train a generic GCD model by client collaboration under the privacy-protected constraint. The Fed-GCD leads to two challenges: 1) representation degradation caused by training each client model with fewer data than centralized GCD learning, and 2) highly heterogeneous label spaces across different clients.

AUTH will extend its research in collaborative knowledge distillation frameworks by focusing on fast DNN agent adaptation in dynamic environments using learning-by-education methods. Furthermore, LENC environment will be ported to decentralized (cloud) computing environment using container technologies. Finally, its scope will be expanded to be used in other tasks, *e.g.* decentralized inference, regression and image segmentation.

12.1.2. Manifold learning and disentangled feature representation (Task 3.2)

QMUL will be focusing on Large Multimodal and Large Language Models and on ways to disentangle representations at different layers so as to provide ways of controlling and interpreting their behavior.

UNITN is currently looking into the use of deep neural networks for learning Symmetric Positive Definite (SPD) matrices. Despite the significant progress, most existing SPD networks use traditional Euclidean classifiers on an approximated space rather than intrinsic classifiers that accurately capture the geometry of SPD manifolds. Inspired by Hyperbolic Neural Networks (HNNs), we consider the use of Riemannian Multinomial Logistics Regression (RMLR) for the classification layers in SPD networks.

12.1.3. Transfer learning (Task 3.3)

UNITN is currently tackling the challenge of open-set bias detection in text-to-image generative models. A new pipeline that identifies and quantifies the severity of biases agnostically, without access to any precompiled set is under investigation. In the first phase, the idea is to leverage a Large Language Model (LLM) to propose biases given a set of captions. Secondly, the target generative model should produce images using the same set of captions. Lastly, a Vision Question Answering model is employed to recognize the presence and extent of the previously proposed biases.

12.1.4. Neural Architecture Search (Task 3.4)

UNITN is preparing the extension of the work presented in Section 7.1 to be submitted to the Pattern Recognition journal. There are several innovations: two new parameter-sharing loss functions, Union and Jaccard, are considered. Also, a new strategy to enable learning the importance





weight for these losses as parameters of the model is investigated. Finally, a novel performance measure, the efficiency-effectiveness score for multi-domain learning, is introduced. This allows to select the methods with the best trade-off between effectiveness on the classification task and the efficiency in computational complexity and number of parameters.

12.1.5. AI at the Edge, decentralised and distributed learning (Task 3.5)

JR will port the AdaFamily algorithm to a distributed training framework like *DeepSpeed* or *Colossal-AI*. The combined formulation allows for a faster porting of the algorithm to a distributed training framework, as only one Algorithm (AdaFamily) has to be ported instead of multiple ones. With respect to the porting LLMs to mobile devices, JR will explore recently introduced LLMs like *phi-2* and GPU acceleration of the model via OpenCL or Vulkan on the device.

CERTH will investigate the performance of the genetic algorithms presented in Section 8.3 in larger-scale federated learning scenarios. This will involve significantly increasing the number of clients and utilizing more complex, diverse datasets. Additionally, a comprehensive study of the algorithms' behavior under varying degrees of data heterogeneity will be conducted, to create a spectrum of non-IID scenarios, ranging from mildly heterogeneous to extremely skewed data distributions across clients. Analyzing the algorithms' adaptability and performance across this spectrum will offer valuable insights for deploying these methods in diverse real-world federated learning applications, where data distributions often vary widely among participants.

12.1.6. Deep quality diversity (Task 3.6)

UM is continuing its work on expanding *CrawLLM* for automated game generation by incorporating deep quality diversity search using the *MELiTA* algorithm. This will allow us to test the algorithm on a more complex test case, with more dimensions requiring orchestration such as multiple types visuals, sound, and game narrative.

12.1.7. Learning to count (Task 3.7)

CNR researchers have a number of ongoing research works concerning learning to quantify. One such work concerns three new algorithms, all very different from each other, for estimating the accuracy of classifiers under dataset shift; these algorithms, all developed by CNR, are currently undergoing thorough experimentation. Another ongoing effort concerns the development of new algorithms for hierarchical quantification, i.e., for the case in which the classes on which quantification needs to be performed are organised in a taxonomy.

12.1.8. Quantum Reinforcement Learning (Task 3.8)

The work developed along the project will continue by the researchers at **BSC**. The implementation of some of the methods presented here on real quantum hardware will allow to test these algorithms in realistic scenarios. Besides this approach, some methods have been extended to other problems in computer vision or simulation, using the same tools under different conditions to check their flexibility.

12.2. Conclusions

In this deliverable, we presented the research results obtained in all WP3 tasks in the last period. The contributions addressed a large spectrum of open challenges associated with new learning





paradigms. They led to high-quality outcomes, as highlighted by the numerous publications in reputed venues discussed in the deliverable.

In Task 3.1, several new methodologies developing partners' existing work on NCD, CIL, representation stationarity, and a teacher-student network education are presented. We also study the impact of recent developments on learning paradigms, such as foundation models and generative AI, on lifelong learning. The presented works are particularly relevant to the use cases of AI4Media since they can significantly reduce catastrophic forgetting in a scenario with several updates being rehearsal-free (i.e., no episodic memory), and where a gallery's features in a visual search system do not require re-computed (re-indexed) when the model is updated in a lifelong learning scenario.

In Task 3.2, the partners have developed methodologies for analyzing the latent representations of Deep Learning models so as to finding meaningful directions/projections. The proposed methods offer insights on how the modeling leads to better understanding of properties such as disentanglement and equivariance, and how discovered directions can be used in various important applications in the domain of media. This includes controlling generation of synthetic images, suppressing the ability of the model to generate inappropriate content and augmenting datasets so as to increase the fairness of models that are trained on them.

In Task 3.3, focus was put on test-time adaptation for semantic segmentation. The proposed approach is effective since the obtained performance is competitive and efficient since it has a low computational cost. The resulting semantic segmentation method can be used by media companies to obtain a fine-grained description of their visual content, and thus improve the access to and understanding of the content they handle.

In Task 3.4, the research addresses the multi-domain learning problem while taking into account a user-defined budget for computational resources. The solution is to prune a single model for multiple domains, making it more compact and efficient. The presented results are competitive with other state-of-the-art methods while offering good trade-offs between classification performance and computational cost according to the user's needs. The presented work is particularly relevant to the use cases of AI4Media since the research addresses the multi-domain learning problem while taking into account a user-defined budget for computational resources, a scenario of vital importance for devices with limited computational power.

Task 3.5 has investigated the application of AI models on the edge, including end user devices and servers, a paradigm that is gaining momentum due to the privacy requirements of user data and the increasing costs of centralizing resources at cloud infrastructures. Acknowledging the computational and storage constraints of edge devices, along with the cost of communication, Task 3.5 has developed solutions along three axes: a) collaborative training paradigms like federated and gossip learning, which allow devices to cooperate in model training without sharing their private data directly, b) model compression techniques, which allow devices to run AI models locally with small storage and acceptable response time, and c) technical tools that facilitate the deployment of AI on the edge, for example, a toolchain for the local execution of LLMs. Overall, Task 3.5 has led to important contributions towards more cost-effective AI that can be deployed on the edge and in decentralized settings.

The main contribution for T3.6 focuses on the MELiTA algorithm, which extends the MAP-Elites framework for generating multimodal artefacts using deep QD and a novel inter-model evaluation process which enhances coherence in generated outputs. Our first steps in expanding MELiTA involve using LLMs for automated game generation, which we call CrawLLM, to re-theme games, and eventually orchestrate generating game content across multiple modalities. Additional research directions include dynamic QD, which introduces a new framework for QD search in dynamic environments, ensuring optimal solutions despite environmental shifts. Lastly, constrained QD search for shell structures using the FI-MAP-Elites algorithm offers a broader range of feasible design alternatives, enhancing the conceptual design phase in engineering.





Learning to quantify (T3.7) is a machine learning task that is receiving increased interest from the scientific community, due to the fact that its attention to aggregated data, as opposed to individual data items, makes it attractive for a number of applications, in the media sector and outside of it, in scenarios in which inferences at the individual level are either unnecessary or even undesired. AI4Media researchers have been at the forefront of these developments, devising novel methods for learning to quantify, novel protocols for testing quantification methods, and novel applications of quantification to important tasks, such as estimating classifier accuracy on out-of-distribution data, estimating the fairness of classifiers, and estimating the fairness of rankers.

Using Quantum hardware and implemented algorithms intended for these devices, we study (T3.8) the capabilities that this novel technology may bring to the processing of media in different formats. We have explored the capabilities of Reinforcement Learning in the control and optimization of Quantum hardware, using a mathematical formulation known as ZX-calculus. As current devices are limited in their performance, this approach may allow the optimal use of these resources in larger datasets that currently possible, helping the development of the field. As a corollary of our results, We are currently exploring methods to processing of data sets formed by images.

In summary, the activity during the last reporting period kept up with the high standards established during the first three years of the project. It led to the publication of **19 conference papers** and **5 journal articles**, accompanied by **open-source source software** in most cases. The high-quality outcomes of the period reflect the active involvement of all partners toward achieving the WP3 objectives and contributing to the global AI4Media goals. While the project reaches its end, the WP3 work paves the way to future contributions to new learning paradigms for AI, applicable to the media domain and beyond.





References

- [1] C. Burgess and H. Kim, “3d shapes dataset.” <https://github.com/deepmind/3dshapes-dataset/>, 2018.
- [2] Z. Wang, K. Qinami, I. C. Karakozis, K. Genova, P. Nair, K. Hata, and O. Russakovsky, “Towards fairness in visual recognition: Effective strategies for bias mitigation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8919–8928, 2020.
- [3] J. Nam, H. Cha, S. Ahn, J. Lee, and J. Shin, “Learning from failure: Training debiased classifier from biased classifier,” in *Advances in Neural Information Processing Systems*, 2020.
- [4] S. Jung, S. Chun, and T. Moon, “Learning fair classifiers with partially annotated group labels,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10348–10357, 2022.
- [5] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, “Cats and dogs,” in *CVPR*, 2012.
- [6] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, “Emerging properties in self-supervised vision transformers,” in *ICCV*, 2021.
- [7] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, “Generalized category discovery,” in *CVPR*, 2022.
- [8] D. Goswami, Y. Liu, B. Twardowski, and J. van de Weijer, “Fecam: Exploiting the heterogeneity of class distributions in exemplar-free continual learning,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [9] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” 2017.
- [10] L. Deng, “The mnist database of handwritten digit images for machine learning research [best of the web],” *IEEE signal processing magazine*, vol. 29, no. 6, pp. 141–142, 2012.
- [11] K. Preechakul, N. Chatthee, S. Wizadwongsa, and S. Suwajanakorn, “Diffusion autoencoders: Toward a meaningful and decodable representation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [12] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
- [13] K. Karkkainen and J. Joo, “Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1548–1558, 2021.
- [14] M. D’Incà, C. Tzelepis, I. Patras, and N. Sebe, “Improving fairness using vision-language driven image augmentation,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4695–4704, 2024.
- [15] J. Oldfield, C. Tzelepis, Y. Panagakis, M. A. Nicolaou, and I. Patras, “Parts of speech-grounded subspaces in vision-language models,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.





- [16] Y. Song, A. Keller, N. Sebe, and M. Welling, “Flow factorized representation learning,” in *Neural Information Processing Systems*, 2023.
- [17] V. Marx, “The big challenges of big data,” *Nature*, vol. 498, no. 7453, pp. 255–260, 2013.
- [18] H. R. Varian, “Beyond big data,” *Business Economics*, vol. 49, no. 1, pp. 27–31, 2014.
- [19] A. Taherkordi, F. Eliassen, and G. Horn, “From iot big data to iot big services,” in *Proceedings of the Symposium on Applied Computing*, pp. 485–491, 2017.
- [20] H. Liu, “Beyond the scale of big data,” *Frontiers in big Data*, vol. 1, p. 1, 2018.
- [21] A. Zaslavsky, C. Perera, and D. Georgakopoulos, “Sensing as a service and big data,” *arXiv preprint arXiv:1301.0159*, 2013.
- [22] R. M. Alguliyev, R. T. Gasimova, and R. N. Abbashi, “The obstacles in big data process,” *International Journal of Modern Education & Computer Science*, vol. 9, no. 3, 2017.
- [23] A. C. Djedouboum, A. A. Abba Ari, A. M. Gueroui, A. Mohamadou, and Z. Aliouat, “Big data collection in large-scale wireless sensor networks,” *Sensors*, vol. 18, no. 12, p. 4474, 2018.
- [24] M.-L. Song, R. Fisher, J.-L. Wang, and L.-B. Cui, “Environmental performance evaluation with big data: Theories and methods,” *Annals of Operations Research*, vol. 270, no. 1, pp. 459–472, 2018.
- [25] J. K. Pugh, L. B. Soros, and K. O. Stanley, “Quality diversity: A new frontier for evolutionary computation,” *Frontiers in Robotics and AI*, vol. 3, 2016.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, 2016.
- [27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, 2017.
- [28] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *ICLR*, 2015.
- [29] A. Oliver, A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow, “Realistic evaluation of deep semi-supervised learning algorithms,” *NeurIPS*, 2018.
- [30] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, “Fixmatch: Simplifying semi-supervised learning with consistency and confidence,” *NeurIPS*, 2020.
- [31] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, “S4l: Self-supervised semi-supervised learning,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1476–1485, 2019.
- [32] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum contrast for unsupervised visual representation learning,” in *CVPR*, 2020.
- [33] H.-X. Yu and W.-S. Zheng, “Weakly supervised discriminative feature learning with state information for person identification,” in *CVPR*, 2020.





- [34] M. Zheng, F. Wang, S. You, C. Qian, C. Zhang, X. Wang, and C. Xu, “Weakly supervised contrastive learning,” in *ICCV*, 2021.
- [35] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” *NeurIPS*, 2017.
- [36] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, “Generalizing from a few examples: A survey on few-shot learning,” *ACM computing surveys (csur)*, vol. 53, no. 3, pp. 1–34, 2020.
- [37] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, “Toward open set recognition,” *IEEE TPAMI*, 2012.
- [38] L. Yi, S. Liu, Q. She, A. I. McLeod, and B. Wang, “On learning contrastive representations for learning with noisy labels,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16682–16691, 2022.
- [39] K. Han, A. Vedaldi, and A. Zisserman, “Learning to discover novel visual categories via deep transfer clustering,” in *CVPR*, 2019.
- [40] E. Fini, E. Sangineto, S. Lathuilière, Z. Zhong, M. Nabi, and E. Ricci, “A unified objective for novel class discovery,” in *ICCV*, 2021.
- [41] J. MacQueen, “Classification and analysis of multivariate observations,” in *5th Berkeley Symp. Math. Statist. Probability*, 1967.
- [42] A. Krizhevsky, G. Hinton, *et al.*, “Learning multiple layers of features from tiny images,” 2009.
- [43] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *CVPR*, 2009.
- [44] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” 2011.
- [45] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, “3d object representations for fine-grained categorization,” in *ICCVW*, 2013.
- [46] K. Han, S.-A. Rebuffi, S. Ehrhardt, A. Vedaldi, and A. Zisserman, “Autonovel: Automatically discovering and learning novel visual categories,” *IEEE TPAMI*, 2021.
- [47] N. Pu, Z. Zhong, and N. Sebe, “Dynamic conceptual contrastive learning for generalized category discovery,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [48] S. Thrun, “A lifelong learning perspective for mobile robot control,” in *Intelligent robots and systems*, pp. 201–214, Elsevier, 1995.
- [49] M. B. Ring, “Child: A first step towards continual learning,” *Machine Learning*, vol. 28, no. 1, pp. 77–104, 1997.
- [50] G. M. van de Ven, T. Tuytelaars, and A. S. Tolias, “Three types of incremental learning,” *Nature Machine Intelligence*, vol. 4, no. 12, pp. 1185–1197, 2022.
- [51] E. Belouadah, A. Popescu, and I. Kanellos, “A comprehensive study of class incremental learning algorithms for visual tasks,” *Neural Networks*, vol. 135, pp. 38–54, 2021.



- [52] M. Masana, X. Liu, B. Twardowski, M. Menta, A. D. Bagdanov, and J. van de Weijer, “Class-incremental learning: survey and performance evaluation on image classification,” 2021.
- [53] Z. Li and D. Hoiem, “Learning without forgetting,” in *European Conference on Computer Vision*, ECCV, 2016.
- [54] L. Pellegrini, V. Lomonaco, G. Graffieti, and D. Maltoni, “Continual learning at the edge: Real-time training on smartphone devices,” *arXiv preprint arXiv:2105.13127*, 2021.
- [55] T. L. Hayes and C. Kanan, “Online continual learning for embedded devices,” in *Conference on Lifelong Learning Agents*, pp. 744–766, PMLR, 2022.
- [56] T. Verma, L. Jin, J. Zhou, J. Huang, M. Tan, B. C. M. Choong, T. F. Tan, F. Gao, X. Xu, D. S. Ting, *et al.*, “Privacy-preserving continual learning methods for medical image classification: a comparative analysis,” *Frontiers in Medicine*, vol. 10, 2023.
- [57] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, “Dinov2: Learning robust visual features without supervision,” 2023.
- [58] G. Petit, M. Soumm, E. Feillet, A. Popescu, B. Delezoide, D. Picard, and C. Hudelot, “An analysis of initial training strategies for exemplar-free class-incremental learning,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2024.
- [59] F. Zhu, Z. Cheng, X.-y. Zhang, and C.-l. Liu, “Class-incremental learning via dual augmentation,” *Advances in Neural Information Processing Systems*, vol. 34, 2021.
- [60] F. Zhu, X.-Y. Zhang, C. Wang, F. Yin, and C.-L. Liu, “Prototype augmentation and self-supervision for incremental learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5871–5880, 2021.
- [61] K. Zhu, W. Zhai, Y. Cao, J. Luo, and Z.-J. Zha, “Self-sustaining representation expansion for non-exemplar class-incremental learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9296–9305, 2022.
- [62] E. Feillet, G. Petit, A. Popescu, M. Reyboz, and C. Hudelot, “Advisil - a class-incremental learning advisor,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2400–2409, 2023.
- [63] K.-Y. Lee, Y. Zhong, and Y.-X. Wang, “Do pre-trained models benefit equally in continual learning?,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 6485–6493, January 2023.
- [64] O. Ostapenko, T. Lesort, P. Rodriguez, M. R. Arefin, A. Douillard, I. Rish, and L. Charlin, “Continual learning with foundation models: An empirical study of latent replay,” in *Proceedings of The 1st Conference on Lifelong Learning Agents* (S. Chandar, R. Pascanu, and D. Precup, eds.), vol. 199 of *Proceedings of Machine Learning Research*, pp. 60–91, PMLR, 22–24 Aug 2022.



- [65] K. Wu, H. Peng, Z. Zhou, B. Xiao, M. Liu, L. Yuan, H. Xuan, M. Valenzuela, X. S. Chen, X. Wang, H. Chao, and H. Hu, “Tinyclip: Clip distillation via affinity mimicking and weight inheritance,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21970–21980, October 2023.
- [66] T. L. Hayes and C. Kanan, “Lifelong machine learning with deep streaming linear discriminant analysis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 220–221, 2020.
- [67] Z. Wang, Z. Zhang, C.-Y. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. Dy, and T. Pfister, “Learning to prompt for continual learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, 2022.
- [68] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” in *International conference on machine learning*, pp. 2256–2265, PMLR, 2015.
- [69] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [70] M. D. McDonnell, D. Gong, E. Abbasnejad, and A. v. d. Hengel, “Premonition: Using generative models to preempt future data changes in continual learning,” *arXiv preprint arXiv:2403.07356*, 2024.
- [71] Q. Jodelet, X. Liu, Y. J. Phua, and T. Murata, “Future-proofing class incremental learning,” *arXiv preprint arXiv:2404.03200*, 2024.
- [72] M. Seo, D. Misra, S. Cho, M. Lee, and J. Choi, “Just say the name: Online continual learning with category names only via data generation,” *arXiv preprint arXiv:2403.10853*, 2024.
- [73] J. Gallardo, T. L. Hayes, and C. Kanan, “Self-supervised training enhances online continual learning,” in *British Machine Vision Conference (BMVC)*, 2020.
- [74] T. L. Hayes, K. Kaffle, R. Shrestha, M. Acharya, and C. Kanan, “Remind your neural network to prevent catastrophic forgetting,” in *European Conference on Computer Vision*, pp. 466–483, Springer, 2020.
- [75] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, “End-to-end incremental learning,” in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII*, pp. 241–257, 2018.
- [76] E. Belouadah and A. Popescu, “Deesil: Deep-shallow incremental learning,” in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.
- [77] M. B. Sariyıldız, K. Alahari, D. Larlus, and Y. Kalantidis, “Fake it till you make it: Learning transferable representations from synthetic imagenet clones,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8011–8021, 2023.
- [78] D. Yi, “Learning face representation from scratch,” *CoRR*, vol. 1411, p. 7923, 2014.
- [79] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, “Large-scale image retrieval with attentive deep local features,” in *ICCV*, pp. 3476–3485, IEEE Computer Society, 2017.



- [80] G. Van Horn, O. Mac Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, and S. Belongie, “The inaturalist species classification and detection dataset,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8769–8778, 2018.
- [81] S. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, “icarl: Incremental classifier and representation learning,” in *Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.
- [82] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, “Learning a unified classifier incrementally via rebalancing,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 831–839, 2019.
- [83] P. Janson, W. Zhang, R. Aljundi, and M. Elhoseiny, “A simple baseline that questions the use of pretrained-models in continual learning,” in *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.
- [84] D.-W. Zhou, H.-J. Ye, D.-C. Zhan, and Z. Liu, “Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need,” *arXiv preprint arXiv:2303.07338*, 2023.
- [85] G. Petit, A. Popescu, H. Schindler, D. Picard, and B. Delezoide, “Fetrl: Feature translation for exemplar-free class-incremental learning,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023.
- [86] M. B. Sariyildiz, Y. Kalantidis, K. Alahari, and D. Larlus, “No reason for no supervision: Improved generalization in supervised models,” in *ICLR 2023-International Conference on Learning Representations*, pp. 1–26, 2023.
- [87] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, *et al.*, “Llama: Open and efficient foundation language models,” *arXiv preprint arXiv:2302.13971*, 2023.
- [88] M. Kim, F. Liu, A. Jain, and X. Liu, “Dcfac: Synthetic face generation with dual condition diffusion model,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12715–12725, 2023.
- [89] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran Associates, Inc., 2019.
- [90] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Conference on Computer Vision and Pattern Recognition, CVPR*, 2016.
- [91] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, “Imagenet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [92] C. Schuhmann, R. Vencu, R. Beaumont, R. Kaczmarczyk, C. Mullis, A. Katta, T. Coombes, J. Jitsev, and A. Komatsuzaki, “Laion-400m: Open dataset of clip-filtered 400 million image-text pairs,” *arXiv preprint arXiv:2111.02114*, 2021.

- [93] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, “Learning transferable visual models from natural language supervision,” in *ICML*, 2021.
- [94] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [95] A. Steiner, A. Kolesnikov, X. Zhai, R. Wightman, J. Uszkoreit, and L. Beyer, “How to train your vit? data, augmentation, and regularization in vision transformers,” *arXiv preprint arXiv:2106.10270*, 2021.
- [96] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255, 2009.
- [97] M. D. McDonnell, D. Gong, A. Parvaneh, E. Abbasnejad, and A. van den Hengel, “Ranpac: Random projections and pre-trained models for continual learning,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [98] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 1, pp. 539–546, IEEE, 2005.
- [99] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE T-PAMI*, 2013.
- [100] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “Cnn features off-the-shelf: an astounding baseline for recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 806–813, 2014.
- [101] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” in *Advances in Neural Information Processing Systems (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, eds.)*, vol. 27, Curran Associates, Inc., 2014.
- [102] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1701–1708, 2014.
- [103] Y. Sun, D. Liang, X. Wang, and X. Tang, “DeepID3: Face recognition with very deep neural networks,” *arXiv preprint arXiv:1502.00873*, 2015.
- [104] R. Ranjan, C. D. Castillo, and R. Chellappa, “L2-constrained softmax loss for discriminative face verification,” *arXiv preprint arXiv:1703.09507*, 2017.
- [105] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2019.
- [106] Q. Meng, S. Zhao, Z. Huang, and F. Zhou, “MagFace: A universal representation for face recognition and quality assessment,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.



- [107] Y. Sun, L. Zheng, Y. Yang, Q. Tian, and S. Wang, “Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline),” in *European Conference on Computer Vision*, 2018.
- [108] A. Hermans, L. Beyer, and B. Leibe, “In defense of the triplet loss for person re-identification,” *arXiv preprint arXiv:1703.07737*, 2017.
- [109] E. Ristani and C. Tomasi, “Features for multi-target multi-camera tracking and re-identification,” in *European Conference on Computer Vision*, pp. 6036–6046, 2018.
- [110] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, “Revisiting oxford and paris: Large-scale image retrieval benchmarking,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5706–5715, 2018.
- [111] F. Radenović, G. Tolias, and O. Chum, “Fine-tuning cnn image retrieval with no human annotation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 7, pp. 1655–1668, 2018.
- [112] W. Chen, Y. Liu, W. Wang, E. M. Bakker, T. Georgiou, P. Fieguth, L. Liu, and M. S. Lew, “Deep learning for instance retrieval: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- [113] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, “Language models are few-shot learners,” *NeurIPS*, 2020.
- [114] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, “Learning transferable visual models from natural language supervision,” in *International conference on machine learning*, pp. 8748–8763, PMLR, 2021.
- [115] L. Caccia, R. Aljundi, N. Asadi, T. Tuytelaars, J. Pineau, and E. Belilovsky, “New insights on reducing abrupt representation change in online continual learning,” in *International Conference on Learning Representations*, 2021.
- [116] M. Davari, N. Asadi, S. Mudur, R. Aljundi, and E. Belilovsky, “Probing representation forgetting in supervised and unsupervised continual learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16712–16721, June 2022.
- [117] T. Barletti, N. Biondi, F. Pernici, M. Bruni, and A. Del Bimbo, “Contrastive supervised distillation for continual representation learning,” *International Conference on Image Analysis and Processing*, 2022.
- [118] N. Asadi, M. Davari, S. Mudur, R. Aljundi, and E. Belilovsky, “Prototype-sample relation distillation: towards replay-free continual learning,” in *International Conference on Machine Learning*, pp. 1093–1106, PMLR, 2023.
- [119] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, *et al.*, “Huggingface’s transformers: State-of-the-art natural language processing,” *arXiv preprint arXiv:1910.03771*, 2019.
- [120] C. Raffel, “Building machine learning models like open source software,” *Communications of the ACM*, vol. 66, no. 2, pp. 38–40, 2023.



- [121] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, *et al.*, “On the opportunities and risks of foundation models,” *arXiv preprint arXiv:2108.07258*, 2021.
- [122] B. Sorscher, R. Geirhos, S. Shekhar, S. Ganguli, and A. S. Morcos, “Beyond neural scaling laws: beating power law scaling via data pruning,” *arXiv preprint arXiv:2206.14486*, 2022.
- [123] Y. Shen, Y. Xiong, W. Xia, and S. Soatto, “Towards backward-compatible representation learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6368–6377, 2020.
- [124] Q. Meng, C. Zhang, X. Xu, and F. Zhou, “Learning compatible embeddings,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9939–9948, 2021.
- [125] N. Biondi, F. Pernici, M. Bruni, and A. Del Bimbo, “Cores: Compatible representations via stationarity,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.
- [126] E. Strubell, A. Ganesh, and A. McCallum, “Energy and policy considerations for deep learning in nlp,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3645–3650, 2019.
- [127] W. N. Price and I. G. Cohen, “Privacy in the age of medical big data,” *Nature medicine*, vol. 25, no. 1, pp. 37–43, 2019.
- [128] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, *et al.*, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.
- [129] C. Wang, Y. Chang, S. Yang, D. Chen, and S. Lai, “Unified representation learning for cross model compatibility,” in *31st British Machine Vision Conference 2020, BMVC 2020*, BMVA Press, 2020.
- [130] B. Zhang, Y. Ge, Y. Shen, S. Su, C. Yuan, X. Xu, Y. Wang, and Y. Shan, “Towards universal backward-compatible representation learning,” *arXiv preprint arXiv:2203.01583*, 2022.
- [131] R. Duggal, H. Zhou, S. Yang, Y. Xiong, W. Xia, Z. Tu, and S. Soatto, “Compatibility-aware heterogeneous visual search,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10723–10732, 2021.
- [132] T. Pan, F. Xu, X. Yang, S. He, C. Jiang, Q. Guo, F. Qian, X. Zhang, Y. Cheng, L. Yang, *et al.*, “Boundary-aware backward-compatible representation via adversarial learning in image retrieval,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15201–15210, 2023.
- [133] A. Iscen, J. Zhang, S. Lazebnik, and C. Schmid, “Memory-efficient incremental learning through feature adaptation,” in *European Conference on Computer Vision*, pp. 699–715, Springer, 2020.
- [134] T. S. T. Wan, J.-C. Chen, T.-Y. Wu, and C.-S. Chen, “Continual learning for visual search with backward consistent feature embedding,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16702–16711, June 2022.



- [135] N. Biondi, F. Pernici, M. Bruni, D. Mugnai, and A. D. Bimbo, “Cl2r: Compatible lifelong learning representations,” *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 18, no. 2s, pp. 1–22, 2023.
- [136] F. Träuble, J. Von Kügelgen, M. Kleindessner, F. Locatello, B. Schölkopf, and P. Gehler, “Backward-compatible prediction updates: A probabilistic approach,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 116–128, 2021.
- [137] Y. Zhou, Z. Li, A. Shrivastava, H. Zhao, A. Torralba, T. Tian, and S.-N. Lim, “Bt²: Backward-compatible training with basis transformation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11229–11238, 2023.
- [138] F. Pernici, M. Bruni, C. Baecchi, and A. Del Bimbo, “Regular polytope networks,” *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
- [139] F. Pernici, M. Bruni, C. Baecchi, and A. Del Bimbo, “Maximally compact and separated features with regular polytope networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.
- [140] V. Papayan, X. Han, and D. L. Donoho, “Prevalence of neural collapse during the terminal phase of deep learning training,” *Proceedings of the National Academy of Sciences*, vol. 117, no. 40, pp. 24652–24663, 2020.
- [141] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” *arXiv preprint arXiv:1807.03748*, 2018.
- [142] P. Chrabaszcz, I. Loshchilov, and F. Hutter, “A downsampled variant of imagenet as an alternative to the cifar datasets,” *arXiv preprint arXiv:1707.08819*, 2017.
- [143] V. V. Ramasesh, A. Lewkowycz, and E. Dyer, “Effect of scale on catastrophic forgetting in neural networks,” in *International Conference on Learning Representations*, 2021.
- [144] A. Krizhevsky, “Learning Multiple Layers of Features from Tiny Images,” technical report, Univ. Toronto, 2009.
- [145] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.
- [146] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, “Designing network design spaces,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10428–10436, 2020.
- [147] N. Biondi, F. Pernici, S. Ricci, and A. Del Bimbo, “Stationary representations: Optimally approximating compatibility and implications for improved model replacements,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28793–28804, 2024.
- [148] C. Zhuang, Z. Xiang, Y. Bai, X. Jia, N. Turk-Browne, K. Norman, J. J. DiCarlo, and D. Yamins, “How well do unsupervised learning algorithms model human real-time and life-long learning?,” *Advances in neural information processing systems*, vol. 35, pp. 22628–22642, 2022.





- [149] R. Rajalingham, E. B. Issa, P. Bashivan, K. Kar, K. Schmidt, and J. J. DiCarlo, “Large-scale, high-resolution comparison of the core visual object recognition behavior of humans, monkeys, and state-of-the-art deep artificial neural networks,” *Journal of Neuroscience*, vol. 38, no. 33, pp. 7255–7269, 2018.
- [150] S. Omidshafiei, D.-K. Kim, M. Liu, G. Tesauero, M. Riemer, C. Amato, M. Campbell, and J. P. How, “Learning to teach in cooperative multiagent reinforcement learning,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, pp. 6128–6136, 2019.
- [151] T. Yang, W. Wang, H. Tang, J. Hao, Z. Meng, H. Mao, D. Li, W. Liu, Y. Chen, Y. Hu, *et al.*, “An efficient transfer learning framework for multiagent reinforcement learning,” *Advances in neural information processing systems*, vol. 34, pp. 17037–17048, 2021.
- [152] E. Nikonova, C. Xue, and J. Renz, “Efficient open-world reinforcement learning via knowledge distillation and autonomous rule discovery,” *arXiv preprint arXiv:2311.14270*, 2023.
- [153] Y. Ba, X. Liu, X. Chen, H. Wang, Y. Xu, K. Li, and S. Zhang, “Cautiously-optimistic knowledge sharing for cooperative multi-agent reinforcement learning,” *arXiv preprint arXiv:2312.12095*, 2023.
- [154] F. Ye and A. G. Bors, “Lifelong teacher-student network learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 6280–6296, 2021.
- [155] Z. Liu, Q. Liu, Y. Li, L. Liu, A. Shrivastava, S. Bi, L. Hong, E. H. Chi, and Z. Zhao, “Wisdom of committee: Distilling from foundation model to specialized application model,” *arXiv preprint arXiv:2402.14035*, 2024.
- [156] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, “Deep mutual learning,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4320–4328, 2018.
- [157] Q. Guo, X. Wang, Y. Wu, Z. Yu, D. Liang, X. Hu, and P. Luo, “Online knowledge distillation via collaborative learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11020–11029, 2020.
- [158] A. Yao and D. Sun, “Knowledge transfer via dense cross-layer mutual-distillation,” in *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pp. 294–311, Springer, 2020.
- [159] Z. Xiao, Q. Yan, and Y. Amit, “Likelihood regret: An out-of-distribution detection score for variational auto-encoder,” *Advances in neural information processing systems*, vol. 33, pp. 20685–20696, 2020.
- [160] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” in *NIPS Deep Learning and Representation Learning Workshop*, 2015.
- [161] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, “Overcoming catastrophic forgetting in neural networks,” *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.
- [162] M. Milani Fard, Q. Cormier, K. Canini, and M. Gupta, “Launch and iterate: Reducing prediction churn,” in *Advances in Neural Information Processing Systems* (D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, eds.), vol. 29, Curran Associates, Inc., 2016.





- [163] G. Zhang, L. Wang, G. Kang, L. Chen, and Y. Wei, “Slca: Slow learner with classifier alignment for continual learning on a pre-trained model,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19148–19158, 2023.
- [164] A. Heng and H. Soh, “Selective amnesia: A continual learning approach to forgetting in deep generative models,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [165] S. Hou, X. Liu, and Z. Wang, “Dualnet: Learn complementary features for image recognition,” in *Proceedings of the IEEE International conference on computer vision*, pp. 502–510, 2017.
- [166] X. Zhu, S. Gong, *et al.*, “Knowledge distillation by on-the-fly native ensemble,” *Advances in neural information processing systems*, vol. 31, 2018.
- [167] G. Wu and S. Gong, “Peer collaborative learning for online knowledge distillation,” in *Proceedings of the AAAI Conference on artificial intelligence*, vol. 35, pp. 10302–10310, 2021.
- [168] M. Zhang, L. Wang, D. Campos, W. Huang, C. Guo, and B. Yang, “Weighted mutual learning with diversity-driven model compression,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 11520–11533, 2022.
- [169] B. Qian, Y. Wang, H. Yin, R. Hong, and M. Wang, “Switchable online knowledge distillation,” in *European Conference on Computer Vision*, pp. 449–466, Springer, 2022.
- [170] S. Zagoruyko and N. Komodakis, “Wide residual networks,” in *BMVC*, pp. 87.1–87.12, 2016.
- [171] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [172] J. Gou, B. Yu, S. J. Maybank, and D. Tao, “Knowledge distillation: A survey,” *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.
- [173] C. Yang, L. Xie, S. Qiao, and A. Yuille, “Knowledge distillation in generations: More tolerant teachers educate better students,” *arXiv preprint arXiv:1805.05551*, 2018.
- [174] J. Yim, D. Joo, J. Bae, and J. Kim, “A gift from knowledge distillation: Fast optimization, network minimization and transfer learning,” in *Proc. CVPR*, 2017.
- [175] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, “Memory aware synapses: Learning what (not) to forget,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 139–154, 2018.
- [176] S. S. Shapiro and M. B. Wilk, “An analysis of variance test for normality (complete samples),” *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.
- [177] T. Fawcett, “An introduction to roc analysis,” *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.
- [178] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Ng, “Reading digits in natural images with unsupervised feature learning,” *NIPS*, 01 2011.
- [179] A. Krizhevsky, “Learning multiple layers of features from tiny images,” 2009.
- [180] I. Valsamara, C. Papaioannidis, and I. Pitas, “Domain expertise assessment for multi-dnn agent systems,” in *IEEE International Workshop on Distributed Intelligent Systems (DistInSys 2024)*, 2024.





- [181] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, 2015.
- [182] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, “beta-vae: Learning basic visual concepts with a constrained variational framework,” *ICLR*, 2016.
- [183] X. Chen, Y. Duan, R. Houthoof, J. Schulman, I. Sutskever, and P. Abbeel, “Infogan: Interpretable representation learning by information maximizing generative adversarial nets,” *NeurIPS*, 2016.
- [184] I. Higgins, D. Amos, D. Pfau, S. Racaniere, L. Matthey, D. Rezende, and A. Lerchner, “Towards a definition of disentangled representations,” *arXiv preprint arXiv:1812.02230*, 2018.
- [185] Y. Song, N. Sebe, and W. Wang, “Orthogonal svd covariance conditioning and latent disentanglement,” *IEEE T-PAMI*, 2022.
- [186] Y. Shen and B. Zhou, “Closed-form factorization of latent semantics in gans,” in *CVPR*, 2021.
- [187] Y. Wei, Y. Shi, X. Liu, Z. Ji, Y. Gao, Z. Wu, and W. Zuo, “Orthogonal jacobian regularization for unsupervised disentanglement in image generation,” in *ICCV*, 2021.
- [188] W. Peebles, J. Peebles, J.-Y. Zhu, A. Efros, and A. Torralba, “The hessian penalty: A weak prior for unsupervised disentanglement,” in *ECCV*, 2020.
- [189] C. Tzelepis, G. Tzimiropoulos, and I. Patras, “WarpedGANSpace: Finding non-linear rbf paths in GAN latent space,” in *ICCV*, 2021.
- [190] Y. Song, A. Keller, N. Sebe, and M. Welling, “Latent traversals in generative models as potential flows,” in *International Conference on Machine Learning*, 2023.
- [191] I. Higgins, L. Chang, V. Langston, D. Hassabis, C. Summerfield, D. Tsao, and M. Botvinick, “Unsupervised deep learning identifies semantic disentanglement in single inferotemporal face patch neurons,” *Nature communications*, 2021.
- [192] J. C. Whittington, W. Dorrell, S. Ganguli, and T. Behrens, “Disentanglement with biological constraints: A theory of functional cell types,” in *ICLR*, 2023.
- [193] D. Bouchacourt, M. Ibrahim, and S. Deny, “Addressing the topological defects of disentanglement via distributed operators,” *arXiv preprint arXiv:2102.05623*, 2021.
- [194] T. S. Cohen and M. Welling, “Transformation properties of learned visual representations,” *ICLR*, 2015.
- [195] T. S. Cohen and M. Welling, “Group equivariant convolutional networks,” in *ICML*, PMLR, 2016.
- [196] J. Köhler, L. Klein, and F. Noé, “Equivariant flows: exact likelihood generative learning for symmetric densities,” in *ICML*, PMLR, 2020.
- [197] V. G. Satorras, E. Hoogeboom, F. B. Fuchs, I. Posner, and M. Welling, “E (n) equivariant normalizing flows,” *NeurIPS*, 2021.
- [198] N. Dey, A. Chen, and S. Ghafurian, “Group equivariant generative adversarial networks,” *ICLR*, 2021.



- [199] E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling, “Equivariant diffusion for molecule generation in 3d,” in *ICML*, PMLR, 2022.
- [200] D. Klindt, L. Schott, Y. Sharma, I. Ustyuzhaninov, W. Brendel, M. Bethge, and D. Paiton, “Towards nonlinear disentanglement in natural data with temporal sparse coding,” *ICLR*, 2021.
- [201] T. A. Keller and M. Welling, “Topographic vaes learn equivariant capsules,” *NeurIPS*, 2021.
- [202] Y. Song, A. Keller, N. Sebe, and M. Welling, “Latent traversals in generative models as potential flows,” in *ICML*, PMLR, 2023.
- [203] J.-D. Benamou and Y. Brenier, “A computational fluid mechanics solution to the monge-kantorovich mass transfer problem,” *Numerische Mathematik*, 2000.
- [204] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *ICLR*, 2014.
- [205] Y. LeCun, “The mnist database of handwritten digits,” 1998.
- [206] H. Kim and A. Mnih, “Disentangling by factorising,” in *ICML*, 2018.
- [207] M. Raissi, P. Perdikaris, and G. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” *Journal of Computational Physics*, 2019.
- [208] J.-T. Hsieh, S. Zhao, S. Eismann, L. Mirabella, and S. Ermon, “Learning neural pde solvers with convergence guarantees,” *ICLR*, 2019.
- [209] J. Brandstetter, D. Worrall, and M. Welling, “Message passing neural pde solvers,” *ICLR*, 2022.
- [210] J. Richter-Powell, Y. Lipman, and R. T. Chen, “Neural conservation laws: A divergence-free perspective,” *NeurIPS*, 2022.
- [211] Q. Zeng, S. H. Bryngelson, and F. Schäfer, “Competitive physics informed networks,” *ICLR*, 2023.
- [212] C. Bajaj, L. McLennan, T. Andeen, and A. Roy, “Recipes for when physics fails: Recovering robust learning of physics informed neural networks,” *Machine Learning: Science and Technology*, 2023.
- [213] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig, “Scaling up visual and vision-language representation learning with noisy text supervision,” in *International Conference on Machine Learning*, pp. 4904–4916, PMLR, 2021.
- [214] L. Yuan, D. Chen, Y.-L. Chen, N. C. F. Codella, X. Dai, J. Gao, H. Hu, X. Huang, B. Li, C. Li, C. Liu, M. Liu, Z. Liu, Y. Lu, Y. Shi, L. Wang, J. Wang, B. Xiao, Z. Xiao, J. Yang, M. Zeng, L. Zhou, and P. Zhang, “Florence: A new foundation model for computer vision,” *ArXiv*, vol. abs/2111.11432, 2021.
- [215] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” *CVPR*, Jun 2022.



- [216] K. Crowson, S. Biderman, D. Kornis, D. Stander, E. Hallahan, L. Castricato, and E. Raff, “VQGAN-clip: Open domain image generation and editing with natural language guidance,” in *ECCV*, pp. 88–105, 2022.
- [217] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski, “StyleCLIP: Text-driven manipulation of StyleGAN imagery,” in *ICCV*, pp. 2085–2094, 2021.
- [218] R. Mokady, A. Hertz, and A. H. Bermano, “ClipCap: Clip prefix for image captioning,” *arXiv preprint arXiv:2111.09734*, 2021.
- [219] Y. Su, T. Lan, Y. Liu, F. Liu, D. Yogatama, Y. Wang, L. Kong, and N. Collier, “Language models can see: Plugging visual controls in text generation,” *ArXiv*, vol. abs/2205.02655, 2022.
- [220] S. Shen, L. H. Li, H. Tan, M. Bansal, A. Rohrbach, K.-W. Chang, Z. Yao, and K. Keutzer, “How much can CLIP benefit vision-and-language tasks?,” in *ICLR*, 2022.
- [221] B. Ni, H. Peng, M. Chen, S. Zhang, G. Meng, J. Fu, S. Xiang, and H. Ling, “Expanding language-image pretrained models for general video recognition,” in *ECCV*, pp. 1–18, Springer, 2022.
- [222] M. Wang, J. Xing, and Y. Liu, “ActionCLIP: A new paradigm for video action recognition,” *ArXiv*, vol. abs/2109.08472, 2021.
- [223] G. Goh, N. C. †, C. V. †, S. Carter, M. Petrov, L. Schubert, A. Radford, and C. Olah, “Multimodal neurons in artificial neural networks,” *Distill*, 2021. <https://distill.pub/2021/multimodal-neurons>.
- [224] S. Menon, I. P. Chandratreya, and C. Vondrick, “Task bias in vision-language models,” *ArXiv*, vol. abs/2212.04412, 2022.
- [225] Y. Yao, J. Zhang, F. Shen, W. Yang, P. Huang, and Z. Tang, “Discovering and distinguishing multiple visual senses for polysemous words,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.
- [226] D. Rampas, P. Pernias, E. Zhong, and M. Aubreville, “Fast text-conditional discrete denoising on vector-quantized latent spaces,” 2022.
- [227] G. A. Miller, “WordNet: A lexical database for english,” *Commun. ACM*, vol. 38, p. 39–41, nov 1995.
- [228] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, “Hierarchical text-conditional image generation with clip latents,” 2022.
- [229] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” 2021.
- [230] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [231] A. Torralba and A. A. Efros, “Unbiased look at dataset bias,” in *CVPR 2011*, pp. 1521–1528, 2011.





- [232] T. Bolukbasi, K.-W. Chang, J. Zou, V. Saligrama, and A. Kalai, “Man is to computer programmer as woman is to homemaker? debiasing word embeddings,” in *NeurIPS*, 2016.
- [233] J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and K.-W. Chang, “Men also like shopping: Reducing gender bias amplification using corpus-level constraints,” in *EMNLP*, 2017.
- [234] L. A. Hendricks, K. Burns, K. Saenko, T. Darrell, and A. Rohrbach, “Women also snowboard: Overcoming bias in captioning models,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [235] J. Buolamwini and T. Gebru, “Gender shades: Intersectional accuracy disparities in commercial gender classification,” in *Proceedings of the 1st Conference on Fairness, Accountability and Transparency* (S. A. Friedler and C. Wilson, eds.), vol. 81 of *Proceedings of Machine Learning Research*, pp. 77–91, PMLR, 23–24 Feb 2018.
- [236] J. Liu, B. Ni, Y. Yan, P. Zhou, S. Cheng, and J. Hu, “Pose transferrable person re-identification,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [237] M. D’incà, C. Beyan, R. Niewiadomski, S. Barattin, and N. Sebe, “Unleashing the transferability power of unsupervised pre-training for emotion recognition in masked and unmasked facial images,” *IEEE Access*, 2023.
- [238] A. Paleyes, R.-G. Urma, and N. D. Lawrence, “Challenges in deploying machine learning: A survey of case studies,” *ACM Computing Surveys*, vol. 55, pp. 1–29, dec 2022.
- [239] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.
- [240] J. Oldfield, C. Tzelepis, Y. Panagakis, M. Nicolaou, and I. Patras, “PandA: Unsupervised learning of parts and appearances in the feature maps of GANs,” in *The Eleventh International Conference on Learning Representations*, 2023.
- [241] J. Oldfield, M. Georgopoulos, Y. Panagakis, M. A. Nicolaou, and I. Patras, “Tensor component analysis for interpreting the latent space of gans,” *arXiv preprint arXiv:2111.11736*, 2021.
- [242] C. Tzelepis, G. Tzimiropoulos, and I. Patras, “WarpedGANSpace: Finding non-linear RBF paths in GAN latent space,” *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6373–6382, 2021.
- [243] S. Bounareli, C. Tzelepis, V. Argyriou, I. Patras, and G. Tzimiropoulos, “Stylemask: Disentangling the style space of stylegan2 for neural face reenactment,” in *17th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2023, Waikoloa Beach, HI, USA, January 5-8, 2023*, pp. 1–8, IEEE, 2023.
- [244] S. Bounareli, C. Tzelepis, V. Argyriou, I. Patras, and G. Tzimiropoulos, “HyperReenact: One-shot reenactment via jointly learning to refine and retarget faces,” *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- [245] D. Xu, S. Yuan, L. Zhang, and X. Wu, “Fairgan: Fairness-aware generative adversarial networks,” in *2018 IEEE international conference on big data (big data)*, pp. 570–575, IEEE, 2018.



- [246] B. Chaudhari, H. Chaudhary, A. Agarwal, K. Meena, and T. Bhowmik, “Fairgen: Fair synthetic data generation,” 2022.
- [247] W. Xu, J. Zhao, F. Iannacci, and B. Wang, “Ffpdg: Fast, fair and private data generation,” 2023.
- [248] S. Dash, V. N. Balasubramanian, and A. Sharma, “Evaluating and mitigating bias in image classifiers: A causal perspective using counterfactuals,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 915–924, January 2022.
- [249] M. Abroshan, M. M. Khalili, and A. Elliott, “Counterfactual fairness in synthetic data generation,” in *NeurIPS 2022 Workshop on Synthetic Data for Empowering ML Research*, 2022.
- [250] F. Zhang, K. Kuang, L. Chen, Y. Liu, C. Wu, and J. Xiao, “Fairness-aware contrastive learning with partially annotated sensitive attributes,” in *The Eleventh International Conference on Learning Representations*, 2023.
- [251] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” *arXiv preprint arXiv:2010.02502*, 2020.
- [252] C. Tzelepis, O. James, G. Tzimiropoulos, and I. Patras, “ContraCLIP: Interpretable GAN generation driven by pairs of contrasting sentences,” 2022.
- [253] M. Chen, H. Xue, and D. Cai, “Domain adaptation for semantic segmentation with maximum squares loss,” in *ICCV*, 2019.
- [254] Q. Zhang, J. Zhang, W. Liu, and D. Tao, “Category anchor-guided unsupervised domain adaptation for semantic segmentation,” *NeurIPS*, 2019.
- [255] P. Zhang, B. Zhang, T. Zhang, D. Chen, Y. Wang, and F. Wen, “Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation,” in *CVPR*, 2021.
- [256] Z. Zheng and Y. Yang, “Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation,” *IJCV*, 2021.
- [257] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez, “Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation,” in *CVPR*, 2019.
- [258] I. Gulrajani and D. Lopez-Paz, “In search of lost domain generalization,” in *ICLR*, 2021.
- [259] S. Choi, S. Jung, H. Yun, J. T. Kim, S. Kim, and J. Choo, “Robustnet: Improving domain generalization in urban-scene segmentation via instance selective whitening,” in *CVPR*, 2021.
- [260] Q. Liu, C. Chen, J. Qin, Q. Dou, and P.-A. Heng, “Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space,” in *CVPR*, 2021.
- [261] D. Li, J. Yang, K. Kreis, A. Torralba, and S. Fidler, “Semantic segmentation with generative models: Semi-supervised learning and strong out-of-domain generalization,” in *CVPR*, 2021.
- [262] D. Peng, Y. Lei, M. Hayat, Y. Guo, and W. Li, “Semantic-aware domain generalized segmentation,” in *CVPR*, 2022.



- [263] H. Li, S. J. Pan, S. Wang, and A. C. Kot, “Domain generalization with adversarial feature learning,” in *CVPR*, 2018.
- [264] S. Zhao, B. Li, X. Yue, Y. Gu, P. Xu, R. Hu, H. Chai, and K. Keutzer, “Multi-source domain adaptation for semantic segmentation,” *NeurIPS*, 2019.
- [265] M. Long, Z. Cao, J. Wang, and M. I. Jordan, “Conditional adversarial domain adaptation,” *NeurIPS*, 2018.
- [266] Y. Zou, Z. Yu, B. Kumar, and J. Wang, “Unsupervised domain adaptation for semantic segmentation via class-balanced self-training,” in *ECCV*, 2018.
- [267] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, “Tent: Fully test-time adaptation by entropy minimization,” in *ICLR*, 2021.
- [268] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt, “Test-time training with self-supervision for generalization under distribution shifts,” in *ICML*, 2020.
- [269] Y. Liu, P. Kothari, B. van Delft, B. Bellot-Gurlet, T. Mordan, and A. Alahi, “Ttt++: When does self-supervised test-time training fail or thrive?,” *NeurIPS*, 2021.
- [270] S. Niu, J. Wu, Y. Zhang, Y. Chen, S. Zheng, P. Zhao, and M. Tan, “Efficient test-time model adaptation without forgetting,” in *ICML*, 2022.
- [271] D. Chen, D. Wang, T. Darrell, and S. Ebrahimi, “Contrastive test-time adaptation,” in *CVPR*, 2022.
- [272] X. Pan, P. Luo, J. Shi, and X. Tang, “Two at once: Enhancing learning and generalization capacities via ibn-net,” in *ECCV*, 2018.
- [273] S. Schneider, E. Rusak, L. Eck, O. Bringmann, W. Brendel, and M. Bethge, “Improving robustness against common corruptions by covariate shift adaptation,” *NeurIPS*, 2020.
- [274] M. J. Mirza, J. Micorek, H. Possegger, and H. Bischof, “The norm must go on: Dynamic unsupervised domain adaptation by normalization,” in *CVPR*, 2022.
- [275] A. Khurana, S. Paul, P. Rai, S. Biswas, and G. Aggarwal, “Sita: Single image test-time adaptation,” *arXiv preprint arXiv:2112.02355*, 2021.
- [276] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, “Playing for data: Ground truth from computer games,” in *ECCV*, 2016.
- [277] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *CVPR*, 2016.
- [278] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, “Bdd100k: A diverse driving dataset for heterogeneous multitask learning,” in *CVPR*, 2020.
- [279] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kotschieder, “The mapillary vistas dataset for semantic understanding of street scenes,” in *ICCV*, 2017.
- [280] G. Varma, A. Subramanian, A. Nambodiri, M. Chandraker, and C. Jawahar, “Idd: A dataset for exploring problems of autonomous navigation in unconstrained environments,” in *WACV*, 2019.



- [281] Y.-H. Chen, W.-Y. Chen, Y.-T. Chen, B.-C. Tsai, Y.-C. Frank Wang, and M. Sun, “No more discrimination: Cross city adaptation of road scene segmenters,” in *ICCV*, 2017.
- [282] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, “Tent: Fully test-time adaptation by entropy minimization,” in *ICLR*, 2021.
- [283] W. Wang, Z. Zhong, W. Wang, X. Chen, C. Ling, B. Wang, and N. Sebe, “Dynamically instance-guided adaptation: A backward-free approach for test-time domain adaptive semantic segmentation,” in *CVPR*, 2023.
- [284] M. Bain, A. Nagrani, G. Varol, and A. Zisserman, “Frozen in time: A joint video and image encoder for end-to-end retrieval,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1728–1738, 2021.
- [285] Y. Ge, Y. Ge, X. Liu, D. Li, Y. Shan, X. Qie, and P. Luo, “Bridging video-text retrieval with multiple choice questions,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16167–16176, June 2022.
- [286] J. Lei, T. L. Berg, and M. Bansal, “Revealing single frame bias for video-and-language learning,” *arXiv preprint arXiv:2206.03428*, 2022.
- [287] X. Zhu, J. Zhu, H. Li, X. Wu, H. Li, X. Wang, and J. Dai, “Uni-perceiver: Pre-training unified architecture for generic perception for zero-shot and few-shot tasks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16804–16815, June 2022.
- [288] F. Cheng, X. Wang, J. Lei, D. Crandall, M. Bansal, and G. Bertasius, “Vindlu: A recipe for effective video-and-language pretraining,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10739–10750, June 2023.
- [289] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu, “Video paragraph captioning using hierarchical recurrent neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4584–4593, 2016.
- [290] X. Wang, L. Zhu, and Y. Yang, “T2vlad: global-local sequence alignment for text-video retrieval,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5079–5088, 2021.
- [291] Z. Yu, D. Xu, J. Yu, T. Yu, Z. Zhao, Y. Zhuang, and D. Tao, “Activitynet-qa: A dataset for understanding complex web videos via question answering,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 9127–9134, 2019.
- [292] L. Parcalabescu, M. Cafagna, L. Muradjan, A. Frank, I. Calixto, and A. Gatt, “VALSE: A task-independent benchmark for vision and language models centered on linguistic phenomena,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Dublin, Ireland), pp. 8253–8280, Association for Computational Linguistics, May 2022.
- [293] L. A. Hendricks and A. Nematzadeh, “Probing image-language transformers for verb understanding,” in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, (Online), pp. 3635–3644, Association for Computational Linguistics, Aug. 2021.



- [294] T. Thrush, R. Jiang, M. Bartolo, A. Singh, A. Williams, D. Kiela, and C. Ross, “Winoground: Probing vision and language models for visio-linguistic compositionality,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5238–5248, 2022.
- [295] R. Shekhar, S. Pezzelle, Y. Klimovich, A. Herbelot, M. Nabi, E. Sangineto, and R. Bernardi, “FOIL it! find one mismatch between image and language caption,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Vancouver, Canada), pp. 255–265, Association for Computational Linguistics, July 2017.
- [296] T. F. H. Runia, C. G. M. Snoek, and A. W. M. Smeulders, “Real-world repetition estimation by div, grad and curl,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [297] A. Sadhu, T. Gupta, M. Yatskar, R. Nevatia, and A. Kembhavi, “Visual semantic role labeling for video understanding,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.
- [298] A. Warstadt, Y. Cao, I. Grosu, W. Peng, H. Blix, Y. Nie, A. Alsop, S. Bordia, H. Liu, A. Parrish, S.-F. Wang, J. Phang, A. Mohananeey, P. M. Htut, P. Jeretic, and S. R. Bowman, “Investigating BERT’s knowledge of language: Five analysis methods with NPIs,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, (Hong Kong, China), pp. 2877–2887, Association for Computational Linguistics, Nov. 2019.
- [299] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, *et al.*, “The “something something” video database for learning and evaluating visual common sense,” in *Proceedings of the IEEE international conference on computer vision*, pp. 5842–5850, 2017.
- [300] L. Zhou, C. Xu, and J. J. Corso, “Towards automatic learning of procedures from web instructional videos,” in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI’18/IAAI’18/EAAI’18, AAAI Press, 2018.
- [301] Y. Tang, D. Ding, Y. Rao, Y. Zheng, D. Zhang, L. Zhao, J. Lu, and J. Zhou, “Coin: A large-scale dataset for comprehensive instructional video analysis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1207–1216, 2019.
- [302] A. Miech, J.-B. Alayrac, I. Laptev, J. Sivic, and A. Zisserman, “Rareact: A video dataset of unusual interactions,” *arXiv preprint arXiv:2008.01018*, 2020.
- [303] B. Wu, S. Yu, Z. Chen, J. B. Tenenbaum, and C. Gan, “STAR: A benchmark for situated reasoning in real-world videos,” in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [304] A. Miech, D. Zhukov, J.-B. Alayrac, M. Tapaswi, I. Laptev, and J. Sivic, “Howto100m: Learning a text-video embedding by watching hundred million narrated video clips,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019.





- [305] J. Lei, L. Li, L. Zhou, Z. Gan, T. L. Berg, M. Bansal, and J. Liu, “Less is more: Clipbert for video-and-language learning via sparse sampling,” in *CVPR*, 2021.
- [306] H. Luo, L. Ji, B. Shi, H. Huang, N. Duan, T. Li, J. Li, T. Bharti, and M. Zhou, “Univl: A unified video and language pre-training model for multimodal understanding and generation,” *arXiv preprint arXiv:2002.06353*, 2020.
- [307] H. Xu, G. Ghosh, P.-Y. Huang, D. Okhonko, A. Aghajanyan, F. Metze, L. Zettlemoyer, and C. Feichtenhofer, “VideoCLIP: Contrastive pre-training for zero-shot video-text understanding,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, (Online and Punta Cana, Dominican Republic), pp. 6787–6800, Association for Computational Linguistics, Nov. 2021.
- [308] H. Luo, L. Ji, M. Zhong, Y. Chen, W. Lei, N. Duan, and T. Li, “Clip4clip: An empirical study of clip for end to end video clip retrieval and captioning,” *Neurocomputing*, vol. 508, pp. 293–304, 2022.
- [309] T.-J. Fu, L. Li, Z. Gan, K. Lin, W. Y. Wang, L. Wang, and Z. Liu, “Violet: End-to-end video-language transformers with masked visual-token modeling,” *arXiv preprint arXiv:2111.12681*, 2021.
- [310] Y. Ma, G. Xu, X. Sun, M. Yan, J. Zhang, and R. Ji, “X-clip: End-to-end multi-grained contrastive learning for video-text retrieval,” in *Proceedings of the 30th ACM International Conference on Multimedia*, p. 638–647, 2022.
- [311] R. Zellers, X. Lu, J. Hessel, Y. Yu, J. S. Park, J. Cao, A. Farhadi, and Y. Choi, “Merlot: Multimodal neural script knowledge models,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 23634–23651, 2021.
- [312] Y. Wang, K. Li, Y. Li, Y. He, B. Huang, Z. Zhao, H. Zhang, J. Xu, Y. Liu, Z. Wang, S. Xing, G. Chen, J. Pan, J. Yu, Y. Wang, L. Wang, and Y. Qiao, “Internvideo: General video foundation models via generative and discriminative learning,” 2022.
- [313] H. Xu, Q. Ye, M. Yan, Y. Shi, J. Ye, Y. Xu, C. Li, B. Bi, Q. Qian, W. Wang, G. Xu, J. Zhang, S. Huang, F. Huang, and J. Zhou, “mplug-2: A modularized multi-modal foundation model across text, image and video,” *ArXiv*, vol. abs/2302.00402, 2023.
- [314] B. Li, Y. Zhang, L. Chen, J. Wang, J. Yang, and Z. Liu, “Otter: A multi-modal model with in-context instruction tuning,” *ArXiv*, vol. abs/2305.03726, 2023.
- [315] H. Zhang, X. Li, and L. Bing, “Video-LLaMA: An instruction-tuned audio-visual language model for video understanding,” in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (Y. Feng and E. Lefever, eds.), (Singapore), pp. 543–553, Association for Computational Linguistics, Dec. 2023.
- [316] J. Li, D. Li, S. Savarese, and S. Hoi, “Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models,” *ArXiv*, vol. abs/2301.12597, 2023.
- [317] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, “Language Models are Unsupervised Multitask Learners,” *OpenAI blog*, vol. 1, no. 8, 2019.
- [318] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, *et al.*, “Opt: Open pre-trained transformer language models,” *arXiv preprint arXiv:2205.01068*, 2022.



- [319] P. Madhyastha, J. Wang, and L. Specia, “VIFIDEL: Evaluating the visual fidelity of image descriptions,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 6539–6550, Association for Computational Linguistics, July 2019.
- [320] I. Kesen, A. Pedrotti, M. Dogan, M. Cafagna, E. C. Acikgoz, L. Parcalabescu, I. Calixto, A. Frank, A. Gatt, A. Erdem, and E. Erdem, “ViLMA: A zero-shot benchmark for linguistic and temporal grounding in video-language models,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [321] N. Zhou, H. Wen, Y. Wang, Y. Liu, and L. Zhou, “Review of deep learning models for spine segmentation,” in *ICMR*, pp. 498–507, 2022.
- [322] Y. Wang, J.-N. Hwang, G. Wang, H. Liu, K.-J. Kim, H.-M. Hsu, J. Cai, H. Zhang, Z. Jiang, and R. Gu, “Rod2021 challenge: A summary for radar object detection challenge for autonomous driving applications,” in *ICMR*, pp. 553–559, 2021.
- [323] K.-T. Nguyen, D.-T. Dinh, M. N. Do, and M.-T. Tran, “Anomaly detection in traffic surveillance videos with gan-based future frame prediction,” in *ICMR*, pp. 457–463, 2020.
- [324] L. Liu, S. Zhang, Z. Kuang, A. Zhou, J.-H. Xue, X. Wang, Y. Chen, W. Yang, Q. Liao, and W. Zhang, “Group fisher pruning for practical network compression,” in *ICML*, pp. 7021–7032, 2021.
- [325] R. Berriel, S. Lathuillere, M. Nabi, T. Klein, T. Oliveira-Santos, N. Sebe, and E. Ricci, “Budget-aware adapters for multi-domain learning,” in *ICCV*, pp. 382–391, 2019.
- [326] Y. Du, Z. Chen, C. Jia, X. Li, and Y.-G. Jiang, “Bag of tricks for building an accurate and slim object detector for embedded applications,” in *ICMR*, pp. 519–525, 2021.
- [327] S. F. dos Santos and J. Almeida, “Less is more: Accelerating faster neural networks straight from jpeg,” in *Iberoamerican Congress on Pattern Recog. (CIARP)*, pp. 237–247, 2021.
- [328] S. F. dos Santos, N. Sebe, and J. Almeida, “The good, the bad, and the ugly: Neural networks straight from jpeg,” in *ICIP*, pp. 1896–1900, 2020.
- [329] A. Marchisio, M. A. Hanif, F. Khalid, G. Plastiras, C. Kyrkou, T. Theodorides, and M. Shafique, “Deep learning for edge computing: Current trends, cross-layer optimizations, and open research challenges,” in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 553–559, 2019.
- [330] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, “Learning multiple visual domains with residual adapters,” in *NeurIPS*, vol. 30, pp. 506–516, 2017.
- [331] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, “Efficient parametrization of multi-domain deep neural networks,” in *CVPR*, pp. 8119–8127, 2018.
- [332] S. Felipe dos Santos, T. Oliveira-Santos, N. Sebe, and J. Almeida, “Budget-aware pruning for multi-domain learning,” in *ICIAP*, 2023.
- [333] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shufflenet: an extremely efficient convolutional neural network for mobile devices,” in *Proc. CVPR*, 2018.





- [334] Y. Zhang, N. Suda, L. Lai, and V. Chandra, “Hello edge: Keyword spotting on microcontrollers,” *CoRR*, vol. abs/1711.07128, 2017.
- [335] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, *et al.*, “Searching for mobilenetv3,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1314–1324, 2019.
- [336] C.-J. Wu, D. Brooks, K. Chen, D. Chen, S. Choudhury, M. Dukhan, K. Hazelwood, E. Isaac, Y. Jia, B. Jia, *et al.*, “Machine learning at facebook: Understanding inference at the edge,” in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 331–344, IEEE, 2019.
- [337] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, “Once-for-all: Train one network and specialize it for efficient deployment,” in *International Conference on Learning Representations*, 2020.
- [338] J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang, “Slimmable neural networks,” in *International Conference on Learning Representations*, 2019.
- [339] J. Yu and T. S. Huang, “Universally slimmable networks and improved training techniques,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1803–1811, 2019.
- [340] L. Guerra, B. Zhuang, I. Reid, and T. Drummond, “Switchable precision neural networks,” *arXiv preprint arXiv:2002.02815*, 2020.
- [341] Q. Jin, L. Yang, and Z. Liao, “Adabits: Neural network quantization with adaptive bit-widths,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2146–2156, 2020.
- [342] A. Lambert, F. Le Bolzer, and F. Schnitzler, “Flexible recurrent neural networks,” in *Machine Learning and Knowledge Discovery in Databases*, 2020.
- [343] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, “On the convergence of fedavg on non-iid data,” *arXiv preprint arXiv:1907.02189*, 2019.
- [344] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.
- [345] M. Blot, D. Picard, M. Cord, and N. Thome, “Gossip training for deep learning,” *arXiv preprint arXiv:1611.09726*, 2016.
- [346] I. Hegedűs, G. Danner, and M. Jelasity, “Gossip learning as a decentralized alternative to federated learning,” in *IFIP International Conference on Distributed Applications and Interoperable Systems*, pp. 74–90, Springer, 2019.
- [347] C. Buciluă, R. Caruana, and A. Niculescu-Mizil, “Model compression,” in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.
- [348] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, “A survey of model compression and acceleration for deep neural networks,” *arXiv preprint arXiv:1710.09282*, 2017.
- [349] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han, “Amc: Automl for model compression and acceleration on mobile devices,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 784–800, 2018.





- [350] E. Upton and G. Halfacree, *Raspberry Pi user guide*. John Wiley & Sons, 2016.
- [351] M. Richardson and S. Wallace, *Getting Started with Raspberry Pi: Electronic Projects with Python, Scratch, and Linux*. Maker Media, Inc., 2014.
- [352] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, “Neural message passing for quantum chemistry,” in *International conference on machine learning*, pp. 1263–1272, PMLR, 2017.
- [353] J. Feng, Y. Chen, F. Li, A. Sarkar, and M. Zhang, “How powerful are k-hop message passing graph neural networks,” *arXiv preprint arXiv:2205.13328*, 2022.
- [354] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.
- [355] Q. Huang, H. He, A. Singh, S.-N. Lim, and A. R. Benson, “Combining label propagation and simple models out-performs graph neural networks,” *arXiv preprint arXiv:2010.13993*, 2020.
- [356] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *International conference on machine learning*, pp. 1597–1607, PMLR, 2020.
- [357] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, “Beyond homophily in graph neural networks: Current limitations and effective designs,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 7793–7804, 2020.
- [358] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *arXiv preprint arXiv:1710.10903*, 2017.
- [359] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, “Gaan: Gated attention networks for learning on large and spatiotemporal graphs,” *arXiv preprint arXiv:1803.07294*, 2018.
- [360] R. Sato, M. Yamada, and H. Kashima, “Random features strengthen graph neural networks,” in *Proceedings of the 2021 SIAM international conference on data mining (SDM)*, pp. 333–341, SIAM, 2021.
- [361] N. Keriven and S. Vaiter, “What functions can graph neural networks compute on random graphs? the role of positional encoding,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [362] N. Keriven and G. Peyré, “Universal invariant and equivariant graph neural networks,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [363] J. Ellson, E. Gansner, L. Koutsofios, S. C. North, and G. Woodhull, “Graphviz—open source graph drawing tools,” in *International Symposium on Graph Drawing*, pp. 483–484, Springer, 2001.
- [364] L. Peel, D. B. Larremore, and A. Clauset, “The ground truth about metadata and community detection in networks,” *Science advances*, vol. 3, no. 5, p. e1602548, 2017.
- [365] F. Wu and B. A. Huberman, “Finding communities in linear time: a physics approach,” *The European Physical Journal B*, vol. 38, no. 2, pp. 331–338, 2004.





- [366] T. Zhang and B. Wu, “A method for local community detection by finding core nodes,” in *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 1171–1176, IEEE, 2012.
- [367] E. Krasanakis, E. Schinas, S. Papadopoulos, Y. Kompatsiaris, and A. Symeonidis, “Boosted seed oversampling for local community ranking,” *Information Processing & Management*, vol. 57, no. 2, p. 102053, 2020.
- [368] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,” *IEEE signal processing magazine*, vol. 30, no. 3, pp. 83–98, 2013.
- [369] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, “Graph signal processing: Overview, challenges, and applications,” *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.
- [370] L. Stanković, M. Daković, and E. Sejdić, “Introduction to graph signal processing,” in *Vertex-Frequency Analysis of Graph Signals*, pp. 3–108, Springer, 2019.
- [371] H. Tong, C. Faloutsos, and J.-Y. Pan, “Fast random walk with restart and its applications,” in *Sixth international conference on data mining (ICDM’06)*, pp. 613–622, IEEE, 2006.
- [372] R. Andersen, F. Chung, and K. Lang, “Local partitioning for directed graphs using pagerank,” *Internet Mathematics*, vol. 5, no. 1-2, pp. 3–22, 2008.
- [373] B. Bahmani, A. Chowdhury, and A. Goel, “Fast incremental and personalized pagerank,” *Proceedings of the VLDB Endowment*, vol. 4, no. 3, 2010.
- [374] K. Kloster and D. F. Gleich, “Heat kernel based community detection,” in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1386–1395, 2014.
- [375] J. Gasteiger, A. Bojchevski, and S. Günnemann, “Predict then propagate: Graph neural networks meet personalized pagerank,” *arXiv preprint arXiv:1810.05997*, 2018.
- [376] R. Andersen, F. Chung, and K. Lang, “Using pagerank to locally partition a graph,” *Internet Mathematics*, vol. 4, no. 1, pp. 35–64, 2007.
- [377] A. Susnjara, N. Perraudin, D. Kressner, and P. Vandergheynst, “Accelerated filtering on graphs using lanczos method,” *arXiv preprint arXiv:1509.04537*, 2015.
- [378] J. Leskovec and A. Krevl, “SNAP Datasets: Stanford large network dataset collection.” <http://snap.stanford.edu/data>, June 2014.
- [379] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, “A limited memory algorithm for bound constrained optimization,” *SIAM Journal on scientific computing*, vol. 16, no. 5, pp. 1190–1208, 1995.
- [380] A. Galántai, “Convergence of the nelder-meard method,” *Numerical Algorithms*, pp. 1–30, 2021.
- [381] H. Lyu, “Convergence of block coordinate descent with diminishing radius for nonconvex optimization,” *arXiv preprint arXiv:2012.03503*, 2020.



- [382] D. E. Finkel and C. Kelley, “Additive scaling and the direct algorithm,” *Journal of Global Optimization*, vol. 36, no. 4, pp. 597–608, 2006.
- [383] A. R. Al-Roomi, “Unconstrained Single-Objective Benchmark Functions Repository,” 2015.
- [384] E. D. D. Team, “Deeplearning4j: Open-source distributed deep learning for the JVM,” 2016.
- [385] T. Abeel, Y. Van de Peer, and Y. Saeys, “Java-ml: A machine learning library,” *Journal of Machine Learning Research*, vol. 10, pp. 931–934, 2009.
- [386] E. Raff, “JSAT: Java statistical analysis tool, a library for machine learning,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 792–796, 2017.
- [387] G. Demosthenous and V. Vassiliades, “Continual learning on the edge with tensorflow lite,” *arXiv preprint arXiv:2105.01946*, 2021.
- [388] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, “Collective classification in network data,” *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.
- [389] A. P. Bradley, “The use of the area under the roc curve in the evaluation of machine learning algorithms,” *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.
- [390] K. Järvelin and J. Kekäläinen, “Ir evaluation methods for retrieving highly relevant documents,” in *ACM SIGIR Forum*, vol. 51, pp. 243–250, ACM New York, NY, USA, 2017.
- [391] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola, “Autogluon-tabular: Robust and accurate automl for structured data,” *arXiv preprint arXiv:2003.06505*, 2020.
- [392] A. Hagberg, P. Swart, and D. S Chult, “Exploring network structure, dynamics, and function using networkx,” tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.
- [393] G. Csardi, T. Nepusz, *et al.*, “The igraph software package for complex network research,” *InterJournal, complex systems*, vol. 1695, no. 5, pp. 1–9, 2006.
- [394] M. Defferrard, L. Martin, R. Pena, and N. Perraudin, “Pygsp: Graph signal processing in python.”
- [395] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma, *et al.*, “Deep graph library: Towards efficient and scalable deep learning on graphs.” 2019.
- [396] M. Fey and J. E. Lenssen, “Fast graph representation learning with pytorch geometric,” *arXiv preprint arXiv:1903.02428*, 2019.
- [397] O. Ferludin, A. Eigenwillig, M. Blais, D. Zelle, J. Pfeifer, A. Sanchez-Gonzalez, S. Li, S. Abu-El-Haija, P. Battaglia, N. Bulut, J. Halcrow, F. M. G. de Almeida, S. Lattanzi, A. Linhares, B. Mayer, V. Mirrokni, J. Palowitch, M. Paradkar, J. She, A. Tsitsulin, K. Vilella, L. Wang, D. Wong, and B. Perozzi, “TF-GNN: graph neural networks in tensorflow,” *CoRR*, vol. abs/2207.03522, 2022.
- [398] D. Grattarola and C. Alippi, “Graph neural networks in tensorflow and keras with spektral,” *arXiv preprint arXiv:2006.12138*, 2020.



- [399] J. X. Parreira, D. Donato, S. Michel, and G. Weikum, “Efficient and decentralized pagerank approximation in a peer-to-peer web search network,” in *Proceedings of the 32nd international conference on Very large data bases*, pp. 415–426, 2006.
- [400] X. Zhang, J. You, H. Xue, and J. Wang, “A decentralized pagerank based content dissemination model at the edge of network,” *International Journal of Web Services Research (IJWSR)*, vol. 17, no. 1, pp. 1–16, 2020.
- [401] P. Hu and W. C. Lau, “Localized algorithm of community detection on large-scale decentralized social networks,” *arXiv preprint arXiv:1212.6323*, 2012.
- [402] B. Lubachevsky and D. Mitra, “A chaotic asynchronous algorithm for computing the fixed point of a nonnegative matrix of unit spectral radius,” *Journal of the ACM (JACM)*, vol. 33, no. 1, pp. 130–150, 1986.
- [403] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, “How powerful are graph neural networks?,” *arXiv preprint arXiv:1810.00826*, 2018.
- [404] A. Loukas, “What graph neural networks cannot learn: depth vs width,” *arXiv preprint arXiv:1907.03199*, 2019.
- [405] Q. Nguyen, M. C. Muckamala, and M. Hein, “On the loss landscape of a class of deep neural networks with no bad local valleys,” *arXiv preprint arXiv:1809.10749*, 2018.
- [406] R.-Y. Sun, “Optimization for deep learning: An overview,” *Journal of the Operations Research Society of China*, vol. 8, no. 2, pp. 249–294, 2020.
- [407] P. Kidger and T. Lyons, “Universal approximation with deep narrow networks,” in *Conference on learning theory*, pp. 2306–2327, PMLR, 2020.
- [408] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, “Pitfalls of graph neural network evaluation,” *arXiv preprint arXiv:1811.05868*, 2018.
- [409] A. Bojchevski and S. Günnemann, “Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking,” *arXiv preprint arXiv:1707.03815*, 2017.
- [410] G. Namata, B. London, L. Getoor, B. Huang, and U. EDU, “Query-driven active surveying for collective classification,” in *10th International Workshop on Mining and Learning with Graphs*, vol. 8, 2012.
- [411] O. J. Dunn, “Multiple comparisons among means,” *Journal of the American statistical association*, vol. 56, no. 293, pp. 52–64, 1961.
- [412] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *The Journal of Machine learning research*, vol. 7, pp. 1–30, 2006.
- [413] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, *et al.*, “Graph attention networks,” *stat*, vol. 1050, no. 20, pp. 10–48550, 2017.
- [414] R. Abboud, I. I. Ceylan, M. Grohe, and T. Lukasiewicz, “The surprising power of graph neural networks with random node initialization,” *arXiv preprint arXiv:2010.01179*, 2020.
- [415] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.



- [416] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, *et al.*, “Advances and open problems in federated learning,” *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.
- [417] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, *et al.*, “Towards federated learning at scale: System design,” *Proceedings of machine learning and systems*, vol. 1, pp. 374–388, 2019.
- [418] Y. Jiang, J. Konečný, K. Rush, and S. Kannan, “Improving federated learning personalization via model agnostic meta learning. arxiv 2019,” *arXiv preprint arXiv:1909.12488*.
- [419] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.
- [420] R. Storn and K. Price, “Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of global optimization*, vol. 11, pp. 341–359, 1997.
- [421] M. Dorigo, V. Maniezzo, and A. Coloni, “Ant system: optimization by a colony of cooperating agents,” *IEEE transactions on systems, man, and cybernetics, part b (cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.
- [422] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95-international conference on neural networks*, vol. 4, pp. 1942–1948, iee, 1995.
- [423] A. Krizhevsky, “Learning multiple layers of features from tiny images,” tech. rep., University of Toronto, 2009.
- [424] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, “The non-iid data quagmire of decentralized machine learning,” in *International Conference on Machine Learning*, pp. 4387–4398, PMLR, 2020.
- [425] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, “Bayesian nonparametric federated learning of neural networks,” in *International conference on machine learning*, pp. 7252–7261, PMLR, 2019.
- [426] Q. Zhang and H. Li, “Moea/d: A multiobjective evolutionary algorithm based on decomposition,” *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.
- [427] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-iid data,” *arXiv preprint arXiv:1806.00582*, 2018.
- [428] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, “A comprehensive survey on model compression and acceleration,” *Artificial Intelligence Review*, vol. 53, no. 7, 2020.
- [429] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, “Model compression and acceleration for deep neural networks: The principles, progress, and challenges,” *IEEE Signal Processing Magazine*, vol. 35, no. 1, 2018.
- [430] N. Passalis, M. Tzelepi, and A. Tefas, “Heterogeneous knowledge distillation using information flow modeling,” in *Proc. CVPR*, 2020.





- [431] S. Zagoruyko and N. Komodakis, “Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer,” in *Proc. ICLR*, 2017.
- [432] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, “Fitnets: Hints for thin deep nets,” in *Proc. ICLR*, 2015.
- [433] Y. Bengio, *Learning deep architectures for AI*. Now Publishers Inc, 2009.
- [434] S. I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, and H. Ghasemzadeh, “Improved knowledge distillation via teacher assistant,” in *Proc. AAAI*, 2020.
- [435] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, “Learning structured sparsity in deep neural networks,” in *Proc. NeurIPS*, 2016.
- [436] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both weights and connections for efficient neural network,” in *Proc. NeurIPS*, 2015.
- [437] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, “Pruning filters for efficient convnets,” in *Proc. ICLR*, 2017.
- [438] Y. Tian, D. Krishnan, and P. Isola, “Contrastive representation distillation,” in *International Conference on Learning Representations*, 2020.
- [439] I. Sarridis, C. Koutlis, G. Kordopatis-Zilos, I. Kompatsiaris, and S. Papadopoulos, “Indistill: Information flow-preserving knowledge distillation for model compression,” *arXiv preprint arXiv:2205.10003*, 2022.
- [440] N. Passalis and A. Tefas, “Learning deep representations with probabilistic knowledge transfer,” in *Proc. ECCV*, 2018.
- [441] B. Zhao, Q. Cui, R. Song, Y. Qiu, and J. Liang, “Decoupled knowledge distillation,” in *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pp. 11953–11962, 2022.
- [442] Y. Jin, J. Wang, and D. Lin, “Multi-level logit distillation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24276–24285, 2023.
- [443] G. Habib, T. J. Saleem, and B. Lall, “Knowledge Distillation in Vision Transformers: A Critical Review,” Feb. 2023. [arXiv:2302.02108](https://arxiv.org/abs/2302.02108) [cs].
- [444] E. Huynh, “Vision Transformers in 2022: An Update on Tiny ImageNet,” May 2022. [arXiv:2205.10660](https://arxiv.org/abs/2205.10660) [cs].
- [445] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” Apr. 2015. [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) [cs].
- [446] O. Zafrir, “Model-compression-research-package by intel labs,” Nov. 2021.
- [447] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” Mar. 2015. [arXiv:1503.02531](https://arxiv.org/abs/1503.02531) [cs, stat].
- [448] R. Reed, “Pruning algorithms—a survey,” *IEEE transactions on Neural Networks*, vol. 4, no. 5, pp. 740–747, 1993.



- [449] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, “Filter pruning via geometric median for deep convolutional neural networks acceleration,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4340–4349, 2019.
- [450] Y. Yoo, D. Han, and S. Yun, “Extd: Extremely tiny face detector via iterative filter reuse,” *arXiv preprint arXiv:1906.06579*, 2019.
- [451] J. Jeong, B. Kim, J. Yu, and Y. Yoo, “Eresfd: Rediscovery of the effectiveness of standard convolution for lightweight face detection,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 988–998, 2024.
- [452] A. Kumar, A. M. Shaikh, Y. Li, H. Bilal, and B. Yin, “Pruning filters with l1-norm and capped l1-norm for cnn compression,” *Applied Intelligence*, vol. 51, pp. 1152–1160, 2021.
- [453] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, “Soft filter pruning for accelerating deep convolutional neural networks,” *arXiv preprint arXiv:1808.06866*, 2018.
- [454] S. Yang, P. Luo, C.-C. Loy, and X. Tang, “Wider face: A face detection benchmark,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5525–5533, 2016.
- [455] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, and J. Gao, “Large language models: A survey,” 2024.
- [456] H. Touvron, L. Martin, K. R. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. M. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, and D. Esiobu, “Llama 2: Open foundation and fine-tuned chat models,” *ArXiv*, vol. abs/2307.09288, 2023.
- [457] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. Gonzalez, and I. Stoica, “Judging llm-as-a-judge with mt-bench and chatbot arena,” *ArXiv*, vol. abs/2306.05685, 2023.
- [458] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de Las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, “Mistral 7b,” *ArXiv*, vol. abs/2310.06825, 2023.
- [459] S. Singapuram and F. Lai, “Swan: A neural engine for efficient dnn training on smartphone socs,” in *EEE Working Conference on Mining Software Repositories*, 2022.
- [460] M. Abdin, J. Aneja, and ebastien Bubeck, “Phi-2: The surprising power of small language models,” tech. rep., Microsoft Research, 2023.
- [461] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2015.
- [462] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, “Optimal distributed online prediction using mini-batches,” *Journal of Machine Learning Research (JMLR)*, vol. 13, pp. 165–202, 2012.
- [463] R. M. Schmidt, F. Schneider, and P. Hennig, “Descending through a crowded valley - benchmarking deep learning optimizers,” in *ICML*, 2021.



- [464] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.
- [465] J. Zhuang, T. Tang, Y. Ding, S. C. Tatikonda, N. C. Dvornek, X. Papademetris, and J. S. Duncan, “Adabelief optimizer: Adapting stepsizes by the belief in observed gradients,” in *NeurIPS*, 2020.
- [466] Y. Wang, Y.-Y. Kang, C. Qin, Y. Xu, H. Wang, Y. Zhang, and Y. R. Fu, “Adapting stepsizes by momentumized gradients improves optimization and generalization,” *Currently under review for ICLR 2022*, vol. abs/2106.11514, 2021.
- [467] Z. Qu, Y. Ye, and Z. Zhou, “Diagonal preconditioning: Theory and algorithms,” *CoRR*, 2020.
- [468] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Ng, “Reading digits in natural images with unsupervised feature learning,” 2011.
- [469] A. Krizhevsky, “Learning multiple layers of features from tiny images,” 2009.
- [470] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.
- [471] G. Huang, Z. Liu, and K. Q. Weinberger, “Densely connected convolutional networks,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, 2017.
- [472] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.
- [473] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” *ICML 2019*, vol. abs/1905.11946, 2019.
- [474] G. Ritchie, “Some empirical criteria for attributing creativity to a computer program,” *Minds and Machines*, vol. 17, pp. 76–99, 2007.
- [475] P. Galanter, “Artificial intelligence and problems in generative art theory,” in *Proceedings of the Conference on Electronic Visualisation & the Arts*, pp. 112–118, 2019.
- [476] M. Alfonseca, M. Cebrián, and A. De la Puente, “A simple genetic algorithm for music generation by means of algorithmic information theory,” in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 3035–3042, 2007.
- [477] P. Machado, J. Romero, M. Nadal, A. Santos-del Riego, J. Correia, and A. Carballal, “Computerized measures of visual complexity,” *Acta Psychologica*, vol. 160, pp. 43–57, 2015.
- [478] J. Secretan, N. Beato, D. D’Ambrosio, A. Rodriguez, A. Campbell, and K. Stanley, “Picbreeder: evolving pictures collaboratively online,” in *Proceeding of the SIGCHI conference on Human factors in Computing Systems*, 2008.
- [479] A. Liapis, G. Yannakakis, and J. Togelius, “Adapting models of visual aesthetics for personalized content creation,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 3, pp. 213–228, 2012.



- [480] H. Takagi, “Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation,” *Proceedings of the Institute of Electrical and Electronics Engineers*, vol. 89, no. 9, pp. 1275–1296, 2001.
- [481] A. Hoover, P. Szerlip, and K. Stanley, “Interactively evolving harmonies through functional scaffolding,” in *Proceedings of the Genetic and evolutionary Computation Conference*, 2011.
- [482] C. Johnson, “Stepwise evolutionary learning using deep learned guidance functions,” in *Proceedings of the International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pp. 50–62, Springer International Publishing, 2019.
- [483] B. Roziere, F. Teytaud, V. Hosu, H. Lin, J. Rapin, M. Zameshina, and O. Teytaud, “EvolGAN: Evolutionary generative adversarial networks,” in *Proceedings of the Asian Conference on Computer Vision*, 2021.
- [484] A. Radford, J. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, “Learning transferable visual models from natural language supervision,” in *International Conference on Machine Learning*, pp. 8748–8763, PMLR, 2021.
- [485] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, “Simple and controllable music generation,” *arXiv preprint arXiv:2306.05284*, 2023.
- [486] R. Girdhar, A. El-Nouby, Z. Liu, M. Singh, K. Alwala, A. Joulin, and I. Misra, “ImageBind: One embedding space to bind them all,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [487] J. Mouret and J. Clune, “Illuminating search spaces by mapping elites,” *arXiv preprint arXiv:1504.04909*, 2015.
- [488] M. Fontaine and S. Nikolaidis, “Differentiable quality diversity,” in *Proceedings of the Neural Information Processing Systems Conference*, 2021.
- [489] D. Gravina, A. Khalifa, A. Liapis, J. Togelius, and G. Yannakakis, “Procedural content generation through quality-diversity,” in *Proceedings of the IEEE Conference on Games*, 2019.
- [490] B. Viana, L. Pereira, and C. Toledo, “Illuminating the space of enemies through MAP-Elites,” in *Proceedings of the IEEE Conference on Games*, 2022.
- [491] A. Alvarez, S. Dahlskog, J. Font, and J. Togelius, “Empowering quality diversity in dungeon design with interactive constrained MAP-Elites,” in *Proceedings of the IEEE Conference on Games*, 2019.
- [492] A. Alvarez and J. Font, “TropeTwist: Trope-based narrative structure generation,” in *Proceedings of the Foundations of Digital Games conference*, 2022.
- [493] S. Colton, “Evolving neural style transfer blends,” in *Proceedings of the International Conference on Computational Intelligence in Music, Sound, Art and Design*, 2021.
- [494] A. Liapis, G. N. Yannakakis, M. J. Nelson, M. Preuss, and R. Bidarra, “Orchestrating game generation,” *IEEE Transactions on Games*, vol. 11, no. 1, pp. 48–68, 2019.
- [495] M. Cook, S. Colton, and J. Gow, “The angelina videogame design system—part ii,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 9, no. 3, 2017.



- [496] R. Gallotta, G. Todd, M. Zammit, S. Earle, A. Liapis, J. Togelius, and G. N. Yannakakis, “Large language models and games: A survey and roadmap,” *arXiv preprint arXiv:2402.18659*, 2024.
- [497] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.
- [498] J. Dormans, “Adventures in level design: generating missions and spaces for action adventure games,” in *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, 2010.
- [499] A. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, *et al.*, “Mixtral of experts,” *arXiv preprint arXiv:2401.04088*, 2024.
- [500] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, and J. Müller, “Sdxl: Improving latent diffusion models for high-resolution image synthesis,” *arXiv preprint arXiv:2307.01952*, 2023.
- [501] L. Zhang, A. Rao, and M. Agrawala, “Adding conditional control to text-to-image diffusion models,” *arXiv preprint arXiv:2302.05543*, 2023.
- [502] M. Zammit, A. Liapis, and G. N. Yannakakis, “Map-elites with transverse assessment for multimodal problems in creative domains,” in *Proceedings of the 6th International Conference on Computational Intelligence in Music, Sound, Art and Design (EvoMusArt)*, 2024.
- [503] T. T. Nguyen, *Continuous dynamic optimisation using evolutionary algorithms*. PhD thesis, University of Birmingham, 2011.
- [504] T. T. Nguyen and X. Yao, “Dynamic time-linkage problems revisited,” in *Applications of Evolutionary Computing* (M. Giacobini, A. Brabazon, S. Cagnoni, G. A. D. Caro, A. Ekárt, A. I. Esparcia-Alcázar, M. Farooq, A. Fink, and P. Machado, eds.), 2009.
- [505] H. G. Cobb and J. J. Grefenstette, “Genetic algorithms for tracking changing environments,” in *Proceedings of the International Conference on Genetic Algorithms*, 1993.
- [506] J. J. Grefenstette, “Evolvability in dynamic fitness landscapes: a genetic algorithm approach,” in *Proceedings of the Congress on Evolutionary Computation*, vol. 3, 1999.
- [507] C. Cui, T. Feng, N. Yang, and J. Chen, “Memory based differential evolution algorithms for dynamic constrained optimization problems,” in *International Conference on Computational Intelligence and Security*, 2015.
- [508] T. T. Nguyen and X. Yao, “Benchmarking and solving dynamic constrained problems,” in *IEEE Congress on Evolutionary Computation*, 2009.
- [509] T. T. Nguyen and X. Yao, “Continuous dynamic constrained optimization—the challenges,” *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 6, 2012.
- [510] M. B. Abello, L. T. Bui, and Z. Michalewicz, “An adaptive approach for solving dynamic scheduling with time-varying number of tasks — part I,” in *IEEE Congress of Evolutionary Computation*, 2011.

- [511] Y. Jin, M. Olhofer, and B. Sendhoff, “On evolutionary optimization of large problems using small populations,” in *Advances in Natural Computation* (L. Wang, K. Chen, and Y. S. Ong, eds.), 2005.
- [512] T. T. Nguyen, S. Yang, and J. Branke, “Evolutionary dynamic optimization: A survey of the state of the art,” *Swarm and Evolutionary Computation*, vol. 6, 2012.
- [513] H. S. Urade and R. Patel, “Dynamic particle swarm optimization to solve multi-objective optimization problem,” *Procedia Technology*, vol. 6, 2012.
- [514] M. Flageat and A. Cully, “Fast and stable MAP-Elites in noisy domains using deep grids,” in *Proceedings of the Artificial Life Conference*, 2020.
- [515] M. C. Fontaine, S. Lee, L. B. Soros, F. D. M. Silva, J. Togelius, and A. K. Hoover, “Mapping Hearthstone deck spaces through MAP-elites with sliding boundaries,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2019.
- [516] R. Gallotta, K. Arulkumaran, and L. B. Soros, “Preference-learning emitters for mixed-initiative quality-diversity algorithms,” *IEEE Transactions on Games*, 2023.
- [517] F. Otto and B. Rasch, *Finding Form: Towards an Architecture of the Minimal*. Axel Menges, 1996.
- [518] M. Burry, *The Expiatory Church of the Sagrada Família*. Phaidon Press, 1993.
- [519] D. Piker, “Kangaroo: Form finding with computational physics,” *Architectural Design*, vol. 83, no. 2, 2013.
- [520] R. Vierlinger, *Multi Objective Design Interface*. PhD thesis, TU Wien, 2013.
- [521] M. Kyropoulou, “Shading design for outdoor learning in warm and hot climates using evolutionary computation: A case study in houston tx,” in *Proceedings of the Annual Modeling and Simulation Conference (ANNSIM)*, pp. 682–693, 2022.
- [522] C. Cui, H. Ohmori, and M. Sasaki, “Computational morphogenesis of 3d structures by extended eso method,” *Journal of the International Association for Shell and Spatial Structures*, vol. 44, no. 1, 2003.
- [523] A. Fabris, A. Esuli, A. Moreo, and F. Sebastiani, “Measuring fairness under unawareness of sensitive attributes: A quantification-based approach,” *Journal of Artificial Intelligence Research*, vol. 76, pp. 1117–1180, 2023.
- [524] A. Esuli, A. Fabris, A. Moreo, and F. Sebastiani, *Learning to quantify*. Cham, CH: Springer Nature, 2023.
- [525] P. González, A. Castaño, N. V. Chawla, and J. J. del Coz, “A review on quantification learning,” *ACM Computing Surveys*, vol. 50, no. 5, pp. 74:1–74:40, 2017.
- [526] J. G. Moreno-Torres, T. Raeder, R. Alaíz-Rodríguez, N. V. Chawla, and F. Herrera, “A unifying view on dataset shift in classification,” *Pattern Recognition*, vol. 45, no. 1, pp. 521–530, 2012.
- [527] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, eds., *Dataset shift in machine learning*. Cambridge, US: The MIT Press, 2009.

- [528] J. Barranquero, J. Díez, and J. J. del Coz, “Quantification-oriented learning based on reliable classifiers,” *Pattern Recognition*, vol. 48, no. 2, pp. 591–604, 2015.
- [529] A. Bella, C. Ferri, J. Hernández-Orallo, and M. J. Ramírez-Quintana, “Quantification via probability estimators,” in *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM 2010)*, (Sydney, AU), pp. 737–742, 2010.
- [530] A. Esuli, A. Moreo, and F. Sebastiani, “A recurrent neural network for sentiment quantification,” in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM 2018)*, (Torino, IT), pp. 1775–1778, 2018.
- [531] G. Forman, “Quantifying counts and costs via classification,” *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 164–206, 2008.
- [532] W. Hassan, A. G. Maletzke, and G. E. Batista, “Accurately quantifying a billion instances per second,” in *Proceedings of the 7th IEEE International Conference on Data Science and Advanced Analytics (DSAA 2020)*, (Sydney, AU), pp. 1–10, 2020.
- [533] L. Milli, A. Monreale, G. Rossetti, F. Giannotti, D. Pedreschi, and F. Sebastiani, “Quantification trees,” in *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM 2013)*, (Dallas, US), pp. 528–536, 2013.
- [534] A. Moreo and F. Sebastiani, “Tweet sentiment quantification: An experimental re-evaluation,” *PLOS ONE*, vol. 17, pp. 1–23, September 2022.
- [535] P. Pérez-Gállego, A. Castaño, J. R. Quevedo, and J. J. del Coz, “Dynamic ensemble selection for quantification tasks,” *Information Fusion*, vol. 45, pp. 1–15, 2019.
- [536] T. Schumacher, M. Strohmaier, and F. Lemmerich, “A comparative evaluation of quantification methods.” arXiv:2103.03223v1 [cs.LG], 2021.
- [537] B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. M. Mooij, “On causal and anticausal learning,” in *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, (Edinburgh, UK), 2012.
- [538] T. Fawcett and P. Flach, “A response to Webb and Ting’s ‘On the application of ROC analysis to predict classification performance under varying class distributions’,” *Machine Learning*, vol. 58, no. 1, pp. 33–38, 2005.
- [539] A. Storkey, “When training and test sets are different: Characterizing learning transfer,” in *Dataset shift in machine learning* (J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, eds.), pp. 3–28, Cambridge, US: The MIT Press, 2009.
- [540] P. González, A. Moreo, and F. Sebastiani, “Binary quantification and dataset shift: An experimental investigation,” *Data Mining and Knowledge Discovery*, 2024. Forthcoming.
- [541] P. González, A. Castaño, E. E. Peacock, J. Díez, J. J. Del Coz, and H. M. Sosik, “Automatic plankton quantification using deep features,” *Journal of Plankton Research*, vol. 41, no. 4, pp. 449–463, 2019.
- [542] O. Beijbom, J. Hoffman, E. Yao, T. Darrell, A. Rodriguez-Ramirez, M. Gonzalez-Rivero, and O. Hoegh-Guldberg, “Quantification in-the-wild: Data-sets and baselines.” arXiv:1510.04811 [cs.LG], 2015.

- [543] D. Kottke, C. Sandrock, G. Kreml, and B. Sick, “A stopping criterion for transductive active learning,” in *Proceedings of the 33rd European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML / PKDD 2022)*, (Grenoble, FR), pp. 468–484, 2022.
- [544] A. Moreo, A. Esuli, and F. Sebastiani, “Distributional random oversampling for imbalanced text classification,” in *Proceedings of the 39th ACM Conference on Research and Development in Information Retrieval (SIGIR 2016)*, (Pisa, IT), pp. 805–808, 2016.
- [545] D. Card and N. A. Smith, “The importance of calibration for estimating proportions from annotations,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2018)*, (New Orleans, US), pp. 1636–1646, 2018.
- [546] A. Esuli and F. Sebastiani, “Optimizing text quantifiers for multivariate loss functions,” *ACM Transactions on Knowledge Discovery and Data*, vol. 9, no. 4, p. Article 27, 2015.
- [547] A. Moreo and F. Sebastiani, “Re-assessing the “classify and count” quantification method,” in *Proceedings of the 43rd European Conference on Information Retrieval (ECIR 2021)*, vol. II, (Lucca, IT), pp. 75–91, 2021.
- [548] G. Forman, “Counting positives accurately despite inaccurate classification,” in *Proceedings of the 16th European Conference on Machine Learning (ECML 2005)*, (Porto, PT), pp. 564–575, 2005.
- [549] V. González-Castro, R. Alaiz-Rodríguez, L. Fernández-Robles, R. Guzmán-Martínez, and E. Alegre, “Estimating class proportions in boar semen analysis using the Hellinger distance,” in *Proceedings of the 23rd International Conference on Industrial Engineering and other Applications of Applied Intelligent Systems (IEA/AIE 2010)*, (Cordoba, ES), pp. 284–293, 2010.
- [550] M. C. du Plessis and M. Sugiyama, “Semi-supervised learning of class balance under class-prior change by distribution matching,” *Neural Networks*, vol. 50, pp. 110–119, 2014.
- [551] A. Maletzke, D. Moreira dos Reis, E. Cherman, and G. Batista, “DyS: A framework for mixture models in quantification,” in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*, (Honolulu, US), pp. 4552–4560, 2019.
- [552] B. Dussap, G. Blanchard, and B. Chérif-Abdellatif, “Label shift quantification with robustness guarantees via distribution feature matching,” in *Proceedings of the 34th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML / PKDD 2023)*, (Torino, IT), pp. 69–85, 2023.
- [553] V. González-Castro, R. Alaiz-Rodríguez, and E. Alegre, “Class distribution estimation based on the Hellinger distance,” *Information Sciences*, vol. 218, pp. 146–164, 2013.
- [554] A. Esuli, A. Moreo, F. Sebastiani, and G. Sperduti, “A detailed overview of LeQua 2022: Learning to quantify,” in *Working Notes of the 13th Conference and Labs of the Evaluation Forum (CLEF 2022)*, (Bologna, IT), 2022.
- [555] A. Moreo, P. González, and J. J. del Coz, “Kernel density estimation for multiclass quantification,” *Machine Learning*, 2024. Submitted for publication.

- [556] L. Qi, M. Khaleel, W. Tavanapong, A. Sukul, and D. A. M. Peterson, “A framework for deep quantification learning,” in *Proceedings of the 2020 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2020)*, (Ghent, BE), pp. 232–248, 2020.
- [557] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. R. Salakhutdinov, and A. J. Smola, “Deep sets,” in *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*, (Long Beach, US), pp. 3391–3401, 2017.
- [558] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, “Set transformer: A framework for attention-based permutation-invariant neural networks,” in *International conference on machine learning*, pp. 3744–3753, PMLR, 2019.
- [559] A. Esuli, A. Moreo, F. Sebastiani, and G. Sperduti, “A detailed overview of LeQua@CLEF 2022: Learning to quantify,” in *Proceedings of the Working Notes of CLEF 2022 - Conference and Labs of the Evaluation Forum, Bologna, Italy, September 5th-8th, 2022*, vol. 3180 of *CEUR Workshop Proceedings*, (Bologna, Italy), pp. 1849–1868, CEUR-WS.org, 2022.
- [560] M. Avi-Aharon, A. Arbelle, and T. R. Raviv, “Deephist: Differentiable joint and color histogram layers for image-to-image translation,” *arXiv preprint arXiv:2005.03995*, 2020.
- [561] J. Peeples, W. Xu, and A. Zare, “Histogram layers for texture analysis,” *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 4, pp. 541–552, 2022.
- [562] Z. Wang, H. Li, W. Ouyang, and X. Wang, “Learnable histogram: Statistical context features for deep neural networks,” in *European Conference on Computer Vision*, pp. 246–262, Springer, 2016.
- [563] I. Yusuf, G. Igwegbe, and O. Azeez, “Differentiable histogram with hard-binning,” *arXiv preprint arXiv:2012.06311*, 2020.
- [564] O. Pérez-Mon, A. Moreo, J. J. del Coz, and P. González, “Quantification using permutation-invariant networks based on histograms,” *Neural Computing and Applications*, 2023. Submitted for publication.
- [565] D. Tasche, “Class prior estimation under covariate shift: No problem?,” in *Proceedings of the 2nd International Workshop on Learning to Quantify: Methods and Applications (LQ 2022)*, *ECML/PKDD*, (Grenoble (France)), 2022.
- [566] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang, “Domain adaptation under target and conditional shift,” in *ICML*, pp. 819–827, 2013.
- [567] Z. C. Lipton, Y. Wang, and A. J. Smola, “Detecting and correcting for label shift with black box predictors,” in *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, (Stockholm, SE), pp. 3128–3136, 2018.
- [568] M. Bunse and K. Morik, “Unification of algorithms for quantification and unfolding,” in *Proceedings of the 2nd International Workshop on Learning to Quantify (LQ 2022)*, (Grenoble, IT), pp. 1–10, 2022.
- [569] M. Bunse, “On multi-class extensions of adjusted classify and count,” in *Proceedings of the 2nd International Workshop on Learning to Quantify (LQ 2022)*, (Grenoble, IT), pp. 43–50, 2022.

- [570] T. Jaenich, A. Moreo, A. Fabris, G. McDonald, A. Esuli, I. Ounis, and F. Sebastiani, “Quantifying query fairness under unawareness,” in *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM 2024)*, (Boise, US), 2024. Submitted for publication.
- [571] D. Bacciu, F. Errica, A. Micheli, and M. Podda, “A gentle introduction to deep learning for graphs,” *Neural Networks*, vol. 129, pp. 203–221, 2020.
- [572] L. Tang, H. Gao, and H. Liu, “Network quantification despite biased labels,” in *Proceedings of the 8th Workshop on Mining and Learning with Graphs (MLG 2010)*, (Washington, US), pp. 147–154, 2010.
- [573] A. Micheli and D. Tortorella, “Addressing heterophily in node classification with graph echo state networks,” *Neurocomputing*, vol. 550, p. 126506, 2023.
- [574] J. C. Platt, “Probabilistic outputs for support vector machines and comparison to regularized likelihood methods,” in *Advances in Large Margin Classifiers* (A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, eds.), pp. 61–74, Cambridge, MA: The MIT Press, 2000.
- [575] M. Saerens, P. Latinne, and C. Decaestecker, “Adjusting the outputs of a classifier to new a priori probabilities: A simple procedure,” *Neural Computation*, vol. 14, no. 1, pp. 21–41, 2002.
- [576] D. Tasche, “Fisher consistency for prior probability shift,” *Journal of Machine Learning Research*, vol. 18, pp. 95:1–95:32, 2017.
- [577] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society, B*, vol. 39, no. 1, pp. 1–38, 1977.
- [578] A. Micheli, A. Moreo, M. Podda, F. Sebastiani, W. Simoni, and D. Tortorella, “Learning to quantify graph nodes,” in *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, (Vancouver, CA), 2024. Submitted for publication.
- [579] S. Garg, S. Balakrishnan, Z. C. Lipton, B. Neyshabur, and H. Sedghi, “Leveraging unlabeled data to predict out-of-distribution performance,” in *Proceedings of the 10th International Conference on Learning Representations (ICLR 2022)*, (Virtual Event), 2022.
- [580] L. Volpi, A. Esuli, A. Moreo, and F. Sebastiani, “Using quantification to predict classifier accuracy under prior probability shift,” in *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, (Vancouver, CA), 2024. Submitted for publication.
- [581] M. Bunse, P. González, A. Moreo, and F. Sebastiani, eds., *Proceedings of the 3rd International Workshop on Learning to Quantify (LQ 2023)*. 2023.
- [582] M. Bunse, P. González, A. Moreo, and F. Sebastiani, “Report on the 3rd International Workshop on Learning to Quantify (LQ 2023),” *SIGKDD Explorations*, vol. 25, no. 2, pp. 25–28, 2023.
- [583] M. Bunse, A. Moreo, F. Sebastiani, and M. Senz, “Ordinal quantification through regularization,” in *Proceedings of the 33rd European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML / PKDD 2022)*, (Grenoble, FR), pp. 36–52, 2022.

- [584] M. Bunse, A. Moreo, F. Sebastiani, and M. Senz, “Regularization-based methods for ordinal quantification,” *Data Mining and Knowledge Discovery*, 2024. Forthcoming.
- [585] J. J. del Coz, P. González, A. Moreo, and F. Sebastiani, eds., *Proceedings of the 2nd International Workshop on Learning to Quantify (LQ 2022)*. 2022.
- [586] M. Nägele and F. Marquardt, “Optimizing zx-diagrams with deep reinforcement learning,” 2023.
- [587] B. Coecke and A. Kissinger, *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, 2017.
- [588] J. van de Wetering, “Zx-calculus for the working quantum computer scientist,” 2020.
- [589] N. de Beaudrap, A. Kissinger, and J. van de Wetering, “Circuit extraction for zx-diagrams can be p-hard,” in *Schloss Dagstuhl – Leibniz-Zentrum für Informatik*, 2022.

