AI4media

ARTIFICIAL INTELLIGENCE FOR THE MEDIA AND SOCIETY

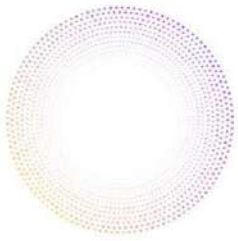
D5.3

Report on Computationally Demanding Deep Learning

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	Report on Computationally Demanding Deep Learning
Deliverable number	D5.3
Deliverable version	1.0
Previous version(s)	-
Contractual date of delivery	August 31, 2023
Actual date of delivery	September 6, 2023
Deliverable filename	AI4Media_D5.3.pdf
Nature of deliverable	Report
Dissemination level	Public
Number of pages	68
Work Package	WP5
Task(s)	T5.5
Parner responsible	BSC
Author(s)	Dario Garcia-Gasulla, Adrian Tormos, Enrique Lopez (BSC), Nicu Sebe (UNITN), Lorenzo Seidenari (UNIFI), Alberto Messina, Roberto Iacoviello (RAI)
Editor	Adrian Tormos (BSC)
Project Officer	Evangelia Markidou

Abstract	This deliverable presents the initial research outcomes of AI4Media research activities related to task 5.5, "Computationally demanding learning". The document presents research advances on the topics of image and video synthetic upscaling, image and video synthetic enhancing detection, video quality assessment, super-resolution model evaluation, self-supervised learning, efficiency improvement of Visual Transformer training, and efficient computation of matrix operations. We also present relevant publications and links to software and discuss the research outcomes' relevance to AI4Media use cases. Lastly, we discuss the partners' ongoing work and future research plans in the context of the task.

Keywords	Multimedia, artificial intelligence, content enhancement, synthetic upscaling, synthetic content detection, media quality assessment, Visual Transformers, self-supervised learning, efficient training, efficient matrix operations, eigendecomposition

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

Copyright

Contributors

NAME	ORGANIZATION
Dario Garcia-Gasulla	BSC
Adrian Tormos	BSC
Enrique Lopez	BSC
Nicu Sebe	UNITN
Lorenzo Seidenari	UNIFI
Alberto Messina	RAI
Roberto Iacoviello	RAI

Peer Reviews

NAME	ORGANIZATION
Fulvio Negro	RAI
Maurizio Montagnuolo	RAI
Julien Tourille	CEA

Revision History

Version	Date	Reviewer	Modifications
0.1	10/05/2023	Adrian Tormos (BSC)	Initial version, Table of contents and sections
0.2	21/07/2023	Adrian Tormos (BSC)	Pre-final version, ready for internal review
0.3	04/08/2023	Fulvio Negro, Maurizio Montagnuolo (RAI), Julien Tourille (CEA)	Internal review
1.0	06/09/2023	Adrian Tormos (BSC)	Final version, ready for submission

Table of Abbreviations and Acronyms

Abbreviation	Meaning
AI	Artificial Intelligence
BN	Batch Normalization
BSRGAN	Blind Super-Resolution Generative Adversarial Network
BTC	Basic test cell
CF	Colourfulness
CHI	Calinski-Harabaz Index
CNN	Convolutional Neural Networks
DC	Divide-and-Conquer
DCT	Discrete Cosine Transform
DL	Deep Learning
DSLR	Digital Single-Lens Reflex
ED	Eigendecomposition
GDPR	General Data Protection Regulation
GLCM	Grey-level Co-ocurrence Matrix
HR	High-Resolution
LR	Low-Resolution
MAE	Mean Absolute Error
MLP	Multilayer Perceptron
MOS	Mean Opinion Score
MPA	Matrix Padé Approximants
MSE	Mean Squared Error
MTP	Matrix Taylor Polynomial
NLP	Natural Language Processing
NS	Newton-Schulz
PCA	Principal Component Analysis
PSNR	Peak Signal-to-Noise Ratio
PVS	Processed video sequences
RVRT	Recurrent Video Restoration Transformer
SI	Spatial Information
SR	Super-Resolution
SRC	Source reference sequence
SRDM	Super-Resolution Detection Method
SSIM	Structural Similarity Index
SSL	Self-Supervised representation Learning
SUDDS	Synthetic Upscaling Detector with DCT features and Staircase module
SVD	Singular Value Decomposition
t-SNE	t-Distributed Stochastic Neighbor Embedding

TI	Temporal Information
TSARA	Two-step Authentic Resolution Assessment
UHD	Ultra High-Definition
UMAP	Uniform Manifold Approximation and Projection
VMAF	Video Multimethod Assessment Fusion
VQA	Video Quality Assessment
VSR	Video Super-Resolution
VT	Visual Transformer
W-MSE	Whitening Mean Squared Error
WCT	Whitening and Coloring Transform
ZCA	Zero-phase Component Analysis

Contents

List of Tables

List of Figures

1 Executive Summary

This deliverable presents the research outcomes of Task 5.5 "Computationally Demanding Learning", during months M19 to M36 of the project, which correspond to the first 18 months of the task duration. We expose at length the motivations, developed methods and obtained results, and refer to publications and open software published by the corresponding partners. We also list the relevant contributions of each research outcome to the WP8 use cases.

Research directions in T5.5 span several topics in relation with computation-heavy learning. This includes image and video synthetic upscaling (also known as super-resolution), high-resolution image and video handling, efficient methods for training large deep neural networks and fast computation of common mathematical operations in deep learning. The results of the research in these fields have been successfully published in top conferences, and have resulted in exploitable open access implementations.

The following research outcomes have been produced, grouped in three categories, i.e. image and video enhancement, efficient neural network training, and fast mathematical computation:

- **Image and video enhancing**: We present several contributions in relation with image and video enhancement, including: a dataset to aid the training of neural networks in super-resolution-related tasks, like synthetic image and video upscaling or detection; a synthetic upscaling detector neural network; and a framework for the quality assessment of synthetic video upscaling.
- **Efficient neural network training**: We tackle the task of creating efficient neural network training methodologies, specifically proposing improvements in strategies for self-supervised learning and the training of Visual Transformers.
- **Fast mathematical computing**: We develop new, faster algorithms for mathematical matrix operations commonly used in the training of deep neural networks, including the differentiable square-root and eigendecomposition of a matrix.

We intend to extend the presented research outcomes, integrate them into WP8 use case demonstrations when possible and to further encourage joint collaboration between partners that brings mutually beneficial results.

2 Introduction

Current state-of-the-art Artificial Intelligence (AI) applications and training methods in most domains require an amount of computational resources that is not effectively obtainable by a majority of practitioners or interested industry members.

Some of the most common bottlenecks in these trainings include the need for copious amounts of data, often requiring several terabytes of free space; or large training times due to the constantly growing scale of the models, only alleviated by the use of very big quantities of GPUs. In addition, in image-based domains, like medical imaging, autonomous driving or media content managing, most current approaches downsample the images to sizes between 200×200 to 500×500 causing undesirable information losses. Handling this latter limitation is of special importance for media outlets, as 4K resolution (3840×2160 px) media is becoming the current standard.

In Task 5.5, "Computationally Demanding Learning", of AI4Media, we explore ways of efficiently handling the scaling of neural networks to larger sizes and, particularly, larger image resolutions. We research architectures that can handle high-resolution images and, in addition, we put special interest in the task of image and video synthetic upscaling (*i.e.* synthetically increasing the size of an image or video), also called Super-Resolution (SR). We also research efficient training methods for large neural networks and efficient algorithms that can be applied to deep learning.

Note that while the original Description of the Action stated that the focus of Task 5.5 was on SR, we have expanded its scope to embrace other aspects of computationally demanding learning too.

During the first 18 months of the task's duration, research focused on the following topics:

- **Image and video super-resolution**. Handling, producing and emitting media in 4K resolution is becoming more adopted by the day, making it crucial that artificial intelligence models can handle this resolution. It is also of particular interest to research the task of super-resolution as a possible way of alleviating the computational demands that come with handling 4K resolution data. In subsection 3.1, BSC presents a video dataset that is recorded simultaneously in both 1080p and 4K resolutions with the purpose of expanding the available data in high-resolution for the tasks of image and video super-resolution. Expanding the dataset with synthetically upscaled videos with SR models allows the dataset to also be used as a benchmark in SR detection. In subsection 3.2, BSC uses a neural network that joins low and high-level features with Discrete Cosine Transform features to detect synthetically upscaled images and videos, having competitive performance compared with state-of-the-art methods. The detector is able to handle arbitrarily high resolutions, as it works with patches of the input images. In subsection 3.3, RAI shows its progress on defining a SR model selection scheme according to the video's domain, motion complexity and resolution. The selection scheme includes both an objective and subjective assessment of the SR model candidates in the considered categories.
- **Efficient training methods for large neural networks**. Most of the current state-of-the-art deep learning models are very large, and require very powerful computational devices to be trained. Researching ways of training models more efficiently helps alleviate these computational needs and helps bring state-of-the-art AI to more practitioners. Self-Supervised representation Learning (SSL) methods generally train a neural network by making it contrast images of one type (positives) with the rest (negatives), to learn what differentiates them. In subsection 4.1, UNITN propose a new SSL algorithm that needs less negative samples than usual, making training more efficient. It is also competitive with the current state-of-the-art. In subsection 4.2, UNITN propose an algorithm to train Visual Transformers (VT) that includes an auxiliary task that helps maintain performance when training with small amounts

of data. Their experimentation show that this task improves the training and generalisation abilities of VTs.

- **Efficient mathematical computations**. The principle and inverse square roots of a positive semi-definite matrix, as well as the eigendecomposition (ED) of a matrix, are calculations that are performed in several computer vision related applications, making finding fast and reliable solutions for them particularly interesting. In subsection 5.1, UNITN present two more efficient variants of the current used methods for calculating the principle and inverse square root, and demonstrate that they give considerable speed-up while being as effective. In subsection 5.2, they propose an algorithm for ED for mini-batches of small and medium matrices, purposed for batch efficiency in deep learning applications, and also show that it achieves competitive performance against current ED implementations.

These works are described, separated by topic, in section 3, section 4 and section 5 respectively. Their description includes links to relevant publications and open software published by the AI4Media partners, and an explanation on how their contributions connect with AI4Media use cases and requirements. Lastly, section 6 describes the plans of the partners for future work and extensions in relation to T5.5 and concludes the deliverable.

3 Image and video super-resolution

According to the Visual Networking Index by Cisco [7], it is estimated that by 2023, two-thirds of the installed flat-panel TV sets will have 4K resolution (3840×2160 px) or larger. The continuously increasing adoption of higher resolution data is accompanied by media outlets, who have started producing and emitting content in 4K.

Due to the fact that handling high-resolution images is very computationally demanding, state-of-the-art approaches in some image-related domains still work with data that is downsampled to resolutions that range from 200×200 to 600×600 pixels [8, 9], suffering information loss in the process. Nonetheless, as the world turns to higher and higher resolutions, it is important that AI methods and models do not fall behind and evolve accordingly to handle images in this resolution.

The tendency towards 4K content has also caused technology companies to turn to SR techniques to upscale lower-resolution content to 4K. Some examples include televisions[1], smartphones[2], external hardware devices[3,4], or software-based solutions which require modern GPUs[5,6]. Moreover, film production companies have widely embraced the practice of artificial upscaling for movies in recent years. The main reason is saving resources, as the time and money required to record and edit directly in 4K is substantially higher. In addition to the regular release of new films, film studios and distributors often seize the opportunity to revisit and remaster older movies for re-release in 4K resolution. SR algorithms have allowed content that was originally in a smaller resolution like 1080p (1920×1080 px) or 720p (1280×720 px) to meet the viewer's expectations for detail and clarity, considering the limitations of SR networks.

In this section, we provide three research outcomes related to the handling of high-resolution image and video data. In subsection 3.1, BSC presents a dataset recorded in 1080p and 4K resolutions at the same time, that expands the current available data in those resolutions for SR-related tasks. In subsection 3.2, BSC presents a detector architecture for upscaled content via SR methods. The architecture is tested on videos in 4K, in which it achieves competitive results compared to state-of-the-art methods and a superior capability to detect previously unseen SR methods. In subsection 3.3, RAI defines a SR neural network quality assessment and selection scheme that includes an objective and subjective assessment of the neural network candidates.

3.1 BSC4K: A 4K and 1080p video dataset for super-resolution tasks

Contributing partners: BSC

Digital content manipulation techniques, such as deepfakes, automatic colorization, or generative models, have found numerous applications in recent years. Among these is SR, which aims to increase the resolution of lower quality images or videos and enhance the fine details that are missing in the Low-Resolution (LR) source. Image enhancing applications are being applied in medical imaging [10, 11, 12], security camera image footage [13, 14], remote sensing tasks [15, 16], gaming [17], and the entertainment industry [18, 19].

Training a supervised SR model requires of paired low-resolution and high-resolution (LR-HR) samples that belong to the same image. The network will learn how to create the HR sample given only the LR input. To obtain these LR-HR pairs, most contributions downscale a set of the HR images synthetically. The operation used to get LR images is a key factor that heavily impacts

[1] https://www.youtube.com/watch?v=ZS46kYeMx00

[2] https://www.samsungmobilepress.com/feature-stories/how-samsung-galaxy-cameras-combine-super-resolution-technologies-with-ai-technology-to-produce-high-quality-images-of-the-moon/

[3] https://www.apple.com/es/apple-tv-4k/

[4] https://www.nvidia.com/en-us/shield/support/shield-tv/ai-upscaling/

[5] https://blogs.nvidia.com/blog/2023/02/28/rtx-video-super-resolution/

[6] https://www.topazlabs.com/gigapixel-ai

Category	People	Vegetation	Text	Vehicles	Animals	Buildings	Textures	Close-up	Blurry	Focus change	Zoom
No. of videos	4	20	1	5	1	10	5	4	1	2	5

Table 2. Current coverage of different content texture categories for BSC4K.

the model's ability to perform in real-world scenarios. If one simple downsampling operation is applied to generate all LR training samples, the model will specialize in handling the resulting artifacts of that specific downscaling process [20, 21, 22, 23]. Commonly used downscaling methods range from using simple algorithms like bicubic interpolation [24, 25, 26, 27], to the use of more complex pipelines that try to emulate real-world degradations to mitigate degradation specialisation [23, 28, 29]. Training strategies that use the former are called blind, while the ones that use the latter are called non-blind.

The continuous adoption of high-resolution 4K (3840×2160 px) content has lead to multimedia-related companies accompanying this change by using image and video SR techniques to upscale the content to 4K from lower resolutions like 720p (1280×720 px) or 1080p (1920×1080 px)[7,8,9].

As in any other computer vision task, the quality and diversity of the datasets are integral to the success of SR, especially as we transition towards high-definition formats such as 4K. Most existing SR methods require LR-HR image pairs for training and evaluation, and most of the existing datasets generate the LR image synthetically. There are several paired real-world datasets, but there is a lack of accepted guidelines within academia for training and evaluation of higher-resolution (4K) content.

3.1.1 The BSC4K dataset

In an effort to expand the currently available data on 4K SR, we present the BSC4K dataset (work in progress), which contains paired video sequences at 1080p and 4K resolution recorded simultaneously. The motivation of the dataset is to overcome the challenges introduced by artificially downscaling HR content to obtain the LR counterparts. Having both the original LR recording and the capability to generate synthetic LR images allow us to study the domain gap that exists in non-blind SR methods and compare them to blind SR methods. The LR images can also be used to generate upscaled HR versions of the images, thus also allowing its use in SR detection. Note that the dataset is currently unpublished and still a work in progress. We plan to openly share it during the last year of the project.

The first version of the dataset contains 33 4K and 33 1080p videos, cut to 64 frames each, recorded indoor and outdoor with a single Digital Single-Lens Reflex (DSLR) camera. Some samples from the dataset can be seen in Figure 1. We restrict the SR problem to ×2 scaling, as the camera records at 4K and 1080p. The main advantages of our dataset are that the acquisition process is straightforward, while the post-processing primarily entails the separation of videos into individual frames.

The dataset is still being developed, and is expected to be expanded in order to include more diverse content and environmental settings. Nevertheless, we try to cover several relevant categories in this first iteration even if the environments and weather of the videos in the dataset are more similar. We include various types of motion speeds, zoom-out and zoom-in, blur and a variety of content types, as shown in Table 2. We include scenes with minimal and high movement, as well as sequences involving panning shots and sequences that incorporate handheld camera movement.

[7] https://www.youtube.com/watch?v=ZS46kYeMx00
[8] https://www.apple.com/es/apple-tv-4k/
[9] https://blogs.nvidia.com/blog/2023/02/28/rtx-video-super-resolution/

Figure 1. Example frames from the BSC4K dataset.

This dataset has been generated by taking videos with a Panasonic Lumix DC-S5 in MOV containers with a H.264 codec at a resolution of 5.9k pixels. The sensor of the S5 camera is a full-frame 35.6x23.8mm CMOS sensor. The signal captured by the sensor, before being converted into H.264, was passed through to an HDMI interface towards a Blackmagic Video Assist 5" 3G, recording in a MOV container with a ProRes HQ codec at a resolution of 1080p. The lens used has been a Panasonic Lumix S 20-60mm zoom lens at varying settings of focal length, ISO (the two native ISOs: 640 and 4000), shutter speed (from 30 to 80), aperture (from 3.5 to 22) and white balance (from 3200 to 5600k).

According to the AVC-Intra documentation[10], AVC-Intra 100 records the full 1920x1080 raster, representative of master-quality recording.

After generating the .MOV files, we employ FFMPEG[11] to split the videos into frames in .png format. Due to a small difference between timecodes of 4K and 1080p videos, we methodically align the frames pairs of each video. This alignment procedure does not alter the content of the images. Instead, it ensures that the frame numbers from both videos align with the same timestamp and are consistent across both resolutions.

[10] https://resources.avid.com/SupportFiles/attach/FAQ_AVC-Intra.pdf
[11] https://ffmpeg.org/

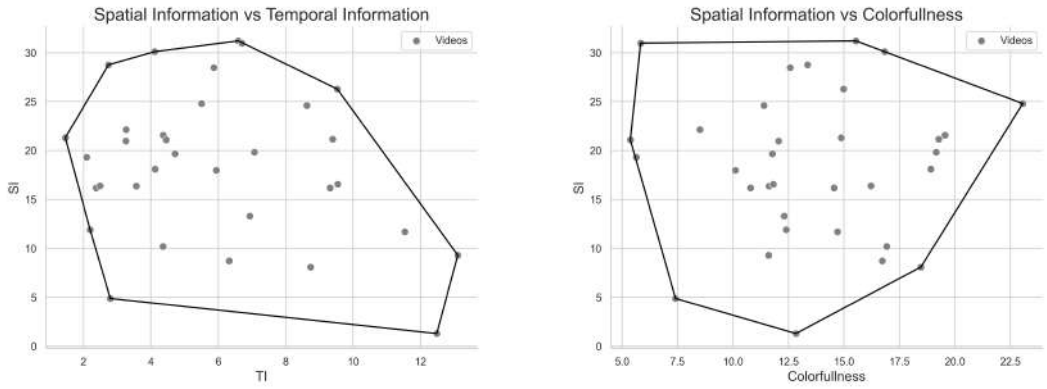


Figure 2. Illustration of the coverage of three video descriptors: Spatial Information (SI), Temporal Information (TI), and Colourfulness (CF). Each point in the scatter plot represents a single video in the dataset, mapped based on its SI, TI, and CF values. A broad coverage across these metrics indicates a diverse set of videos in terms of visual complexity and colorimetry. The greater the spread of points, the more varied and representative the dataset, making it better suited for robust video processing and quality evaluation tasks.

3.1.2 Experiments

3.1.2.1 Content description Following the method proposed by Winkler [30], we characterise the video sequences by using three descriptors: Spatial Information (SI), Temporal Information (TI), and Colourfulness (CF). We adopt the SI and TI indicators defined in ITU-T Rec. P.910 (11/21) [31] and implemented in [12].

Figure 2 shows the current coverage of the three metrics for all videos in the dataset. This can serve as a reference for future iterations or other datasets, and can help evaluate the relation between spatial and temporal information with performance.

3.1.2.2 Deep feature analysis Prior research on image SR [32] indicated that SR models appear to discern the specific degradation types inherent in their training data. It further suggests that differences in data distribution might deactivate this discernment ability. In this work, the authors extract deep features from an SR convolutional model and projected using Principal Component Analysis (PCA) to reduce their dimensionality. The reduced feature maps are subsequently clustered through t-Distributed Stochastic Neighbor Embedding (t-SNE) [33].

We adopt a similar methodology to demonstrate the difference at feature level between synthetic degradations and the original LR recordings present in our proposed dataset. These degradations can also be used to substantiate the observations in [32] and recognize the *semantics* for modern Video Super-Resolution (VSR) networks.

Instead of PCA, we apply a mean pooling layer to condense spatial information into a single value per channel. We inevitably suffer a loss of information, but we are more interested in the activation values of the neurons and can ignore their spacial information or localization. Instead of t-SNE, we reduce dimensionality to two dimensions with UMAP [34]. To better illustrate and measure the discrimination ability, we adopt the Calinski-Harabaz Index (CHI) [35], a ratio of the mean between-cluster dispersion to the mean within-cluster dispersion.

We compare the deep features of two models that share the same backbone: BasicVSR [26] (blind) and RealBasicVSR [36] (non-blind). We compare our dataset's original LR images to

[12] https://github.com/VQEG/siti-tools

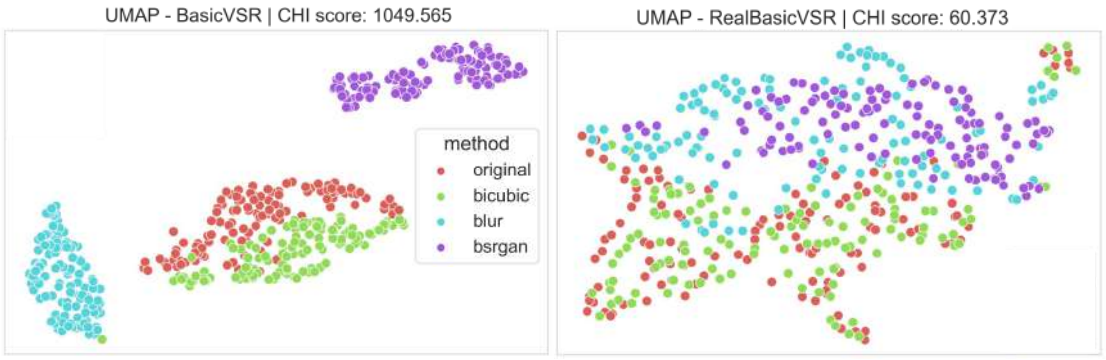


Figure 3. Deep feature representation differences between original and synthetic videos obtained by all degradation types (Left: BasicVSR, Right: Real-BasicVSR)

three synthetic degradations: bicubic interpolation, blur and Blind Super-Resolution Generative Adversarial Network (BSRGAN) [29].

In Figure 3, we can see that the features from BasicVSR allow to discriminate the different degradations, including the original LR images. It shows that the original LR images, while close to the bicubic downsampling, is a differentiated distribution. On the other hand, the figure shows that RealBasicVSR's deep features do not capture that information, also illustrated by the large difference between CHI values (1049.57 vs 60.37).

3.1.3 Dataset licensing and GDPR Compilance

We intend to make the dataset open-access to promote accessible scientific research before the end of the project. The open license will allow researchers to download, use and modify the data freely as long as it is cited correctly.

In the process of recording and saving the data, we have taken care to respect each person's privacy. We have followed the guidelines in the General Data Protection Regulation (GDPR) to ensure that our practices correspond to their standards.

3.1.4 Conclusions

- We are working on the BSC4K dataset, a dataset that pairs 1080p and 4K video sequences recorded simultaneously. This is an effort in contributing to the available resources for super-resolution tasks, specifically in 4K resolution.
- We demonstrate that the original recorded LR images in BSC4K constitute a different distribution to common synthetic degradations like bicubic interpolation, blur and BSRGAN.

3.1.5 Relevant publications

- Cuena E., "Super-resolution assessment and detection", Master thesis, July 2023.

3.1.6 Relevance to AI4Media use cases and media industry applications

Our dataset provides a series of videos that expand the available data in super-resolution related tasks in 4K. Having more available images and videos allows models to be trained more easily, contributing to the creation of better SR models that are of interest to media outlets. Because of this, our dataset contributes to the AI4Media user stories 1A4-10 (Detection of synthetic upscaling of a video) and 3B2-1 (Video super resolution), by helping create AI models that detect synthetic content and that handle videos in high resolution, respectively.

3.2 Synthetic Upscaling Detector with DCT features and Staircase module (SUDDS)

The increasing tendency to adopt the 4K resolution has led to a surge of interest in the field of SR detection and also created a new set of challenges that threaten the authenticity of visual media, especially after the recent and socially impactful development of generative models. Latent Diffusion Models for image [37] and video [38], image-to-image [39], text-to-image [40], and advanced SR methods based on Deep Learning (DL) are an example of the recent AI innovation in the digital content field. Digital forgeries, ranging from elementary manipulations like object cloning or removal to complex alterations involving deepfakes[13] and SR pose substantial issues across different sectors, including digital forensics, the legal system, media veracity, and privacy [41]. Therefore, developing effective and reliable *forgery* detection mechanisms has become paramount [42]. The urgency of this research line is clearly motivated by the widespread availability of these techniques through popular applications like Adobe's Photoshop[14], Deepfacelab[15], and TopazLab[16].

SR methods present significant benefits, such as preserving and restoring old footage or enhancing the quality of video in resource-limited contexts, but can also be misused. Ethical concerns include the use of synthetic HR images for decision-making in sensitive domains like medicine [43] or law enforcement [44]. Legal concerns include the incoming AI Act [45], which enforces the disclosure of synthetic images. This highlights the need for robust SR detection mechanisms, and especially ones that can handle high-resolution data like 4K images and videos.

To try to improve the current state-of-the-art, we propose the Synthetic Upscaling Detector with DCT features and Staircase module (SUDDS), inspired by the work of Lu et al. [6] (BTURA). Our network's architecture is based on their feature extractor, which processes small patches of interest from the training images. Choosing some patches instead of the whole image highly reduces the computational load while still being able to handle high-resolution. The feature extractor consists of a ResNet-18 (pre-trained on ImageNet [3]), where intermediate features are extracted from each block, grouped by a Global Average Pooling operation, and concatenated, as seen in Figure 4. A two-layer Multilayer Perceptron (MLP) then takes the features from the feature extractor and outputs a probability for each category. We add a Dropout layer in the MLP to avoid overfitting and keep the softmax function to the output, which transforms the values into probabilities for each class. We refer to this architecture of a feature extractor and MLP as the baseline.

We add two new modules to the architecture. Firstly, the staircase structure, proposed in [46], attempts to fully utilize the visual information from low-level to high-level and learn the better feature representations for quality evaluation. The assumption is that the bottom convolutional layers from the ResNet capture the low-level information, such as edges and corners, while the

[13] https://github.com/deepfakes/faceswap
[14] https://www.adobe.com/creativecloud/photography/discover/photo-manipulation.html
[15] https://github.com/iperov/DeepFaceLab
[16] https://www.topazlabs.com/

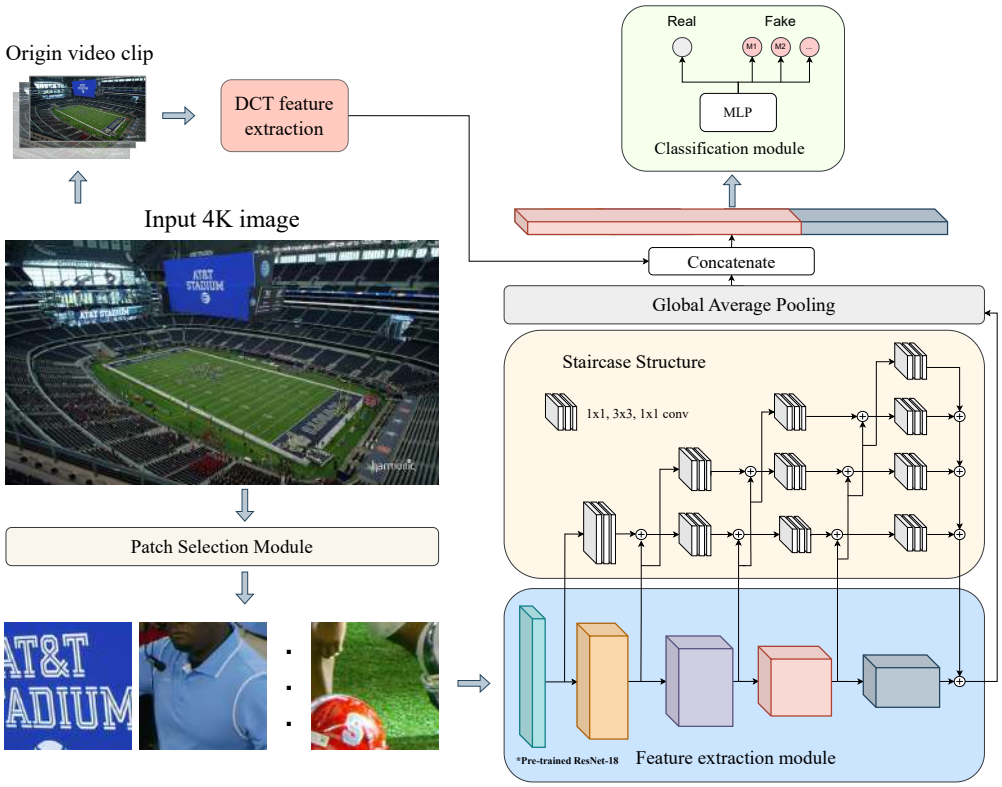


Figure 4. Network architecture of SUDDS. Original feature extraction and patch selection module from from Lu et al [6]. DCT feature extraction and Staircase Structure modules are new additions.

more advanced layers capture the semantic information. The staircase architecture hierarchically integrates low and high-level features into a final feature map that is the input to the classifier.

The second module integrates a technique to combine local features from the patches and global features from the videos. By using the same methodology described in [47], we save the Discrete Cosine Transform (DCT) features for each video in the dataset. The DCT essentially decomposes an image to its spatial frequency spectrum. At training time, those features are concatenated with the local features from the feature extractor or staircase architecture. This means that patches from the same image will contain the same global DCT-based features, but different local features. Both modules are optional and evaluated independently, allowing for a broader understanding of the learning process at hand.

Finally, the concatenated feature map is fed into a classifier, constituted by a two-layer MLP (where the intermediate layer has a size of 256), which outputs a probability value for each class. We study the setting of binary classification, where all synthetic upscaling methods are grouped together, and multiclass classification, where each method is represented by an individual label.

Please note that this research outcome is also relevant to T6.2 (Manipulation and synthetic content detection in multimedia) of AI4Media, but we have chosen to report it within T5.5 because of our focus in handling high-resolution resolution images. The training and experimentation has been entirely performed with data in 4K resolution.

Detection method	SR method: Original	Bicubic	BasicVSR	RealBasicVSR	RVRT	Nearest Neighbors	SwinIR-Classical	SwinIR-Real	Real-ESRGAN
DCT [47]	0.00	**1.00**	**1.00**	0.06	**1.00**	**1.00**	**1.00**	0.40	0.00
TSARA [51]	0.72	**1.00**	0.44	0.00	0.36	0.05	0.30	0.00	0.00
SRDM-Patches [52]	0.88	0.85	0.40	0.59	0.42	0.40	0.65	0.20	0.68
SUDDS (ours)	**0.94**	**1.00***	0.90*	**1.00***	0.80*	0.90	**1.00**	**0.90**	**0.94**

*Table 3. Accuracy Metrics for each SR method in the BSC4K dataset. Bold indicates best method in detecting a certain SR method. * denotes SR methods that are part of the training set in BVI-DVC-SR.*

3.2.1 Experiments

To validate our SUDDS model, we compare existing detection methods with our own in detecting upscaled videos with state-of-the-art SR methods. We define the task as binary classification, where a video can be classified as original or fake. We also perform an ablation study, in which we study SUDDS performance when removing the new components. Finally, we also study its generalisation ability by training it with only one SR method and evaluating it on others.

We use the BVI-DVC [48] dataset for training, which contains 200 video sequences in 4K resolution, and synthetically downscaled 1080p counterparts. We expand it by upscaling the 1080p videos with four SR methods: bicubic interpolation, BasicVSR [26] (blind), RealBasicVSR [36] (non-blind) and RVRT [27] (non-blind). We call this expanded version BVI-DVC-SR. We perform a 175/25 train/validation split.

We evaluate on our 32 BSC4K videos (dataset described in subsection 3.1), also expanded on the four aforementioned SR methods. To evaluate generalisation capabilities, we expand BSC4K with four additional methods: nearest neighbor interpolation (traditional), Real-ESRGAN (GAN-based), and SwinIR-Classical and SwinIR-Real (transformer-based).

Following the methodology in Lu *et al.* [6], we train on non-overlapping image patches of 224×224 pixels. To avoid redundancy in the training data, we only use the first frame of every video and we only select the k patches with the most complex textures using the Grey-level Co-ocurrence Matrix (GLCM) [49, 50] and classified by a majority voting mechanism based on the k individual predictions.

In all the experiments we perform a data augmentation pipeline consisting in a horizontal flip ($p = 0.35$) and a random brightness adjust of up to 20% ($p = 0.5$). We also use Coarse Dropout to encourage the network to find more patterns that identify fake images.

3.2.1.1 SUDDS detection performance We compare our proposal with three existing SR detection methods. One of them [47] is based on DCT components, so there are no learnable parameters. The other two are DL-based networks from which we take the available weights to test them with our data: Two-step Authentic Resolution Assessment (TSARA) [51] and Super-Resolution Detection Method (SRDM) [52]. We test the former directly, as it is trained with 4K images, but we modify the evaluation pipeline for the latter, which is trained with lower resolution image crops. Specifically, we crop each input 4K frame into non-overlapping 240 × 240 patches, that

Training	Staircase	DCT	Original	Nearest Neighbors	SwinIR-Classical	SwinIR-Real	Real-ESRGAN
Binary	No	No	0.61	0.95	**1.00**	0.09	**0.50**
	Yes	No	0.62	0.94	**1.00**	**0.10**	**0.50**
	Yes	Yes	**1.00**	**1.00**	0.80	**0.10**	0.21
Multiclass	No	No	0.90	0.06	**1.00**	0.50	0.50
	Yes	No	0.63	0.10	**1.00**	0.70	0.79
	Yes	Yes	**0.94**	**0.9**	**1.00**	**0.90**	**0.94**

Table 4. Accuracy comparison with module combinations in binary and multiclass classification trainings.

are further cropped to match the input of SRDM (224 × 224). Then, we aggregate the individual prediction from each patch by taking the mean value to get a frame level prediction. To get the final video-level prediction, we average the frame-level scores.

We train our model with the BVI-DVC-SR dataset, and evaluate over our expanded BSC4K videos. As we can see in Table 3, SUDDS outperforms the rest on unseen modern upscaling methods, notably being the only one able to detect blind SR methods consistently. We select the model trained on a multiclass setting to perform the final test.

3.2.1.2 Ablation study We study SUDDS performance when removing the staircase and DCT components in the settings of binary and multiclass classification. In the former, SUDDS is trained to discriminate just between original and fake images, without classifying the SR method. In the latter, we train our model to classify according to the SR method (*i.e.* each SR method has its own label). In evaluation, the model just needs to detect an instance as original or fake. As in paragraph 3.2.1.1, we train with the full BVI-DVC-SR and evaluate on our videos.

The results are in Table 4. In the binary classification setting, we can see that the full SUDDS architecture performs the best for original images and in the detection of two out of four unseen SR methods, having a notable drop in performance in the other two. On the other hand, in the multiclass classification setting, it is clear that the full SUDDS architecture is the best at detecting original and all unseen SR methods as fake. And as opposed to the best binary setting detector, the full SUDDS can detect all unseen SR methods when trained in the multiclass setting.

3.2.1.3 Generalisation capability To measure how well SUDDS generalises to other unseen SR methods, we train it with a single SR method on the BVI-DVC-SR train split, and evaluate it on the other methods with the validation split.

Results in Table 5 indicate there is a clear similarity between the artifacts produced by BasicVSR and RVRT, as a model trained on one is capable of detecting the other with high accuracy. This may suggest that the two methods share similar characteristics and generate the same patterns in the outscaled images, which may be caused by both using a recurrent video architecture. Generally, the model trained with one specific method can accurately identify original videos, but it shows a propensity to misclassify fake videos from outside the training distribution.

	Validation Method				
Training Method	Original	Bicubic	BasicVSR	Real-BasicVSR	RVRT
Bicubic	0.963	0.999	0.156	0.001	0.172
BasicVSR	0.981	0.751	0.982	0.015	0.977
RealBasicVSR	0.999	0.001	0.001	0.990	0.001
RVRT	0.940	0.754	0.981	0.014	0.980

Table 5. Accuracy scores of each training-validation pair. The network is trained on four individual methods and evaluated with all of them. The more intense the blue color of a cell, the more acceptable is the performance of the training method in detecting the validation method.

3.2.2 Conclusions

- We present the Synthetic Upscaling Detector with DCT features and Staircase module (SUDDS), a new super-resolution detection architecture, which adds a DCT-feature based module and a staircase structure module that joins low-level and high-level information.
- By processing patches instead of entire images, SUDDS alleviates the computational load of the process, especially at inference time, and is able to handle images in 4K resolution.
- We show that SUDDS is a competitive detector in comparison to state-of-the-art SR detection methods, having a superior capability to detect previously unseen SR methods.

3.2.3 Relevant publications

- Cuena E., "Super-resolution assessment and detection", Master thesis, July 2023.

3.2.4 Relevance to AI4Media use cases and media industry applications

Our proposed architecture detects a type of synthetically modified images and videos, those upscaled via super-resolution. These are of interest to the media industry for legal concerns related to the AI Act, and ethical concerns regarding super-resolution methods generating information that was not there in the original data. For example, a model may produce a fake vehicle registration number in an upscaled image that was too blurry to distinguish in the original, effectively generating false information. Regarding AI4Media, this work tackles the task of user story 1A4-10 (Detection of synthetic upscaling of a video).

3.3 Super-resolution neural network selection optimisation

Contributing partners: RAI

RAI's effort in Task 5.5 is focused on the design and development of a workflow to optimise the selection of the most suitable SR neural network depending on the main characteristics of the content. To achieve this goal, a multi-dimensional content type classification scheme, driven from actual scenarios in broadcasting and multimedia content management, will be proposed. This classification scheme will be used in conjunction with objective and subjective evaluation experiments on various combinations of SR networks and content types, to select the most suitable super-resolution model according to the content type. Implementing a content classification scheme will provide valuable insights and help manage the huge variety of video content.

By following this approach, the envisaged advantage would be that of researching, defining and optimising the development of a series of specific SR network architectures adapted to the main content characteristics, as opposed to the objective of finding a one-fits-all architecture. This is based on the empirical observation that many of the existing solutions fall short in specific critical aspects (e.g., spatial details, motion complexity) rather than in general and that existing content genres are typically made up of a non-uniform combination of these aspects. Thus, we foresee that in some subcases there might be less sophisticated solutions and that more complex architectures can be limited to fewer cases, and as such that the overall computational and energy-related footprint needed for training would be substantively optimised.

3.3.1 Multi-dimensional Content Type Classification Scheme

As a broadcaster, RAI has a huge expertise in video content [53] and is well-equipped to propose a classification scheme for media content type that appropriately considers the specific requirements and objectives of Task T5.5. Television genres encompass different storytelling styles, characterisation, dialogue, emotional and visual elements [17]. While some television content adhere strictly to specific genre e.g., news, sports, commercials, some others span across several genre and sub-genre. Although there is no exhaustive classification scheme of all television genres, some genres emerge more frequently and have achieved a major popularity over the years.

Determining the best classification scheme for video content to evaluate which super-resolution neural network fits better for what kind of content requires careful consideration of various factors. These include the explicit objectives of the evaluation, the need to consider representative real-world scenarios, such as those included in the broadcaster's content, the selection of appropriate performance metrics, the consideration of diverse resolution levels to account for varying levels of detail (e.g., low, medium, and high resolution), and the availability of a suitable dataset.

While there is no one-size-fits-all classification scheme, here are some commonly used categories that could be considered to support the correct selection of a super-resolution neural network for a specific kind of video content:

- "News" is information about current events. News programming includes local news and national daytime newscasts;
- "Talk show" category includes television programmes based on discussions between hosts and guests. Most talk shows deal with pop culture, current affairs and politics;
- "Sport" programming has proven to be one of the most popular television formats. Unlike many other genres, sport is often broadcast live, providing a feeling of immediacy to the viewing experience;
- "Commercial" refers to short advertising messages or promotional videos created for the purpose of promoting products, services or brands to television audiences. Commercials are typically short in duration, ranging from a few seconds to a couple of minutes;
- "Cartoon" is also known as animated television series. It refers to a genre of television programming that features fictional or animated characters brought to life through the art of animation. These shows are primarily created for entertainment and are typically targeted towards children, although many cartoons have gained popularity among viewers of all ages;
- "Archive" is the genre referring to television programmes that deal with the exploration, curation, and presentation of historical or pre-existing audiovisual material usually from the

[17] https://en.wikipedia.org/wiki/List_of_television_formats_and_genres

broadcaster's archive. Examples include videos with motion blur, noise, compression artifacts, or other forms of degradation commonly encountered in real-world scenarios;

- "Variety show" is about programmes that highlight the talents of their guests. Variety shows include musical acts, dancing, stand-up comedy and sketch comedy. Their purpose is to entertain the audience;

Designing a super resolution framework necessitates not only content classification based on genre but also an enhanced description of the video content's features. For instance, "motion complexity" is the category used to label motion sequences that encapsulate temporal changes and dynamics present in videos. It can range from simple motion, such as camera panning or object translation, to complex motion involving fast-moving objects, occlusions, or scene changes. By evaluating super-resolution networks applied to motion sequences, researchers can assess the model's capability to handle complex motion patterns and generate high-resolution outputs that maintain temporal consistency.

Thus, a possible classification scheme could be based on the following dimensions:

- Genre
- Motion complexity
- Resolution

One of the purposes of the proposed classification scheme is to evaluate the efficacy of super-resolution networks on videos belonging to different genres and with varying levels of resolutions and motion complexity. The latter is particularly useful to assess their capacity to process various motion patterns and preserve temporal coherence. Additionally, evaluating super-resolution networks on a diversity of scene types offers insights into their performance in a variety of visual contexts and helps identify any potential bias or restriction in the network's generalization capabilities. Moreover, assessing super-resolution networks across a range of resolution levels gives a wider understanding of their ability to generalize and improve visual quality with different input conditions.

By incorporating these classes of sequences in the evaluation process, researchers would be able to establish comprehensive benchmark datasets that accurately reflect real-world scenarios. This facilitates fair comparisons between different super-resolution networks and ensures that their performance is evaluated across a diverse range of patterns, enabling more robust conclusions regarding their effectiveness.

3.3.2 Objective and subjective assessment

To perform the correct selection of a super-resolution neural network for a specific kind of video content it is necessary to set up a proper assessment workflow.

The importance of multimedia quality for modern communication systems is evident in many aspects. With the proliferation of devices capable of playing audio and video content, consumer awareness of perceived quality has grown significantly. Furthermore, the consumption of multimedia content is expected to raise due to the increased accessibility of the Internet together with several video and audio[18] streaming services. Consequently, both consumers and service providers are eager to make effective use of these technologies, making the perceived quality of multimedia content an issue of great interest to all stakeholders.

[18] https://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-highlights/pdf/Global_2021_Forecast_Highlights.pdf

It is widely accepted that human end-users are the most accurate judges of video quality, and their opinions can be collected performing Video Quality Assessment (VQA) through subjective experiments. These typically involve a panel of participants, referred to as *test subjects*, to evaluate the perceived quality of a set of images or videos. However, it is not always possible to employ this methodology because subjective experiments are time-consuming and expensive. Contrariwise, the objective VQA methods, which are usually computational models of quality estimation that can be automated, are considered time-efficient and more suitable for real-time applications.

RAI's plan for the next period is to assess the different outputs of super-resolution neural networks under evaluation through a mix of both objective and subjective experiments, which will also allow to better correlate the proposed objective metrics with the subjective tests.

3.3.3 Objective evaluations

As for the objective experiments, the following objective metrics to measure the similarity between two images or video frames will be considered. These metrics are widely used in image and video processing applications, such as image compression, quality assessment and image restoration. They are full-reference metrics, i.e., they require access to the original undistorted image for comparison.

PSNR The Peak Signal-to-Noise Ratio (PSNR) is a metric [19] used to quantify the fidelity of a signal representation, calculated as the ratio of the maximum possible power of a signal to the power of corrupting noise. It is often employed to assess the quality of digital signal transmission. In the case of digital images, each pixel can be considered as a component of a signal with 8-bit or 10-bit RGB values.

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right) \tag{1}$$

Where, MAX is the maximum valid value for a pixel and MSE is the Mean Squared Error between the high-resolution image and the super-resolved image. It is a pixel-by-pixel comparison over the entire image. The PSNR index ranges between 0 and ∞. Empirical evaluations [54] demonstrated that an acceptable quality value should exceed 40: processed images have a perceived quality similar to the original.

SSIM SSIM stands for Structural Similarity Index Measure [55]. SSIM takes into account both structural information and image pixel values, with the goal of capturing perceived visual quality.

The SSIM algorithm compares the similarities in luminance, contrast and structure between two images. To this end, it divides the images into small windows and calculates the similarity measures for each window. The final SSIM score is then calculated as the average of the similarity measures across all windows.

The SSIM index varies between 0 and 1, where 1 indicates perfect similarity between the images, while 0 means no similarity at all. The closer the SSIM score is to 1, the more similar the images are perceived to be.

VMAF VMAF stands for Video Multimethod Assessment Fusion[20]. It is a widely used objective video quality metric developed by Netflix. VMAF is based on a machine learning model that was trained using a large dataset of subjective human ratings. The model considers various characteristics of video frames, including spatial and temporal factors as well as perceptual properties such as

[19] https://it.wikipedia.org/wiki/Peak_signal-to-noise_ratio
[20] https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652

Image System	Reference	Aspect Ratio	Optimal Horizontal Viewing Angle	Optimal Viewing Distance
720 x 483	ITU-R BT.601	4:3	11°	7H
640 x 480	VGA	4:3	11°	7H
720 x 576	ITU-R BT.601	4:3	13°	6H
1024 x 768	XGA	4:3	17°	4.5H
1280 x 720	ITU-R BT.1543	16:9	21°	4.8H
1920 x 1080	ITU-R BT.709	16:9	31°	3.2H
3840 x 2160	ITU-R BT.2020	16:9	58°	1.6H
7680 x 4320	ITU-R BT.2020	16:9	96°	0.8H

Table 6. Optimal horizontal viewing angle, optimal viewing distance in picture heights (H)

contrast and texture. The VMAF score is generally reported on a scale from 0 to 100, where higher scores indicate better perceived quality.

3.3.4 Subjective evaluations

Subjective image assessment techniques are employed to determine the effectiveness of television systems by taking measurements that more accurately reflect the responses of potential viewers of the systems being evaluated.

The most valid measure of video quality is obtained through the collection of the ratings by a human panel, usually following the ITU BT.500 [56] recommendation, referred to as *subjective experiments*. The traditional method for subjective experiments, which is still the preferred approach, plans to carry them out in a controlled laboratory environment. This solution involves several crucial factors such as a careful planning, a clear evaluation method and the selection of suitable test materials. The opinion scores of each individual subject are used to calculate a representative value known as Mean Opinion Score (MOS) and serves as the ground truth of quality for a particular test stimulus.

The designed workflow for subjective evaluations of video quality will be based on the expert viewing protocol with the participation of a small number of viewers, all selected from experts in the field of video processing.

3.3.4.1 Lab set-up for subjective evaluations The used display must be a panel with performance typical of professional applications e.g., broadcast studios. The diagonal size of the display may vary, with a minimum requirement of 22 inches, which can be extended up to 50 inches or even more when evaluating High-Definition Television (HDTV) or higher resolution imaging systems. The display should allow proper luminance and colour setting and calibration, using a professional light measuring instrument.

The users viewing distance from the TV set should be chosen according to the screen resolution and the height of the active part of the screen, following the design viewing distance guidelines described in Table 6.

Any direct or reflected source of light falling on the screen should be eliminated. Additionally, the level of ambient light should be maintained at a low level, such as approximately 5 cd/m^2

The number of simultaneous evaluators in front of the monitor may vary according to the size of the screen to ensure the same image rendition and stimulus presentation for all viewers. Participants engaged in an Experimental Visual Perception (EVP) study are required to possess expertise in the

relevant field of investigation. Consequently, there is no need to conduct screening for visual acuity or colour blindness, as the selection should be limited to qualified individuals. A minimum of nine distinct evaluators is considered necessary to provide a sufficient number of diverse perspectives for the study.

3.3.4.2 The basic test cell The material presented to the experts should be organized by creating a basic test cell (BTC) for each pair of encoding conditions to be evaluated (see Figure 5). The source reference sequences (SRCs) and processed video sequences (PVSs) to be considered

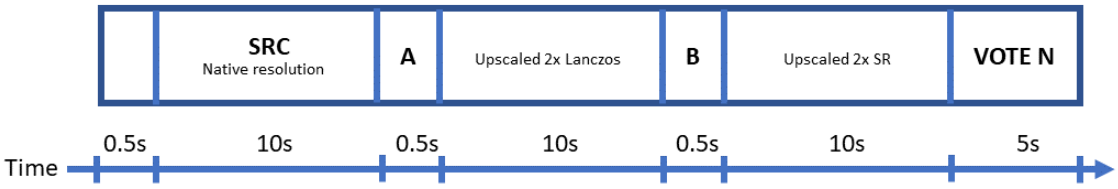


Figure 5. Timings of a basic test cell for the expert viewing protocol

in a BTC should always be relative to the same video sequence, so that the experts can identify any improvement in visual quality provided by the super-resolution algorithms under evaluation. For example, sequence A should be upscaled with a state-of-the-art non-learnable algorithm e.g., Lanczos, while sequence B with a super-resolution algorithm based on DL.

The BTC should be organised as follows:

- 0.5 seconds with the screen set to a mid-grey (mean value in the luminance scale)
- 10 seconds presentation of the reference uncompressed video clip
- 0.5 seconds showing the message “A” (first video to assess) on a mid-grey background
- 10 seconds presentation of an impaired version of the video clip
- 0.5 seconds showing the message “B” (second video to assess) on a mid-grey background
- 10 seconds presentation of an impaired version of the video clip
- 5 seconds showing a message that asks the viewers to express their opinion.

The ‘Vote’ label should be followed by a number that helps to get synchronised on the scoring sheet.

3.3.4.3 Scoring sheet As depicted in Figure 5, the display of the video clips should be arranged in a manner such that the unimpaired reference (SRC) is presented first, followed by two impaired video sequences (PVS). The order in which the PVSs are presented must be altered randomly for each BTC and viewers must not be aware of the order of presentation.

An 11-degree numerical scale is employed, ranging from 10 (imperceptible impairments) to 0 (very annoying impairments).

Figure 6 provides guidance regarding the interpretation of the 11-degree numerical scale.

Viewers are requested to complete a questionnaire which consists of two boxes (labelled ’A’ and ’B’) for each BTC, and to enter a score from the 11-grade numerical scale into each of the two boxes. An illustration of a scorecard for a session consisting of 24 BTCs is provided in Figure 7.

Score	Impairment item	
10	Imperceptible	
9	Slightly perceptible	somewhere
8		everywhere
7	Perceptible	somewhere
6		everywhere
5	Clearly perceptible	somewhere
4		everywhere
3	Annoying	somewhere
2		everywhere
1	Severely annoying	somewhere
0		everywhere

Figure 6. Meaning of the 11 grades numerical scale

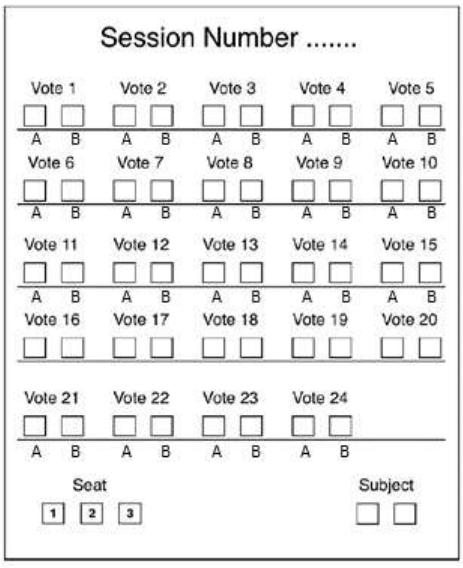


Figure 7. Example of scoring sheet for a 24-BTC expert viewing session

For each BTC, viewers must fill in both the box designated by the letter A (to rate the video clip presented first) and the box designated by the letter B (to rate the video clip presented second).

The presentation of the original, non-compromised video clip makes it easier for experts to assess possible impairments.

The meaning of the 11-degree numerical scale must be carefully explained during the 'training sessions'.

3.3.4.4 Test design The order of presentation of each BTC should be random, to ensure that the same video clip is not displayed consecutively, as well as the same impaired clip.

At the beginning of each viewing session a "stabilization phase" should be conducted. This foresees to show to the audience the "best", the "worst" and two "mid quality" BTC among those included in the test session. This will allow the viewers to have an immediate impression of the quality range.

If the viewing session is longer than 20 minutes, the test designer should split it into two (or more) separate viewing sessions, each of them not exceeding 20 minutes. In such a case, the "stabilization phase" should be provided before each viewing session.

It is recommended to organise a short training viewing session prior to each experiment, even if this procedure involves experts users. The video material used in the training session may be the same that will be used during the actual sessions; however, the order of presentation should be varied. The viewers should be instructed on the use of the 11-grade scale by asking them to carefully observe the video clips displayed after messages "A" and "B" on the screen, and check whether they can detect any difference to the video clip presented initially (the SRC).

At the end of each session, scores should be gathered and logged onto an electronic spreadsheet to calculate the mean values. It is desirable to conduct a post-screening of the viewers, utilizing a linear Pearson's correlation. The 'correlation' function should be applied to all scores of each subject in comparison to the MOS; a threshold may be set to determine whether a viewer is 'acceptable' or 'rejected' (Recommendation ITU-T P.913 [57] proposes the use of a 'reject' threshold value of 0.75).

3.3.5 Conclusions

- RAI is examining the connections between content properties (such as genre, motion complexity, and resolution) and deep models, with the goal of formulating simpler solutions for a given video content.
- RAI is working on identifying the most appropriate objective metric to enhance the process of selecting the optimal deep model for each video.

3.3.6 Relevance to AI4Media use cases and media industry applications

This research result will be useful for UC3: AI in Vision - High quality video production and content automation.

Editorial teams and content creators are aware of consumer demand for high-quality content characterised by higher resolutions, greater colour fidelity and richer details. T5.5 work in relation to UC3 will enable efficient and high-quality improvements, including processes such as super-resolution and image denoising. The research described in subsection 3.3 will support UC3 identifying the most effective multidimensional classification scheme and objective metrics to be used within a super-resolution framework. This will optimise the selection of the most suitable super-resolution model based on the main characteristics of the content. Improvements will be verified through a quality assessment methodology that can be adapted to any form of content.

In addition, from a content creation and production perspective, UC3 will also benefit from T5.5 results to improve the video quality of content. As an example, T5.5 tools could be applied to the broadcaster's archives or other content sources to deliver enhanced videos about a place or a monument, in order to address the lack of on-site journalists and content when and where the event starts and takes place.

4 Efficient training for large neural networks

Deep learning models are notorious for their appetite for data. The more data you can give them, the better they perform. Unfortunately, in most real-life situations, this is not possible. In this section we present two solutions addressing this problem: 1) using whitening for self-supervised representation learning (SSL); and 2) efficient training of visual transformers with small datasets.

Whitening for self-supervised representation learning. Most of the current SSL methods are based on the *contrastive loss* and the instance-discrimination task, where augmented versions of the same image instance ("positives") are contrasted with instances extracted from other images ("negatives"). For the learning to be effective, many negatives should be compared with a positive pair, which is computationally demanding. In subsection 4.1, we propose a different direction and a new loss function for SSL, which is based on the *whitening* of the latent-space features. The whitening operation has a "scattering" effect on the batch samples, avoiding degenerate solutions where all the sample representations collapse to a single point. Our solution does not require asymmetric networks and it is conceptually simple. Moreover, since negatives are not needed, we can extract multiple positive pairs from the same image instance.

Efficient training of visual transformers with small datasets. Visual Transformers (VTs) have emerged as an architectural paradigm alternative to Convolutional networks (CNNs). Differently from CNNs, VTs can capture global relations between image elements and they potentially have a larger representation capacity. However, the lack of the typical convolutional inductive bias makes these models more data hungry than common CNNs. In subsection 4.2, we empirically analyse different VTs, comparing their robustness in a small training set regime, and we show that, despite having a comparable accuracy when trained on a large dataset such as ImageNet, their performance on smaller datasets can be largely different. Moreover, we propose an auxiliary self-supervised task which can extract additional information from images with only a negligible computational overhead. This task encourages the VTs to learn spatial relations within an image and makes the VT training much more robust when training data is scarce. Our task is used jointly with the standard (supervised) training and it does not depend on specific architectural choices, thus it can be easily plugged in the existing VTs. Using an extensive evaluation with different VTs and datasets, we show that our method can improve (sometimes dramatically) the final accuracy of the VTs.

4.1 Whitening for Self-Supervised Representation Learning

Contributing partners: UNITN

In self-supervision, label-based information is replaced by a prediction problem using some form of *context* or using a *pretext* task. Pioneering work in this direction was done in Natural Language Processing (NLP), in which the co-occurrence of words in a sentence is used to learn a language model [58, 59, 60]. In Computer Vision, typical contexts or pretext tasks are based on: (1) the temporal consistency in videos [61, 62, 63], (2) the spatial order of patches in still images [64, 65, 66] or (3) simple image transformation techniques [67, 68, 69]. The intuitive idea behind most of these methods is to collect pairs of *positive* and *negative* samples: two positive samples should share the same semantics, while negatives should be perceptually different. A triplet loss [70, 71, 72, 61, 62] can then be used to learn a metric space representing the human perceptual similarity. However, most of the recent studies use a contrastive loss [73] or one of its variants [74, 75, 76], while [77] shows the relation between the triplet and the contrastive losses.

It is worth noticing that the success of both kinds of losses is strongly affected by the number and the quality of the negative samples. For instance, in the case of the triplet loss, a common practice is to select *hard/semi-hard* negatives [71, 72]. On the other hand, [76] has shown that the

contrastive loss needs a large number of negatives to be competitive. This implies using batches with a large size, which is computationally demanding, especially with high-resolution images. In order to alleviate this problem, [69] uses a *memory bank* of negatives, which is composed of feature-vector representations of all the training samples. [68] conjectures that the use of large and fixed-representation vocabularies is one of the keys to the success of self-supervision in NLP. The solution proposed in MoCo [68] extends [69] using a memory-efficient queue of the last visited negatives, together with a *momentum encoder* which preserves the intra-queue representation consistency. [78] has performed large-scale experiments confirming that a large number of negatives (and therefore a large batch size) is required for the contrastive loss to be efficient.

Very recently, [79] proposed an alternative direction, in which only positives are used, together with two networks, where the *online* network tries to predict the representation of a positive extracted by the *target* network. Despite the large success of BYOL [79], the reason why the two networks can avoid a collapsed representation (e.g., where all the images are mapped to the same point) is still unclear [80, 81, 2, 82]. According to [80, 81], one of the important ingredients which is *implicitly* used in BYOL to avoid degenerate solutions, is the use of the Batch Norm (BN) [83] in the projection/prediction heads. In this research we propose to generalize this finding and we show that, using a full *whitening* of the latent space features is *sufficient* to avoid collapsed representations, without the need of additional momentum networks [79], siamese networks with stop-gradient operations [2] or the use of specific, batch-based optimizers like LARS [84, 80].

In more detail, we propose a new SSL loss function, which first *scatters* all the sample representations in a spherical distribution[21] and then *penalizes* the positive pairs which are far from each other (see Figure 8). Specifically, given a set of samples $V = \{\mathbf{v}_i\}$, corresponding to the current mini-batch of images $B = \{x_i\}$, we first project the elements of V onto a spherical distribution using a *whitening* transform [85]. The whitened representations $\{\mathbf{z}_i\}$, corresponding to V, are normalized and then used to compute a Mean Squared Error (MSE) loss which accumulates the error considering only positive pairs $(\mathbf{z}_i, \mathbf{z}_j)$. To avoid a representation collapse, we do not need to *contrast* positives against negatives as in the contrastive loss or in the triplet loss because the optimization process leads to shrinking the distance between positive pairs and, indirectly, scatters the other samples to satisfy the overall spherical-distribution constraint.

4.1.1 Experiments

In our experiments we use the following **datasets**:

- CIFAR-10 and CIFAR-100 [86], two small-scale datasets composed of 32×32 images with 10 and 100 classes, respectively.
- ImageNet [3], the well-known large-scale dataset with about 1.3M training images and 50K test images, spanning over 1000 classes.
- Tiny ImageNet [87], a reduced version of ImageNet, composed of 200 classes with images scaled down to 64×64. The total number of images is: 100K (training) and 10K (testing).
- ImageNet-100 [88], a random 100-class subset of ImageNet.
- STL-10 [89], also derived from ImageNet, with 96×96 resolution images and more than 100K training samples.

[21] Here and in the following, with "spherical distribution" we mean a distribution with a zero-mean and an identity-matrix covariance.

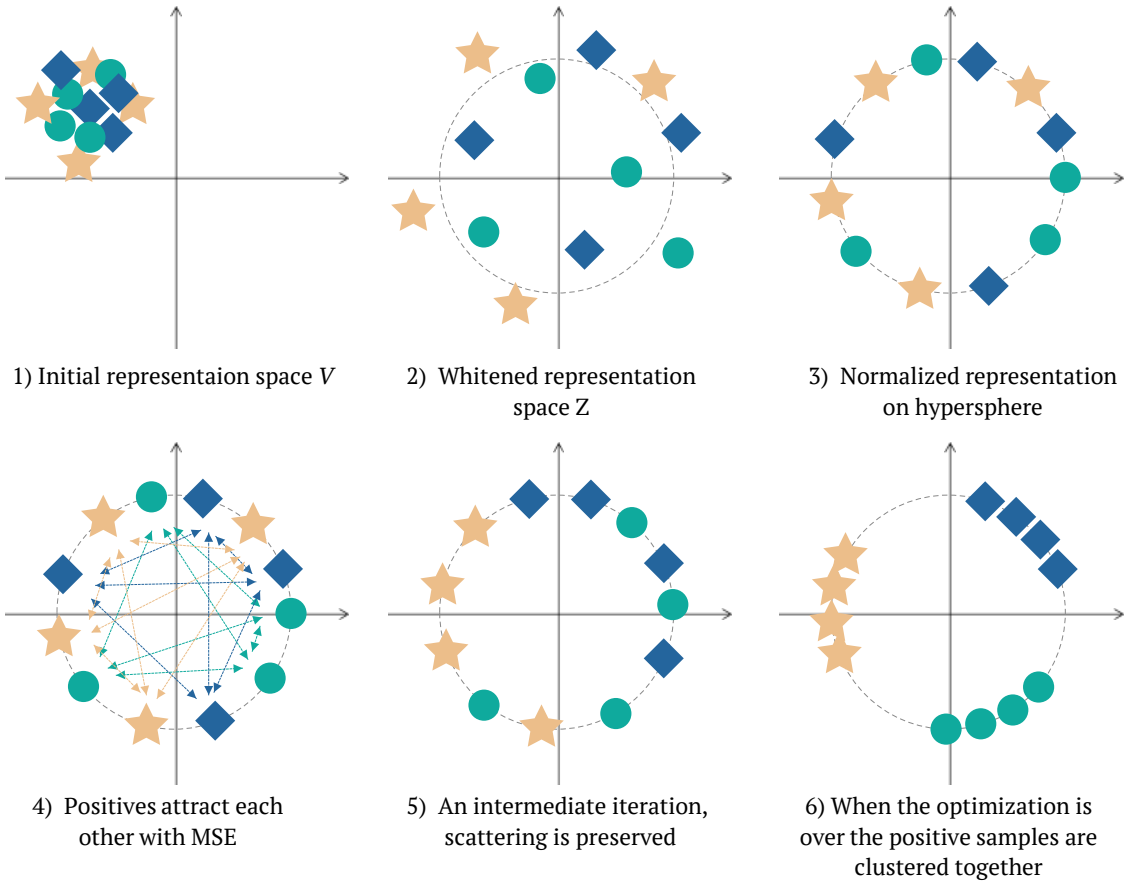


Figure 8. A schematic representation of the W-MSE based optimization process. Positive pairs are indicated with the same shapes and colors. (1) A representation of the batch features in V when training starts. (2, 3) The distribution of the elements after whitening and the L_2 normalization. (4) The MSE computed over the normalized $\mathbf{z}$ *features encourages the network to move the positive pair representations closer to each other. (5) The subsequent iterations move closer and closer the positive pairs, while the relative layout of the other samples is forced to lie in a spherical distribution.*

Settings. The goal of our experiments is to compare W-MSE with state-of-the-art SSL losses, isolating the effects of other settings, such as the architectural choices. For this reason, in the small and medium size dataset experiments of Table 7, we use the same encoder $E(\cdot)$, ResNet-18, for all the compared methods and, similarly, we use ResNet-50 for the ImageNet-based experiments in Table 9. When we do not report previously published results, we independently select the best hyperparameter values for each method and each dataset. In each method, the latent-space features are L_2 normalized, unless otherwise specified. In Table 7, *SimCLR (our repro.)* refers to our implementation of the contrastive loss following the details in [78], with temperature $\tau = 0.5$. In the same table, *BYOL (our repro.)* is our reproduction of [79]. For this method we use the exponential moving average with cosine increasing, starting from 0.99. *W-MSE 2* and *W-MSE 4* correspond to our method with $d = 2$ and $d = 4$ positives extracted per image, respectively. For CIFAR-10 and CIFAR-100, the slicing sub-batch size is 128, for Tiny ImageNet and STL-10, it is 256. In the Tiny ImageNet and STL-10 experiments with W-MSE 2, we use 4 iterations of batch slicing, while in all the other experiments we use only 1 iteration.

Table 7. Classification accuracy (top 1) of a linear classifier and a 5-nearest neighbors classifier for different loss functions and datasets with a ResNet-18 encoder.

Method	CIFAR-10		CIFAR-100		STL-10		Tiny ImageNet	
	linear	5-nn	linear	5-nn	linear	5-nn	linear	5-nn
SimCLR [78] (our repro.)	91.80	88.42	66.83	56.56	90.51	85.68	48.84	32.86
BYOL [79] (our repro.)	91.73	89.45	66.60	**56.82**	**91.99**	**88.64**	**51.00**	**36.24**
W-MSE 2 (ours)	91.55	89.69	66.10	56.69	90.36	87.10	48.20	34.16
W-MSE 4 (ours)	**91.99**	**89.87**	**67.64**	56.45	91.75	88.59	49.22	35.44

Table 8. Classification accuracy on ImageNet-100. Top 1 and 5 correspond to the accuracy of a linear classifier. W-MSE (2 and 4) are based on a ResNet-18 *encoder. † indicates that the results are based on a* ResNet-50 *encoder and the values are reported from [1].*

Method	top 1	top 5	5-nn
MoCo [68] †	72.80	91.64	-
$\mathcal{L}_{\text{align}}$ and $\mathcal{L}_{\text{uniform}}$			
Wang & Isola [1] †	74.60	92.74	-
W-MSE 2 (ours)	76.00	93.14	67.04
W-MSE 4 (ours)	**79.02**	**94.46**	**71.32**

4.1.1.1 Comparison with the state of the art

Table 7 shows the results of the experiments on small and medium size datasets. For W-MSE, 4 samples are generally better than 2. The contrastive loss performs the worst in most cases. The W-MSE 4 accuracy is the best on CIFAR-10 and CIFAR-100, while BYOL leads on STL-10 and Tiny ImageNet, although the gap between the two methods is marginal.

Table 8 shows the results on a larger dataset (ImageNet-100). In the table, MoCo is the contrastive-loss based method proposed in [68], and $\mathcal{L}_{\text{align}}$ and $\mathcal{L}_{\text{uniform}}$ are the two losses proposed in [1]. Note that, while W-MSE (2 and 4) in Table 8 refer to our method with a *ResNet-18* encoder, the other results are reported from [1], where a much larger-capacity network (i.e., a *ResNet-50*) is used as the encoder. *Despite this large difference in the encoder capacity*, both versions of W-MSE significantly outperform the other two compared methods in this dataset. Table 8 also shows that W-MSE 2, the version of our method without multi-cropping, is highly competitive, being its classification accuracy significantly higher than state-of-the-art methods.

Finally, in Table 9 we show the ImageNet results using 100 and 400 training epochs, and we compare W-MSE 4 with the results of other state-of-the-art approaches as reported in [2]. Despite some configuration details are different (e.g., the depth of the projection head, etc.), in all cases the encoder is a ResNet-50. However, SwAV refers to the reproduction of [91] used in [2], where no multi-crop strategy is adopted (hence, $d = 2$), and a multi-crop version of SwAV may likely obtain significantly larger values. Table 9 shows that W-MSE 4 is the state of the art with 100 epochs and it is very close to the 400-epochs state of the art. These results confirm that our method is highly competitive, considering that we have not intensively tuned our hyperparameters and that our network is much simpler than other approaches.

Table 9. Classification accuracy (top 1) of a linear classifier on ImageNet with a ResNet-50 encoder. All results but ours are reported from [2]. † The reproduction of SwAV in [2] does not include a multi-crop strategy.

Method	100 epochs	400 epochs
SimCLR [78]	66.5	69.8
MoCo v2 [90]	67.4	71.0
BYOL [79]	66.5	**73.2**
SwAV [91] †	66.5	70.7
SimSiam [2]	68.1	70.8
W-MSE 4 (ours)	**69.43**	72.56

Table 10. CIFAR-10: accuracy of the contrastive loss with whitened features, trained for 200 epochs.

Whitened features	L_2 normalized	linear	5-nn
✗	✓	89.66	86.55
✓	✓	diverged	
✗	✗	79.48	76.60
✓	✗	77.39	74.14

4.1.1.2 Contrastive loss with whitening In this section, we analyse the effect of the whitening transform in combination with the contrastive loss. Specifically, we use the contrastive loss on whitened features $\mathbf{z} = Whitening(\mathbf{v})$. Table 10 shows the results on CIFAR-10. The first row refers to the standard contrastive loss without whitening. Note that the difference with respect to Table 7 is due to the use of only 200 training epochs. If the features are whitened and then normalized, we observed an unstable training, with divergence after a few epochs. The unnormalized version with whitening converged, but its accuracy is worse than the standard contrastive loss (both normalized and unnormalized).

These experiments show that the whitening transform alone does not improve the SSL performance, and, used jointly with negative contrasting, it may be harmful. Conversely, we use whitening in our W-MSE to avoid a collapsed representation when only positives are used.

4.1.2 Conclusions

Overall, our contributions are the following:

- We propose a new SSL loss function, Whitening MSE (W-MSE). W-MSE constrains the batch samples to lie in a spherical distribution and it is an alternative to positive-negative instance contrasting methods.
- Our loss function does not need a large number of negatives, thus we can include more positives in the current batch. We indeed demonstrate that multiple positive pairs extracted from one image improve the performance.

- We empirically show that our W-MSE loss outperforms the commonly adopted contrastive loss and it is competitive with respect to state-of-the-art SSL methods like [79, 2].

4.1.3 Relevant publications

- A. Ermolov, A. Siarohin, E. Sangineto, and N. Sebe, "Whitening for Self-Supervised Representation Learning", International Conference on Machine Learning, July 2021 [92].
  Zenodo record: https://zenodo.org/record/5543415.

4.1.4 Relevant software and/or external resources

- The Pytorch implementation can be found in https://github.com/htdt/self-supervised.

4.1.5 Relevance to AI4Media use cases and media industry applications

Our whitening tool is generic and can be applied to many applications of computer vision and multimedia in which image and video content is being analyzed. Concretely, our approach could be useful to epics (a) 3A3 (archive exploration), specifically user story 3A3-11 (Visual indexing and search), (b) 4C1 (multimedia analysis in and across multiple modalities) by enhancing the solutions to analyze visual content, and (c) 7A3 ((Re)organisation of visual content) by supporting the efficient training and organization of image and video collections.

4.2 Efficient Training of Visual Transformers with Small-Size Datasets

Contributing partners: UNITN

Visual Transformers (VTs) are progressively emerging architectures in computer vision as an alternative to standard Convolutional Neural Networks (CNNs), and they have already been applied to many tasks, such as image classification [93, 94, 95, 96, 97, 98, 99, 100], object detection [101, 102, 103], segmentation [104], tracking [105], image generation [106, 107] and 3D data processing [108], to mention a few. These architectures are inspired by the well known Transformer [109], which is the de facto standard in Natural Language Processing (NLP) [60, 110], and one of their appealing properties is the possibility to develop a unified information-processing paradigm for both visual and textual domains. A pioneering work in this direction is ViT [93], in which an image is split using a grid of non-overlapping patches, and each patch is linearly projected in the input embedding space, so obtaining a "token". After that, all the tokens are processed by a series of multi-head attention and feed-forward layers, similarly to how (word) tokens are processed in NLP Transformers.

A clear advantage of VTs is the possibility for the network to use the attention layers to model global relations between tokens, and this is the main difference with respect to CNNs, where the receptive field of the convolutional kernels locally limits the type of relations which can be learned. However, this increased representation capacity comes at a price, which is the lack of the typical CNN inductive biases, based on exploiting the locality, the translation invariance and the hierarchical structure of visual information [96, 97, 98]. As a result, VTs need a lot of data for training, usually more than what is necessary to standard CNNs [93]. For instance, ViT is trained with JFT-300M [93], a (proprietary) huge dataset of 303 million (weakly) labeled high-resolution images, and performs worse than ResNets [4] with similar capacity when trained on ImageNet-1K ($\sim$ 1.3 million samples [111]). This is likely due to the fact that ViT needs to learn some local proprieties of the visual data using more samples than a CNN, while the latter embeds these properties in its architectural design [112].

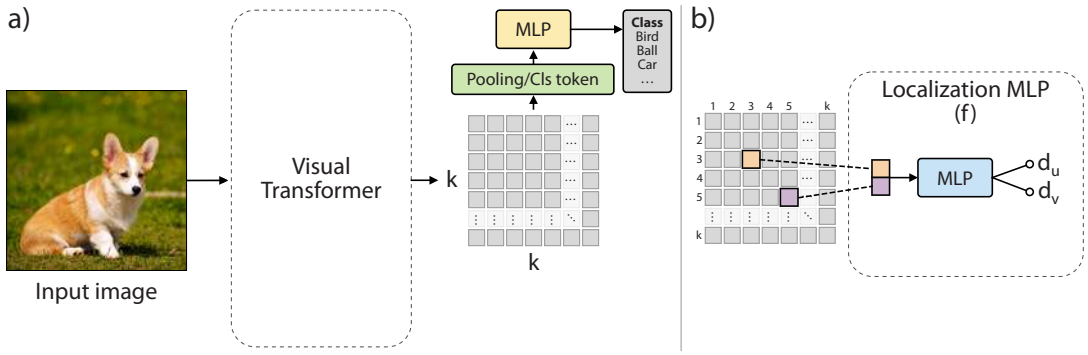


Figure 9. A schematic representation of the VT architecture. (a) A typical second-generation VT. (b) Our localization MLP which takes as input (concatenated) pairs of final token embeddings.

To alleviate this problem, a second generation of VTs has been independently proposed by different groups [95, 96, 97, 98, 100, 99, 107]. A common idea behind these works is to mix convolutional layers with attention layers, in such a way providing a local inductive bias to the VT. These hybrid architectures enjoy the advantages of both paradigms: attention layers model long-range dependencies, while convolutional operations can emphasize the local properties of the image content. The empirical results shown in most of these works demonstrate that these second-generation VTs can be trained on ImageNet outperforming similar-size ResNets on this dataset [95, 96, 97, 98, 100, 99]. However, it is still not clear what is the behaviour of these networks when trained on medium-small datasets. In fact, from an application point of view, most of the computer vision tasks cannot rely on (supervised) datasets whose size is comparable with (or larger than) ImageNet.

In this research, we compare to each other different second-generation VTs by either training them from scratch or fine-tuning them on medium-small datasets, and we empirically show that, despite their ImageNet results being basically on par with each other, their classification accuracy with smaller datasets largely varies. We also compare VTs with same capacity ResNets, and we show that, in most cases, VTs can match the ResNet accuracy when trained with small datasets. Moreover, we propose to use an auxiliary self-supervised *pretext* task and a corresponding loss function to regularize training in a small training set or few epochs regime. Specifically, the proposed task is based on (unsupervised) learning the spatial relations between the output token embeddings. Given an image, we *densely* sample random pairs from the final embedding grid, and, for each pair, we ask the network to guess the corresponding geometric distance. To solve this task, the network needs to encode both local and contextual information in each embedding. In fact, without local information, embeddings representing different input image patches cannot be distinguished from each other, while, without contextual information (aggregated using the attention layers), the task may be ambiguous.

Our task is inspired by ELECTRA [113], in which the (NLP) pretext task is densely defined for each output embedding. Clark et al. [113] show that their task is more *sample-efficient* than commonly used NLP pretext tasks, and this gain is particularly strong with small-capacity models or relatively smaller training sets. Similarly, we exploit the fact that an image is represented by a VT using multiple token embeddings, and we use their relative distances to define a localization task over a subset of all the possible embedding pairs. This way, *for a single image forward pass*, we can compare many embedding pairs with each other, and average our localization loss over all of them. Thus, our task is drastically different from those multi-crop strategies proposed, for instance, in SwAV [91], which need to independently forward each input patch through the network.

Table 11. The size of the datasets used in our empirical analysis.

Dataset		Train size	Test size	Classes
ImageNet-1K [111]		1,281,167	100,000	1000
ImageNet-100 [115]		126,689	5,000	100
CIFAR-10 [86]		50,000	10,000	10
CIFAR-100 [86]		50,000	10,000	100
Oxford Flowers102 [116]		2,040	6,149	102
SVHN [117]		73,257	26,032	10
DomainNet	ClipArt	33,525	14,604	345
	Infograph	36,023	15,582	
	Painting	50,416	21,850	
	Quickdraw	120,750	51,750	
	Real	120,906	52,041	
	Sketch	48,212	20,916	

Moreover, differently from "ordering" based tasks [114], we can define pairwise distances on a large grid without modeling all the possible permutations. See Figure 9 for details.

Since our auxiliary task is self-supervised, our *dense relative localization* loss ($\mathcal{L}_{drloc}$) does not require additional annotation, and we use it jointly with the standard (supervised) cross-entropy as a regularization of the VT training. $\mathcal{L}_{drloc}$ is very easy-to-be-reproduced and, despite this simplicity, it can largely boost the accuracy of the VTs, especially when the VT is either trained from scratch on a small dataset, or fine-tuned on a dataset with a large domain-shift with respect to the pretraining ImageNet dataset. In our empirical analysis, based on different training scenarios, a variable amount of training data and different VT architectures, $\mathcal{L}_{drloc}$ has *always* improved the results of the tested baselines, sometimes boosting the final accuracy of tens of points (and up to 45 points).

4.2.1 Experiments

All the experiments are based on image classification tasks on 11 different datasets: ImageNet-100 (IN-100) [115, 1], which is a subset of 100 classes of ImageNet-1K [111]; CIFAR-10 and CIFAR-100 [86], Oxford Flowers102 [116] and SVHN [117], which are four widely used computer vision datasets; and the six datasets of DomainNet [118], a benchmark commonly used for domain adaptation tasks. We chose the latter because of the large domain-shift between some of its datasets and ImageNet-1K, which makes the fine-tuning experiments non-trivial. Table 11 shows the size of each dataset.

We used, when available, the official VT code (for T2T [95] and Swin [96]) and a publicly available implementation of CvT [97][22]. In the fine-tuning experiments, we use only T2T and Swin because of the lack of publicly available ImageNet pre-trained CvT networks. For each of the three baselines, we chose a model of comparable size to ResNet-50 (25M parameters): see Table 13 for more details. When we plug our loss on one of the adopted baselines we kept unchanged the VT architecture apart from our localization MLP (f). Moreover, in all the experiments, we train the baselines, both with and without our localization loss, using the same data-augmentation protocol for all the models, and we use the VT-specific hyper-parameter configuration suggested by the authors of each VT. We do *not* tune the VT-specific hyperparameters when we use our loss and we keep fixed the values of m and λ (paragraph 4.2.1.1) in all the experiments. We train each model using 8 V100 32GB GPUs.

[22] https://github.com/lucidrains/vit-pytorch

4.2.1.1 Ablation study

In Table 12 (a) we analyze the impact on the accuracy of different values of m (the total number of embedding pairs used per image. Since we use the same grid resolution for all the VTs (i.e., 7×7), also the maximum number of possible embeddings per image is the same for all the VTs ($k^2 = 49$). Using the results of Table 12 (a) (based on CIFAR-100 and Swin), we chose $m = 64$ for all the VTs and all the datasets. Moreover, Table 12 (b) shows the influence of the loss weight λ for each of the three baselines, which motivates our choice of using $\lambda = 0.1$ for both CvT and T2T and $\lambda = 0.5$ for Swin.

These values of m and λ *are kept fixed* in all the other experiments of this paper, independently of the dataset, the main task (e.g., classification, detection, segmentation, etc.), and the training protocol (from scratch or fine-tuning). This is done to emphasise the ease of use of our loss. Finally, in the Supplementary Material, we analyze the influence of the size of the localization MLP (f).

Table 12. CIFAR-100, 100 training epochs: (a) the influence on the accuracy of the number of pair samples (m) in L_{drloc} using Swin, and (b) the influence of the λ value using all the 3 VT baselines.

(a)

Model	Top-1 Acc.
A: Swin-T	53.28
B: A + $\mathcal{L}_{drloc}$, m=32	63.70
C: A + $\mathcal{L}_{drloc}$, m=64	**66.23**
D: A + $\mathcal{L}_{drloc}$, m=128	65.16
E: A + $\mathcal{L}_{drloc}$, m=256	64.87

(b)

Model	λ=0.0	λ=0.1	λ=0.5	λ=1.0
CvT-13	73.50	**74.51**	74.07	72.84
Swin-T	53.28	58.15	**66.23**	64.28
T2T-ViT-14	65.16	**68.03**	67.03	66.53

Table 13. Top-1 accuracy on IN-100 using either 100 or 300 epochs. In the former case, we show the average and the standard deviation values obtained by repeating each single experiment 5 times with 5 different random seeds.

	Model	# Params (M)	ImageNet-100	
			100 epochs	300 epochs
CvT	CvT-13	20	85.62 ± 0.05	90.16
	CvT-13+$\mathcal{L}_{drloc}$	20	**86.09** ± 0.12 (+0.47)	**90.28** (+0.12)
Swin	Swin-T	29	82.66 ± 0.10	89.68
	Swin-T+$\mathcal{L}_{drloc}$	29	**83.95** ± 0.05 (+1.29)	**90.32** (+0.64)
T2T	T2T-ViT-14	22	82.67 ± 0.01	87.76
	T2T-ViT-14+$\mathcal{L}_{drloc}$	22	**83.74** ± 0.08 (+1.07)	**88.16** (+0.40)

4.2.1.2 Training from scratch

In this section, we analyze the performance of both the VT baselines and our regularization loss using small-medium datasets and different number of training epochs, simulating a scenario with limited computational resources and/or limited training data. In fact, while fine-tuning a model pre-trained on ImageNet-1K is the most common protocol when dealing with small training datasets, this is not possible when, e.g., the network input is not an RGB image (e.g., in case of 3D point cloud data [108]) or when using a task-specific backbone architecture [119, 120]. In these cases, the network needs to be trained from scratch on the target dataset, thus, investigating the robustness of the VTs when trained from scratch with relatively small datasets, is useful for those application domains in which a fine-tuning protocol cannot be adopted.

We start by analyzing the impact on the accuracy of the number of training epochs on IN-100. Table 13 shows that, using $\mathcal{L}_{drloc}$, *all the tested VTs* show an accuracy improvement, and this boost

is larger with fewer epochs. As expected, our loss acts as a regularizer, whose effects are more pronounced in a shorter training regime. We believe this result is particularly significant considering the larger computational times which are necessary to train typical VTs with respect to ResNets.

In Table 14, we use all the other datasets and we train from scratch with 100 epochs. First, we note that the accuracy of the VT baselines varies a lot depending on the dataset (which is expected), but also depending on the specific VT architecture. This is largely in contrast with the ImageNet-1K results, where the difference between the three baselines is much smaller. As a reference, when these VTs are trained on ImageNet-1K (for 300 epochs), the differences of their respective top-1 accuracy is much smaller: Swin-T, 81.3 [96]; T2T-ViT-14, 81.5 [95]; CvT-13, 81.6 [97]. Conversely, Table 14 shows that, for instance, the accuracy difference between CvT and Swin is about 45-46 points in Quickdraw and Sketch, 30 points on CIFAR-10, and about 20 points on many other datasets. Analogously, the difference between CvT and T2T is between 20 and 25 points in Sketch, Painting and Flowers102, and quite significant in the other datasets. This comparison shows that CvT is usually much more robust in a small training set regime with respect to the other two VTs, a behaviour which is completely hidden when the training/evaluation protocol is based on large datasets only.

In the same table, we also show the accuracy of these three VTs when training is done using $\mathcal{L}_{drloc}$ as a regularizer. Similarly to the IN-100 results, also in this case our loss *improves the accuracy of all the tested VTs in all the datasets*. Most of the time, this improvement is quite significant (e.g., almost 4 points on SVHN with CvT), and sometimes dramatic (e.g., more than 45 points on Quickdraw with Swin). These results show that a self-supervised auxiliary task can provide a significant "signal" to the VT when the training set is limited, and, specifically, that our loss can be very effective in boosting the accuracy of a VT trained from scratch in this scenario.

In Table 14 we also report the results we obtained using a ResNet-50, trained with 100 epochs and the standard ResNet training protocol (e.g., using Mixup [121] and CutMix [122] data-augmentations, etc.). These results show that the best performing VT (CvT) is usually comparable with a same size ResNet, and demonstrate that VTs can potentially be trained from scratch with darasets smaller than InageNet-1K. Finally, in the last row of the same table, we train the ResNet-50 baseline jointly with our pretext task. In more detail, we replace the VT token embedding grid with the last convolutional feature map of the ResNet, and we apply our loss on top of this map. A comparison between the results of the last 2 rows of Table 14 shows that our loss is useful also when used with a ResNet. When using ResNets, the improvement obtained with our loss is marginal, but it is consistent in 9 out of 10 datasets. The smaller improvement with respect to the analogous VT results may probably be explained by the fact that ResNets already embed local inductive biases in their architecture, thus a localization auxiliary task is less helpful.

4.2.1.3 Fine-tuning In this section, we analyze a typical fine-tuning scenario, in which a model is pre-trained on a big dataset (e.g., ImageNet), and then fine-tuned on the target domain. Specifically, in *all* the experiments, we use VT models pre-trained by the corresponding VT authors on ImageNet-1K *without* our localization loss. The difference between the baselines and ours concerns *only* the fine-tuning stage, which is done in the standard way for the former and using our $\mathcal{L}_{drloc}$ regularizer for the latter. Starting from standard pre-trained models and using our loss only in the fine-tuning stage, emphasises the easy to use of our proposal in practical scenarios, in which fine-tuning can be done without re-training the model on ImageNet.

The results are presented in Table 15. Differently from the results shown in paragraph 4.2.1.2, the accuracy difference between the T2T and Swin baselines is much less pronounced, and the latter outperforms the former in most of the datasets. Moreover, analogously to all the other experiments, using $\mathcal{L}_{drloc}$ leads to an accuracy improvement *with all the tested VTs and in all the datasets*. For

Table 14. Top-1 accuracy of VTs and ResNets, trained from scratch on different datasets (100 epochs).

		CIFAR-10	CIFAR-100	Flowers102	SVHN	ClipArt	Infograph	Painting	Quickdraw	Real	Sketch
CvT	CvT-13	89.02	73.50	54.29	91.47	60.34	19.39	54.79	70.10	76.33	56.98
	CvT-13+$\mathcal{L}_{drloc}$	**90.30**	**74.51**	**56.29**	**95.36**	**60.64**	**20.05**	**55.26**	**70.36**	**77.05**	**57.56**
		(+1.28)	(+1.01)	(+2.00)	(+3.89)	(+0.30)	(+0.67)	(+0.47)	(+0.26)	(+0.68)	(+0.58)
Swin	Swin-T	59.47	53.28	34.51	71.60	38.05	8.20	35.92	24.08	73.47	11.97
	Swin-T+$\mathcal{L}_{drloc}$	**83.89**	**66.23**	**39.37**	**94.23**	**47.47**	**10.16**	**41.86**	**69.41**	**75.59**	**38.55**
		(+24.42)	(+12.95)	(+4.86)	(+22.63)	(+9.42)	(+1.96)	(+5.94)	(+45.33)	(+2.12)	(+26.58)
T2T	T2T-ViT-14	84.19	65.16	31.73	95.36	43.55	6.89	34.24	69.83	73.93	31.51
	T2T-ViT-14+$\mathcal{L}_{drloc}$	**87.56**	**68.03**	**34.35**	**96.49**	**52.36**	**9.51**	**42.78**	**70.16**	**74.63**	**51.95**
		(+3.37)	(+2.87)	(+2.62)	(+1.13)	(+8.81)	(+2.62)	(+8.54)	(+0.33)	(+0.70)	(+20.44)
ResNet	ResNet-50	91.78	72.80	46.92	96.45	63.73	19.81	53.22	71.38	75.28	**60.08**
	ResNet-50+$\mathcal{L}_{drloc}$	**92.03**	**72.94**	**47.65**	**96.53**	**63.93**	**20.79**	**53.52**	**71.57**	**75.56**	59.62
		(+0.25)	(+0.14)	(+0.73)	(+0.08)	(+0.20)	(+0.98)	(+0.30)	(+0.19)	(+0.28)	(-0.46)

Table 15. Pre-training on ImageNet-1K and then fine-tuning on the target dataset (top-1 accuracy, 100 fine-tuning epochs).

		CIFAR-10	CIFAR-100	Flowers102	SVHN	ClipArt	Infograph	Painting	Quickdraw	Real	Sketch
Swin	Swin-T	97.95	88.22	98.03	96.10	73.51	41.07	72.99	75.81	85.48	72.37
	Swin-T+$\mathcal{L}_{drloc}$	**98.37**	**88.40**	**98.21**	**97.87**	**79.51**	**46.10**	**73.28**	**76.01**	**85.61**	**72.86**
		(+0.42)	(+0.18)	(+0.18)	(+1.77)	(+6.00)	(+5.03)	(+0.29)	(+0.20)	(+0.13)	(+0.49)
T2T	T2T-ViT-14	98.37	87.33	97.98	97.03	74.59	38.53	72.29	74.16	84.56	72.18
	T2T-ViT-14+$\mathcal{L}_{drloc}$	**98.52**	**87.65**	**98.08**	**98.20**	**78.22**	**45.69**	**72.42**	**74.27**	**84.57**	**72.29**
		(+0.15)	(+0.32)	(+0.10)	(+1.17)	(+3.63)	(+7.16)	(+0.13)	(+0.11)	(+0.01)	(+0.11)
ResNet	ResNet-50	97.65	85.44	96.59	96.60	75.22	44.30	66.58	72.12	80.40	67.77
	ResNet-50+$\mathcal{L}_{drloc}$	**97.74**	**85.65**	**96.72**	**96.71**	**75.51**	**44.39**	**69.03**	**72.21**	**80.54**	**68.14**
		(+0.09)	(+0.21)	(+0.13)	(+0.11)	(+0.29)	(+0.09)	(+2.45)	(+0.09)	(+0.14)	(+0.37)

instance, on Infograph, Swin with $\mathcal{L}_{drloc}$ improves of more than 5 points, and T2T more than 7 points. In the last two rows of Table 15, we show the ResNet based results. The comparison between ResNet and the VT baselines shows that the latter are very competitive in this fine-tuning scenario, even more than with a training-from-scratch protocol (Table 14). For instance, the two VT baselines (without our loss) are outperformed by ResNet only in 2 out of 10 datasets. This confirms that VTs are likely to be widely adopted in computer vision applications in the near future, independently of the training set size. Finally, analogously to the experiments in paragraph 4.2.1.2, Table 15 shows that our loss is (marginally) helpful also in ResNet fine-tuning.

4.2.2 Conclusions

In summary, our main contributions are:

- We empirically compare to each other different VTs, showing that their behaviour largely differs when trained with small datasets or few training epochs.
- We propose a relative localization auxiliary task for VT training regularization.
- Using an extensive empirical analysis, we show that this task is beneficial to speed-up training and improve the generalization ability of different VTs, independently of their specific architectural design or application task.

4.2.3 Relevant publications

- Y. Liu, E. Sangineto, W. Bi, N. Sebe, B. Lepri, and M. De Nadai, "Efficient Training of Visual Transformers with Small-Size Datasets", Neural Information Processing Systems, December 2021 [123].
  Zenodo record: https://zenodo.org/record/6363240.

4.2.4 Relevant software and/or external resources

- The Pytorch implementation can be found in https://github.com/yhlleo/VTs-Drloc.

4.2.5 Relevance to AI4Media use cases and media industry applications

Our approach for efficient training of visual transformers can be applied in all use cases where visual transformers could be applied, explicitly when the available annotated datasets are small. This can be the case of user stories in 3A3 (archive exploration), specifically 3A3-11 (Visual indexing and search), and 7A3 ((Re)organisation of visual content) by supporting the efficient training of image and video collections.

5 Efficient mathematical computations

Over the past decade, deep learning models have exhibited considerable advancements, reaching or even exceeding human-level performance in a range of visual perception tasks. This remarkable progress has sparked interest in applying deep networks to real-world applications, such as autonomous vehicles, mobile devices, robotics, and edge computing. However, the challenge remains that state-of-the-art models usually demand significant computational resources, leading to impractical power consumption, latency, or carbon emissions in real-world scenarios. This trade-off between effectiveness and efficiency has catalyzed the emergence of a new research focus: computationally efficient deep learning, which strives to achieve satisfactory performance while minimizing the computational cost during inference.

In this section, we provide solutions for the efficient computation of two of the most common and frequent basic operations used in deep learning: Matrix square root and EigenDecomposition (ED).

Matrix square root. Computing the matrix square root or its inverse in a differentiable manner is important in a variety of computer vision tasks. Previous methods either adopt the Singular Value Decomposition (SVD) to explicitly factorize the matrix or use the Newton-Schulz iteration (NS iteration) to derive the approximate solution. However, both methods are not computationally efficient enough in either the forward pass or in the backward pass. In subsection 5.1, we propose two more efficient variants to compute the differentiable matrix square root. For the forward propagation, one method is to use Matrix Taylor Polynomial (MTP), and the other method is to use Matrix Padé Approximants (MPA). The backward gradient is computed by iteratively solving the continuous-time Lyapunov equation using the matrix sign function. Both methods yield considerable speed-up compared with the SVD or the Newton-Schulz iteration. Experimental results on the de-correlated batch normalization and second-order vision transformer demonstrate that our methods can also achieve competitive and even slightly better performances.

EigenDecomposition. This operation is at the heart of many algorithms and applications. One crucial bottleneck limiting its usage is the expensive computation cost, particularly for a mini-batch of matrices in the deep neural networks. In subsection 5.2, we propose a QR[23]-based ED method performing the ED entirely by batched matrix/vector multiplication, which processes all the matrices simultaneously and thus fully utilizes the power of GPUs. Our technique is based on the explicit QR iterations by Givens rotation with double Wilkinson shifts. With several acceleration techniques, the time complexity of QR iterations is reduced from $O(n^5)$ to $O(n^3)$. The numerical test shows that for small and medium batched matrices (*e.g.,* dim<32) our method can be much faster than the Pytorch SVD function. Experimental results on visual recognition and image generation demonstrate that our methods also achieve competitive performances.

5.1 Fast Differentiable Matrix Square Root

Contributing partners: UNITN

Consider a positive semi-definite matrix $\boldsymbol{A}$. The principle square root $\boldsymbol{A}^{\frac{1}{2}}$ and the inverse square root $\boldsymbol{A}^{-\frac{1}{2}}$ (often derived by calculating the inverse of $\boldsymbol{A}^{\frac{1}{2}}$) are mathematically of practical interests, mainly because some desired spectral properties can be obtained by such transformations. An exemplary illustration is given in Figure 10 (a). As can be seen, the matrix square root can shrink/stretch the feature variances along with the direction of principle components, which is known as an effective spectral normalization for covariance matrices. The inverse square root, on the other hand, can be used to whiten the data, *i.e.,* make the data has a unit variance in each

[23]The name "QR" is derived from the letter Q, used to denote orthogonal matrices, and the letter R, used to denote right triangular matrices.

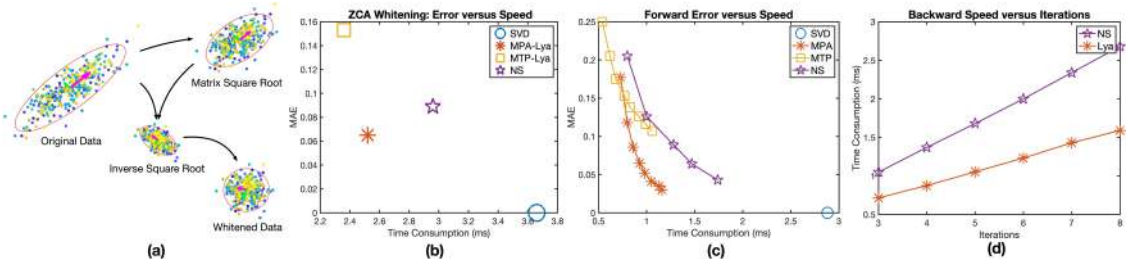


Figure 10. ***(a)*** *Exemplary visualization of the matrix square root and its inverse. Given the original data* $\boldsymbol{X}\in\mathbb{R}^{2\times n}$*, the matrix square root stretches the data along the axis of small variances and squeezes the data in the direction with large variances, serving as a spectral normalization for covariance matrices. The inverse square root, on the other hand, can be used to transform the data into the uncorrelated structure,* i.e., *have the unit variance in each dimension.* ***(b)*** *The comparison of error and speed using the setting of our ZCA whitening experiment. Our MTP and MPA are faster than the SVD and the NS iteration, and our MPA is more accurate than the NS iteration.* ***(c)*** *The comparison of speed and error in the forward pass (FP). The iteration times of the NS iteration range from* 3 *to* 7*, while the degrees of our MTP and MPA vary from* 6 *to* 18*. Our MPA computes the more accurate and faster matrix square root than the NS iteration, and our MTP enjoys the fastest calculation speed.* ***(d)*** *The speed comparison in the backward pass (BP). Our Lyapunov solver is more efficient than the NS iteration as fewer matrix multiplications are involved.*

dimension. Due to the appealing spectral properties, computing the matrix square root or its inverse in a differentiable manner arises in a wide range of computer vision applications, including covariance pooling [124, 125, 126], decorrelated batch normalization [127, 128, 129], and Whitening and Coloring Transform (WCT) [130, 131, 132].

To compute the matrix square root, the standard method is via Singular Value Decomposition (SVD). Given the real Hermitian matrix $\boldsymbol{A}$, its matrix square root is computed as:

$$\boldsymbol{A}^{\frac{1}{2}} = (\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T)^{\frac{1}{2}} = \boldsymbol{U}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{U}^T \tag{2}$$

where $\boldsymbol{U}$ is the eigenvector matrix, and $\boldsymbol{\Lambda}$ is the diagonal eigenvalue matrix. As derived by [133], the partial derivative of the eigendecomposition is calculated as:

$$\frac{\partial l}{\partial \boldsymbol{A}} = \boldsymbol{U}\Big(\boldsymbol{K}^T \odot (\boldsymbol{U}^T\frac{\partial l}{\partial \boldsymbol{U}}) + (\frac{\partial l}{\partial \boldsymbol{\Lambda}})_{\text{diag}}\Big)\boldsymbol{U}^T \tag{3}$$

where l is the loss function, $\odot$ denotes the element-wise product, and $()_{\text{diag}}$ represents the operation of setting the off-diagonal entries to zero. Despite the long-studied theories and well-developed algorithms of SVD, there exist two obstacles when integrated into deep learning frameworks. One issue is the back-propagation instability. For the matrix $\boldsymbol{K}$ defined in Eq. (3), its off-diagonal entry is $K_{ij}={}^{1}/_{(\lambda_i-\lambda_j)}$, where λ_i and λ_j are involved eigenvalues. When the two eigenvalues are close and small, the gradient is very likely to explode, *i.e.,* $K_{ij}\to\infty$. This issue has been solved by some methods that use approximation techniques to estimate the gradients [134, 135, 126]. The other problem is the expensive time cost of the forward eigendecomposition. As the SVD is not supported well by GPUs, performing the eigendecomposition on the deep learning platforms is rather time-consuming. Incorporating the SVD with deep models could add extra burdens to the training process. Particularly for batched matrices, modern deep learning frameworks, such as Tensorflow and Pytorch, give limited optimization for the matrix decomposition within the mini-batch. A (parallel) for-loop is inevitable for conducting the SVD one matrix by another. However, how to efficiently perform the SVD in the context of deep learning has not been touched.

To avoid explicit eigendecomposition, one commonly used alternative is the Newton-Schulz iteration (NS iteration) [136, 137]. It modifies the ordinary Newton iteration by replacing the

matrix inverse but preserves the quadratic convergence. Compared with SVD, the NS iteration is rich in matrix multiplication and more GPU-friendly. Thus, this technique has been widely used to approximate the matrix square root in different applications [124, 125, 128]. The forward computation relies on the following coupled iterations:

$$\boldsymbol{Y}_{k+1} = \frac{1}{2}\boldsymbol{Y}_k(3\boldsymbol{I} - \boldsymbol{Z}_k\boldsymbol{Y}_k), \boldsymbol{Z}_{k+1} = \frac{1}{2}(3\boldsymbol{I} - \boldsymbol{Z}_k\boldsymbol{Y}_k)\boldsymbol{Z}_k \tag{4}$$

where $\boldsymbol{Y}_k$ and $\boldsymbol{Z}_k$ converge to the matrix square root $\boldsymbol{A}^{\frac{1}{2}}$ and the inverse square root $\boldsymbol{A}^{-\frac{1}{2}}$, respectively. Since the NS iteration only converges locally, we need to pre-normalize the initial matrix and post-compensate the resultant approximation as:

$$\boldsymbol{Y}_0 = \frac{1}{||\boldsymbol{A}||_{\mathrm{F}}}\boldsymbol{A}, \ \boldsymbol{A}^{\frac{1}{2}} = \sqrt{||\boldsymbol{A}||_{\mathrm{F}}}\boldsymbol{Y}_k. \tag{5}$$

Each forward iteration involves 3 matrix multiplications, which is more efficient than the forward pass of SVD. The backward pass of the NS iteration is calculated as:

$$\begin{aligned}\frac{\partial l}{\partial \boldsymbol{Y}_k} &= \frac{1}{2}\Big(\frac{\partial l}{\partial \boldsymbol{Y}_{k+1}}(3\boldsymbol{I} - \boldsymbol{Y}_k\boldsymbol{Z}_k) - \boldsymbol{Z}_k\frac{\partial l}{\partial \boldsymbol{Z}_{k+1}}\boldsymbol{Z}_k - \boldsymbol{Z}_k\boldsymbol{Y}_k\frac{\partial l}{\partial \boldsymbol{Y}_{k+1}}\Big), \\ \frac{\partial l}{\partial \boldsymbol{Z}_k} &= \frac{1}{2}\Big((3\boldsymbol{I} - \boldsymbol{Y}_k\boldsymbol{Z}_k)\frac{\partial l}{\partial \boldsymbol{Z}_k} - \boldsymbol{Y}_k\frac{\partial l}{\partial \boldsymbol{Y}_{k+1}}\boldsymbol{Y}_k - \frac{\partial l}{\partial \boldsymbol{Z}_{k+1}}\boldsymbol{Z}_k\boldsymbol{Y}_k\Big)\end{aligned} \tag{6}$$

where the backward pass needs 10 matrix multiplications for each iteration. The backward gradients for the post-compensation and pre-normalization steps are computed as:

$$\begin{aligned}\frac{\partial l}{\partial \boldsymbol{A}}\Big|_{post} &= \frac{1}{2||\boldsymbol{A}||_{\mathrm{F}}^{3/2}}\mathrm{tr}\Big((\frac{\partial l}{\partial \boldsymbol{Y}_k})^T\boldsymbol{Y}_k\Big)\boldsymbol{A}, \\ \frac{\partial l}{\partial \boldsymbol{A}}\Big|_{pre} &= -\frac{1}{||\boldsymbol{A}||_{\mathrm{F}}^3}\mathrm{tr}\Big((\frac{\partial l}{\partial \boldsymbol{Y}_0})^T\boldsymbol{A}\Big)\boldsymbol{A} + \frac{1}{||\boldsymbol{A}||_{\mathrm{F}}}\frac{\partial l}{\partial \boldsymbol{Y}_0} + \frac{\partial l}{\partial \boldsymbol{A}}\Big|_{post}.\end{aligned} \tag{7}$$

These two steps take 4 matrix multiplications in total. Consider the fact that the NS iteration often takes 5 iterations to achieve reasonable performances [125, 128]. The backward pass is much more time-costing than the backward algorithm of SVD. As seen from Figure 10 (b) and (c), although the NS iteration outperforms SVD by 123% in the speed of forward pass, its overall time consumption only improves that of SVD by 17%. The speed improvement could be larger if a more efficient backward algorithm is developed.

To address the drawbacks of SVD and NS iteration, *i.e.* the low efficiency in either the forward or backward pass, we derive two methods **that are efficient in both forward and backward propagation** to compute the differentiable matrix square root. In the forward pass, we propose to use Matrix Taylor Polynomial (MTP) and Matrix Padé Approximants (MPA) to approximate the matrix square root. The former approach is slightly faster but the latter is more numerically accurate (see Figure 10 (c)). Both methods yield considerable speed-up compared with the SVD or the NS iteration in the forward computation. For the backward pass, we consider the gradient function as a Lyapunov equation and propose an iterative solution using the matrix sign function. The backward pass costs fewer matrix multiplications and is more computationally efficient than the NS iteration (see Figure 10 (d)). Through a series of numerical tests, we show that the proposed MTP-Lya and MPA-Lya deliver consistent speed improvement for different batch sizes, matrix dimensions, and some hyper-parameters (*e.g.,* degrees of power series to match and iteration times). Moreover, our proposed MPA-Lya consistently gives a better approximation of the matrix square root than the NS iteration. Besides the numerical tests, experiments on the decorrelated batch

normalization and second-order vision transformer also demonstrate that our methods can achieve competitive and even better performances against the SVD and the NS iteration with the least amount of time overhead.

5.1.1 Experiments

In this section, we first perform a series of numerical tests to compare our proposed methods with the SVD and NS iteration. Subsequently, we validate the effectiveness of our MTP and MPA in two deep learning applications: ZCA (Zero-phase Component Analysis) whitening and covariance pooling.

5.1.1.1 Numerical Tests To comprehensively evaluate the numerical performance, we compare the speed and error for the input of different batch sizes, matrices in various dimensions, different iteration times of the backward pass, and different polynomial degrees of the forward pass. In each of the following tests, the comparison is based on $10,000$ random covariance matrices and the matrix size is consistently 64×64 unless explicitly specified. The error is measured by calculating the Mean Absolute Error (MAE) of the matrix square root computed by the approximate methods (NS iteration, MTP, and MPA) and the accurate methods (SVD).

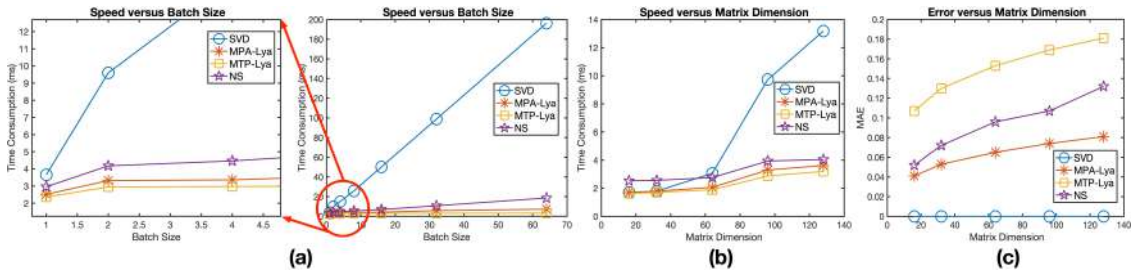


*Figure 11. The results of the numerical tests. **(a)** The computation speed (FP+BP) of each method versus different batch sizes. **(b)** The speed comparison (FP+BP) of each method versus different matrix dimensions. **(c)** The error comparison of each method versus different matrix dimensions. The hyper-parameters follow our experimental setting of ZCA whitening and covariance pooling.*

Forward Error versus Speed Both the NS iteration and our methods have a hyper-parameter to tune in the forward pass, *i.e.,* iteration times for NS iteration and polynomial degrees for our MPA and MTP. To validate the impact, we measure the speed and error for different hyper-parameters. The degrees of our MPA and MTP vary from 6 to 18, and the iteration times of NS iteration range from 3 to 7. We give the preliminary analysis in Figure 10 (c). As can be observed, our MTP has the least computational time, and our MPA consumes slightly more time than MTP but provides a closer approximation. Moreover, the curve of our MPA consistently lies below that of the NS iteration, demonstrating our MPA is a better choice in terms of both speed and accuracy.

Backward Speed versus Iteration Figure 10 (d) compares the speed of our backward Lyapunov solver and the NS iteration versus different iteration times. Our Lyapunov solver is much more efficient than NS iteration. For the NS iteration of 5 times, our Lyapunov solver still has an advantage even when we iterate 8 times.

Speed versus Batch Size In certain applications such as covariance pooling and instance whitening, the input could be batched matrices instead of a single matrix. To compare the speed performance for batched input, we conduct another numerical test. The hyper-parameter choices follow our experimental settings in ZCA whitening. As seen in Figure 11 (a), our MPA-Lya and MTP-Lya are consistently more efficient than the NS iteration and SVD. To give a concrete example,

when the batch size is 64, our MPA-Lya is 2.58X faster than NS iteration and 27.25X faster than SVD, while our MTP-Lya is 5.82X faster than the NS iteration and 61.32X faster than SVD.

As discussed before, the current SVD implementation adopts a for-loop to compute each matrix one by one within the mini-batch. This accounts for why the time consumption of SVD grows almost linearly with the batch size. For the NS iteration, the backward pass is not as batch-friendly as our Lyapunov solver. The gradient calculation defined in Eq. (7) requires measuring the trace and handling the multiplication for each matrix in the batch, which has to be accomplished ineluctably by a for-loop[24]. Our backward pass can be more efficiently implemented by batched matrix multiplication.

Speed and Error versus Matrix Dimension In this numerical test, we compare the speed and error for matrices in different dimensions. The hyper-parameter settings also follow our experiments of ZCA whitening. As seen from Figure 11 (b), our proposed MPA-Lya and MTP-Lya consistently outperform others in terms of speed. In particular, when the matrix size is very small (<32), the NS iteration does not hold a speed advantage over the SVD. By contrast, our proposed methods still have competitive speed against the SVD. Figure 11 (c) presents the approximation error. Our MPA-Lya always has a better approximation than the NS iteration, whereas our MTP-Lya gives a worse estimation but takes the least time consumption, which can be considered as a trade-off between speed and accuracy.

5.1.1.2 ZCA Whitening: Decorrelated Batch Normalization

Following [135], we insert the decorrelated batch normalization layer after the first convolutional layer of ResNet [4]. For our forward pass, we match the MTP to the power series of degree 11 and set the degree for both numerator and denominator of our MPA as 5. We keep iterating 8 times for our backward Lyapunov solver. The detailed architecture changes and the implementation details of other methods are kindly referred to the Appendix.

Table 16 displays the speed and validation error on CIFAR10 and CIFAR100 [138]. Our MTP-Lya is 1.25X faster than NS iteration and 1.48X faster than SVD-Padé, and our MPA-Lya is 1.17X and 1.34X faster. Furthermore, our MPA-Lya achieves state-of-the-art performances across datasets and models. Our MTP-Lya has comparable performances on ResNet-18 but slightly falls behind on ResNet-50. We guess this is mainly because the relatively large approximation error of MTP might affect little on the small model but can hurt the large model.

Table 16. Validation error of different ZCA whitening methods. The covariance matrix is of size 1×64×64. The time consumption is measured for computing the matrix square root (BP+FP) on a workstation equipped with a Tesla K40 GPU and a 6-core Intel(R) Xeon(R) CPU @ 2.20GHz. For each method, we report the results based on five runs.

Methods	Time (ms)	ResNet-18				ResNet-50	
		CIFAR10		CIFAR100		CIFAR100	
		mean±std	min	mean±std	min	mean±std	min
SVD-PI	3.49	4.59±0.09	4.44	21.39±0.23	21.04	19.94±0.44	19.28
SVD-Taylor	3.41	4.50±0.08	4.40	21.14±0.20	**20.91**	19.81±0.24	19.26
SVD-Padé	3.39	4.65±0.11	4.50	21.41±0.15	21.26	20.25±0.23	19.98
NS Iteration	2.96	4.57±0.15	4.37	21.24±0.20	21.01	**19.39±0.30**	**19.01**
Our MPA-Lya	2.52	**4.39±0.09**	**4.25**	**21.11±0.12**	20.95	**19.55±0.20**	19.24
Our MTP-Lya	**2.36**	4.49±0.13	4.31	21.42±0.21	21.24	20.55±0.37	20.12

[24]See the code in the official Pytorch implementation of [125] via this link.

Table 17. Validation top-1/top-5 accuracy of the second-order vision transformer on ImageNet [3]. The covariance matrices are of size 64×48×48, where 64 is the mini-batch size. The time cost is measured for computing the matrix square root (BP+FP) on a workstation equipped with a Tesla 4C GPU and a 6-core Intel(R) Xeon(R) CPU @ 2.20GHz. For the So-ViT-14 model, all the methods achieve similar performances but spend different epochs.

Methods	Time (ms)	Architecture		
		So-ViT-7	So-ViT-10	So-ViT-14
PI	**1.84**	75.93 / 93.04	77.96 / 94.18	82.16 / 96.02 (303 epoch)
SVD-PI	83.43	76.55 / 93.42	78.53 / 94.40	82.16 / 96.01 (278 epoch)
SVD-Taylor	83.29	76.66 / **93.52**	78.64 / 94.49	82.15 / 96.02 (271 epoch)
SVD-Padé	83.25	76.71 / 93.49	78.77 / 94.51	82.17 / 96.02 (265 epoch)
NS Iteration	10.38	76.50 / 93.44	78.50 / 94.44	82.16 / 96.01 (280 epoch)
Our MPA-Lya	3.25	**76.84** / 93.46	**78.83** / **94.58**	82.17 / 96.03 (**254** epoch)
Our MTP-Lya	2.39	76.46 / 93.26	78.44 / 94.33	82.16 / 96.02 (279 epoch)

5.1.1.3 Covariance Pooling: Second-order Vision Transformer

Table 17 compares the speed and performances on three So-ViT architectures with different depths. Our proposed methods predominate the SVD and NS iteration in terms of speed. To be more specific, our MPA-Lya is 3.19X faster than the NS iteration and 25.63X faster than SVD-Padé, and our MTP-Lya is 4.34X faster than the NS iteration and 34.85X faster than SVD-Padé. For the So-ViT-7 and So-ViT-10, our MPA-Lya achieves the best evaluation results and even slightly outperforms the SVD-based methods. Moreover, on the So-ViT-14 model where the performances are saturated, our method converges faster and spends fewer training epochs. The performance of our MTP-Lya is also on par with the other methods.

For the vision transformers, to accelerate the training and avoid gradient explosion, the mixed-precision techniques are often applied and the model weights are in half-precision (*i.e.,* float16). In the task of covariance pooling, the SVD often requires double precision (*i.e.,* float64) to ensure the effective numerical representation of the eigenvalues [126]. The SVD methods might not benefit from such a low precision, as large round-off errors are likely to be triggered. We expect the performance of SVD-based methods could be improved when using a higher precision. The PI suggested in the So-ViT only computes the dominant eigenpair but neglects the rest. In spite of the fast speed, the performance is not comparable with other methods.

5.1.2 Conclusions

In this research, we proposed two fast methods to compute the differentiable matrix square root. In the forward pass, the MTP and MPA are applied to approximate the matrix square root, while an iterative Lyapunov solver is proposed to solve the gradient function for back-propagation. Several numerical tests and computer vision applications demonstrate the effectiveness of our proposed model. In future work, we would like to extend our work to other applications of differentiable matrix square root, such as neural style transfer and covariance pooling for CNNs.

5.1.3 Relevant publications

- Y. Song, N. Sebe, and W. Wang, "Fast Differentiable Matrix Square Root", International Conference on Learning Representations, April 2022 [139].
  Zenodo record: https://zenodo.org/record/6396093.

5.1.4 Relevant software and/or external resources

- The Pytorch implementation can be found in `https://github.com/KingJamesSong/FastDifferentiableMatSqrt`.

5.1.5 Relevance to AI4Media use cases and media industry applications

Our tools provided in this section are generic and can be applied to a large variety of AI applications. We have provided results for ZCA-whitening (see subsection 4.1 for the use of the whitening transform) and covariance pooling used with visual transformers (see subsection 4.2 for the use of visual transformers).

5.2 Batch-efficient Eigen Decomposition for Small and Medium Matrices

Contributing partners: UNITN

The EigenDecomposition (ED) or the Singular Value Decomposition (SVD) explicitly factorize a matrix into the eigenvalue and eigenvector matrix, which serves as a fundamental tool in computer vision and deep learning. Recently, many algorithms integrated the SVD as a meta-layer into their models to perform some desired spectral transformations [140, 124, 141, 142, 127, 131, 134, 129, 143, 144, 145, 135, 126]. The applications vary in global covariance pooling [141, 146, 126], decorrelated Batch Normalization (BN) [127, 134, 129, 147], Perspective-n-Points (PnP) problems [142, 143, 144], and Whitening and Coloring Transform (WCT) [130, 131, 145].

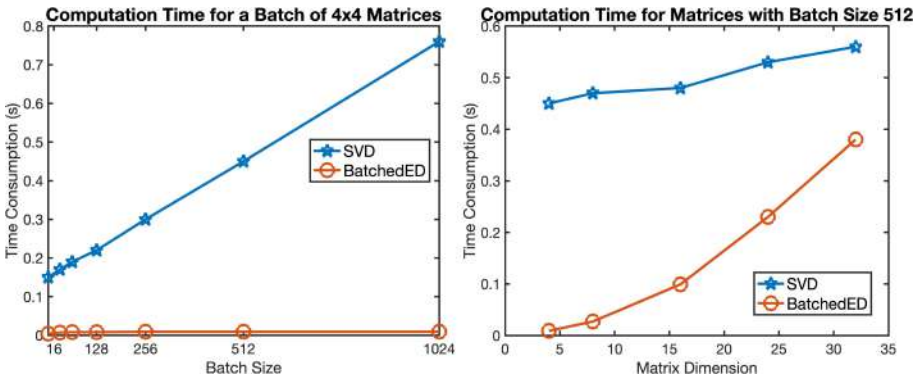


Figure 12. The speed comparison of our Batched ED against the TORCH.SVD. *(*Left*) Time consumption for a mini-batch of* 4×4 *matrices with different batch sizes. (*Right*) Time consumption for matrices with batch size* 512 *but in different matrix dimensions.*

The problem setup of the ED in computer vision is quite different from other fields. In other communities such as scientific computing, batched matrices rarely arise and the ED is usually used to process a single matrix. However, in deep learning and computer vision, the model takes a mini-batch of matrices as the input, which raises the requirement for an ED solver that works for batched matrices efficiently. Moreover, the differentiable ED works as a building block and needs to process batched matrices millions of times during the training and inference. This poses a great challenge to the efficiency of the ED solver and could even stop people from adding the ED meta-layer in their models due to the huge time consumption (see Figure 12).

In the current deep learning frameworks such as Pytorch [148] or Tensorflow [149], the ED solvers mainly adopt the SVD implementation from the linear algebra libraries (*e.g.*, LAPACK [150] and Intel MKL [151]). These solvers can efficiently process a single matrix but do not support

batched matrices on GPUs well. Most of the implementations are based on the Divide-and-Conquer (DC) algorithm [152, 153]. This algorithm partitions a matrix into multiple small sub-matrices and performs the ED simultaneously for each sub-matrix. Aided by the power of parallel and distributed computing, its speed is only mildly influenced by the matrix dimension and can be very fast for a single matrix. The core of the DC algorithm is the characteristic polynomials $\det(\lambda\mathbf{I}-\mathbf{A})=0$, which can be solved by various methods, such as secular equations [153] and spectral division [154]. However, solving the polynomial requires simultaneously localizing all the eigenvalue intervals for each individual matrix. Despite the high efficiency for a single matrix, these DC algorithms do not scale to batched matrices.

Except for the DC algorithm, some ED solvers would use the QR iteration. The QR iteration has many implementation methods and one particular batch-efficient choice is by Givens rotation. The Givens rotation can be implemented via matrix-matrix multiplications, which naturally extends to batched matrices. During the QR iterations, the Givens rotation is applied successively to annihilate the off-diagonal entries until the matrix becomes diagonal. The major drawback limiting the usage of QR iterations is the $O(n^5)$ time cost, which makes this method only applicable to tiny matrices (*e.g., dim*<9). To alleviate this issue, modern QR-based ED implementations apply the technique of *deflation* [155, 156, 157], *i.e.,* partition the matrix into many sub-matrices. The deflation technique can greatly improve the speed of the QR iterations but only works for an individual matrix. For the QR iteration, the convergence speed is related with the adjacent eigenvalue ratio $\frac{\lambda_{i+1}}{\lambda_i}$. For multiple matrices within a mini-batch, the off-diagonal entries of each matrix converge to zero with inconsistent speed and where each matrix can be partitioned is different. Consequently, the deflation technique does not apply to batched matrices either. To give a concrete example, consider 2 matrices of sizes 8×8 in a mini-batch. Suppose that the deflation would split one into two 3×3 and 5×5 matrices, while the other matrix might be partitioned into two 4×4 matrices. In this case, the partitioned matrices cannot be efficiently processed as a mini-batch due to the inconsistent matrix sizes.

To attain a batch-friendly and GPU-efficient ED method dedicated to computer vision field, we propose a QR-based ED algorithm that performs the ED via batched matrix/vector multiplication. Each step of the ED algorithm is carefully motivated for the best batch-efficient and computation-cheap consideration. We first perform a series of batched Householder reflectors to tri-diagonalize the matrix by the batched matrix-vector multiplication. Afterward, the explicit QR iteration by matrix rotation with double Wilkinson shifts [158] is conducted to diagonalize the matrix. The proposed shifts make the last two diagonal entries of the batched matrices have consistent convergence speed. Thereby the convergence is accelerated and the matrix dimension can be progressively shrunk during the QR iterations. Besides the dimension reduction, we also propose some economic computation methods based on the complexity analysis. The time complexity of QR is thus reduced from $O(n^5)$ to $O(n^3)$. The numerical tests demonstrate that, for matrices whose dimensions are smaller than 24, our Pytorch implementation is consistently much faster than the default SVD routine for any batch size. For matrices with larger dimensions (*e.g., dim*=32 or 36), our method could also have an advantage when the batch size is accordingly large (see also Figure 12). We validate the effectiveness of our method in several applications of differentiable SVD, including decorrelated BN, covariance pooling for vision transformers, and neural style transfer. Our Batched ED achieves competitive performances against the SVD.

5.2.1 Experiments

In this section, we first perform a numerical test to compare our method with SVD for matrices in different dimensions and batch sizes. Subsequently, we evaluate the effectiveness of the proposed methods in three computer vision applications: decorrelated BN, second-order vision transformer,

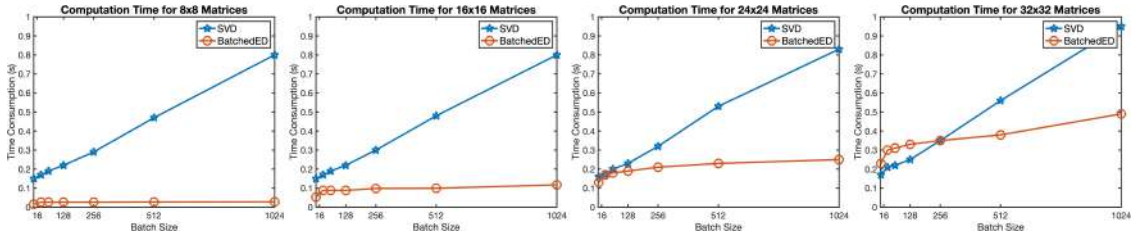


Figure 13. The speed comparison of our Batched ED against TORCH.SVD *for different batch sizes and matrix dimensions. Our implementation is more batch-friendly and the time cost does not vary much against different batch sizes. For matrices in small and moderate sizes, our method can be significantly faster than the Pytorch SVD.*

and neural style transfer.

5.2.1.1 Numerical Test Figure 13 depicts the computational time of our Batched ED against the SVD for different matrix dimensions and batch sizes. The time cost of the SVD grows almost linearly with the batch size, while the time consumption of our Batched ED only has slight or mild changes against varying batch sizes. For matrices whose dimensions are smaller than 24, our Batched ED is consistently faster than the SVD for any batch size. When the matrix dimension is 32, our method is faster than the SVD from batch size 256 on. The speed of our Batched ED is more advantageous for smaller matrix dimensions and larger batch sizes.

Table 18. Validation error of decorrelated BN on ResNet-18 [4]. The results are reported based on 5 runs, and we measure the time of the forward ED in a single step.

Solver	Group	Size	Time (s)	CIFAR10		CIFAR100	
				mean±std	min	mean±std	min
SVD	16	16×4×4	0.172	4.52±0.09	4.33	**21.24±0.17**	20.99
Batched ED			**0.006**	**4.37±0.11**	**4.29**	21.25±0.20	**20.90**
SVD	8	8×8×8	0.170	4.55±0.13	4.34	21.32±0.31	20.88
Batched ED			**0.016**	**4.36±0.11**	**4.25**	**20.97±0.27**	**20.62**
SVD	4	4×16×16	0.165	4.52±0.14	4.33	21.30±0.33	**20.86**
Batched ED			**0.075**	**4.45±0.11**	**4.32**	**21.19±0.21**	20.98

5.2.1.2 Decorrelated BN Following [147], we first conduct an experiment on the task of ZCA whitening. In the whitening process, the inverse square root of the covariance is multiplied with the feature as $(\boldsymbol{X}\boldsymbol{X}^T)^{-\frac{1}{2}}\boldsymbol{X}$ to eliminate the correlation between each dimension. We insert the ZCA whitening meta-layer into the ResNet-18 [4] architecture and evaluate the validation error on CIFAR10 and CIFAR100 [138]. Table 18 compares the performance of our Batched ED against the SVD. Depending on the number of groups, our method can be 2X faster, 10X faster, and even 28X faster than the SVD. Furthermore, our method outperforms the SVD across all the metrics on CIFAR10. With CIFAR100, the performance is also on par.

5.2.1.3 Second-order Vision Transformer We turn to the experiment on the task of global covariance pooling for the Second-order Vision Transformer (So-ViT) [159]. To leverage the rich semantics embedded in the visual tokens, the covariance square root of the visual tokens $(\boldsymbol{X}\boldsymbol{X}^T)^{\frac{1}{2}}$

Table 19. Validation accuracy on ImageNet [3] for the second-order vision transformer with different depths. Here 32 and 36 denote the spatial dimension of visual tokens. We report the time consumption of the forward ED in a single step.

Solver	Size	Time (s)	Architecture	
			So-ViT-7	So-ViT-10
SVD	768×32×32	0.767	76.01 / **93.10**	**77.97** / **94.10**
Batched ED		**0.431**	**76.04** / 93.05	77.91 / 94.08
SVD	768×36×36	0.835	**76.10** / **93.14**	78.09 / 94.13
Batched ED		**0.612**	76.07 / 93.10	**78.11** / **94.19**

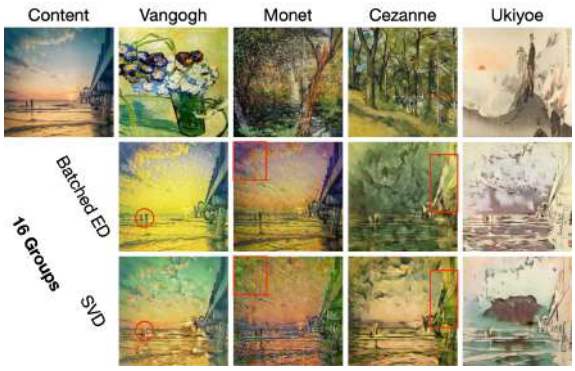


Figure 14. Exemplary visual comparison. The red circle/rectangular indicates the region with subtle details. In this example, our method generates sharper images with more coherent style information and less artifacts. Zoom in for a better view.

are used to assist the classification task. Since the global covariance matrices are typically very ill-conditioned [126], this task poses a huge challenge to the stability of the ED algorithm. We choose the So-ViT architecture with different depths and validate the performance on ImageNet [3]. As observed from Table 19, our Batched ED has the competitive performance against the standard SVD. Moreover, our method is about 44% and 27% faster than the SVD for covariance in different sizes.

5.2.1.4 Universal Style Transfer Now we apply our Batched ED in the WCT for neural style transfer. Given the content feature $\boldsymbol{X}_c$ and the style feature $\boldsymbol{X}_s$, the WCT performs successive whitening $((\boldsymbol{X}_c\boldsymbol{X}_c)^{-\frac{1}{2}}\boldsymbol{X}_c)$ and coloring $((\boldsymbol{X}_s\boldsymbol{X}_s)^{\frac{1}{2}}\boldsymbol{X}_c)$ to transfer the target style. We follow [130, 145] to use the LPIPS distance and the user preference as the evaluation metrics. Table 20 presents the quantitative comparison with different groups. Our Batched ED achieves very competitive performance and predominates the speed. To give a concrete example, when the group number is 64, our method is about 35X faster than the default SVD. Figure 14 displays the exemplary visual comparison. In this specific example, our Batched ED generates images with better visual appeal.

Similar to the finding in [131], we also observe that the number of groups has an impact on the extent of transferred style. As shown in Figure 15, when more groups are used, the style in the transferred image becomes more distinguishable and the details are better preserved. Since the number of groups determines the number of divided channels and the covariance size, more

Table 20. The LPIPS distance between the transferred image and the content image and the user preference (%) on the Artworks [5] dataset. We report the time consumption of the forward ED that is conducted 10 times to exchange the style and content feature at different network depths. The batch size is set to 4.

Solver	Group	Size	Time (s)	LPIPS [160] (↑)	Preference (↑)
SVD	64	256×4×4	3.146	0.5776	**48.25**
Batched ED			**0.089**	**0.5798**	47.75
SVD	32	128×8×8	2.306	**0.5722**	47.75
Batched ED			**0.257**	0.5700	**48.75**
SVD	16	64×16×16	1.973	0.5614	46.25
Batched ED			**0.876**	**0.5694**	**47.75**

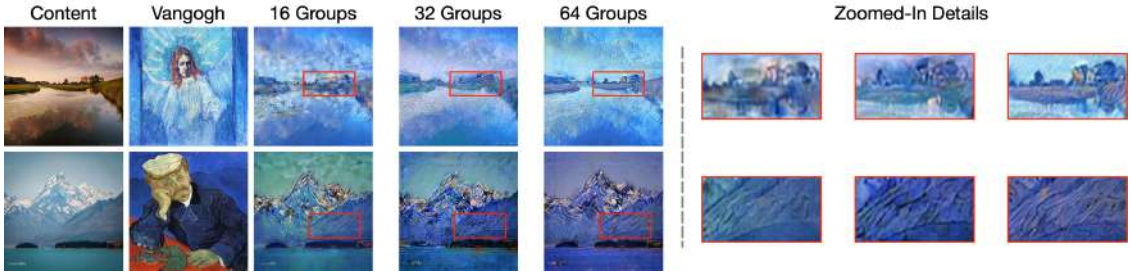


Figure 15. Visual illustration of the impact of groups. When more groups are used, the strength of the target style is increased and the details are better preserved.

groups correspond to smaller covariance and this might help to better capture the local structure. Despite this superficial conjecture, giving a more comprehensive and rigorous analysis is worth further research.

To sum up, our ED solver has demonstrated the superior batch efficiency for small matrices in various real-world experiments and numerical tests. The limitation on large matrices indicates the key difference: *our method is more batch-efficient, while* TORCH.EIG/SVD *is more dimension-efficient.*

5.2.2 Conclusions

The contributions of this research are as follows:

- We propose an ED algorithm for a mini-batch of small and medium matrices which is dedicated to many application scenarios of computer vision. Each step of ED is carefully motivated and designed for the best batch efficiency.
- We propose dedicated acceleration techniques for our Batched ED algorithm. The progressive dimension shrinkage is proposed to reduce the matrix size during the iterations, while some economic computation methods grounded on the complexity analysis are also developed.
- Our batch-efficient ED algorithm is validated in several applications of differentiable SVD. The experiments on visual recognition and image generation demonstrate that our method achieves very competitive performances against the SVD encapsuled in the current deep learning platforms.

5.2.3 Relevant publications

- Y. Song, N. Sebe, and W. Wang, Batch-efficient Eigen Decomposition for Small and Medium Matrices, European Conference on Computer Vision, October 2022 [161].
  Zenodo record: https://zenodo.org/record/7566134.

5.2.4 Relevant software and/or external resources

- The Pytorch implementation can be found in https://github.com/KingJamesSong/BatchED.

5.2.5 Relevance to AI4Media use cases and media industry applications

Our tools provided in this section are generic and can be applied to a large variety of applications. In the section we have provided evidences of their use in visual transformers and style transfer but the applicability is very large as ED is used practically always when matrices are involved.

6 Ongoing Work and Conclusions

This document presents the outcomes of AI4Media research activities in Task 5.5 during months M19-M36 of the project, which correspond to the first 18 months of the duration of the task. The contributions mainly deal with the task of image and video super-resolution, and efficient methods for deep neural network training. A dataset and a quality assessment workflow have been presented to contribute to the field of super-resolution, as well as a super-resolution detector neural network architecture. Efficient algorithms have been proposed for self-supervised learning and the training of Visual Transformers. More efficient versions of mathematical matrix operations commonly used in deep learning have also been developed. As a result, this task has produced a total of 4 conference publications and 4 open source software implementations. And as previously reported, 5 out of 7 contributions are mapped to one or more AI4Media use cases, while the remaining 2 are of general purpose and interest.

Lastly, below, we summarize the ongoing work and plans for future work of each associated partner in relation with this task:

- **BSC** is currently working on a publication expanding the contribution of the SUDDS detector architecture, including extra experimentation like leave-one-out cross-validation (LOOCV) over the SR methods to detect for better performance assessment. Also related to SR detection, we will work on expanding the BSC4K dataset with synthetically upscaled images of more SR models. We also intend to expand the BSC4K dataset with more diverse content, with the ultimate goal of being of use to finetune a blind SR model, before publishing it with an open-access license. We will work on a publication presenting the contributions of the BSC4K dataset too. Finally, we will collaborate with RAI in the creation of a video domain and content multi-dimensional content type classification scheme, and a VSR model selection pipeline that will be used in UC3.
- **UNITN** will systematically study how to improve the covariance conditioning by enforcing orthogonality to the Pre-SVD layer. Existing orthogonal treatments on the weights will be first investigated. However, these techniques can improve the conditioning but would hurt the performance. To avoid such a side effect, we will consider two solutions, i.e., the Nearest Orthogonal Gradient (NOG) and Optimal Learning Rate (OLR) applied to decorrelated Batch Normalization (BN) and Global Covariance Pooling (GCP).
- **RAI** will persist the investigation on the best objective metric to optimise the selection of the best SR model for each video considering non-reference metrics as well. These metrics do not rely on comparing with the original file, which is useful if the high resolution version of the original file is not available. Furthemore, RAI will continue the investigation of the relations between content properties (e.g., genre, motion complexity and resolution) and SR deep models. The primary goal is to derive less sophisticated solutions for SR tasks. In this way, more complex architectures will be limited to a smaller number of scenarios. Consequently, the overall computational and energy-related requirements for training will be substantially optimized.
- **UNIFI** will work on the detection of synthetically generated or upscaled images, by exploiting generative models. We will follow the line of work by [162] studying how reconstruction error computed via generative models can be used as a score to detect synthetic images. In particular, we will evaluate the efficacy of Consistency Models to detect generated images and also attribute the image to the respective model. In the case of locally edited images, we will also evaluate the use of the obtained features to localize such alterations.

References

[1] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *ICML*, 2020.

[2] X. Chen and K. He, "Exploring simple siamese representation learning," *arXiv:2011.10566*, 2020.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[5] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *CVPR*, 2017.

[6] W. Lu, W. Sun, X. Min, W. Zhu, Q. Zhou, J. He, Q. Wang, Z. Zhang, T. Wang, and G. Zhai, "Deep neural network for blind visual quality assessment of 4k content," *IEEE Transactions on Broadcasting*, 2022.

[7] U. Cisco, "Cisco annual internet report (2018–2023) white paper," *Cisco: San Jose, CA, USA*, vol. 10, no. 1, pp. 1–35, 2020.

[8] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu, "Coca: Contrastive captioners are image-text foundation models," *arXiv preprint arXiv:2205.01917*, 2022.

[9] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li, *et al.*, "Internimage: Exploring large-scale vision foundation models with deformable convolutions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14408–14419, 2023.

[10] W. Ahmad, H. Ali, Z. Shah, and S. Azmat, "A new generative adversarial network for medical images super resolution," *Scientific Reports*, vol. 12, no. 1, p. 9533, 2022.

[11] C. You, G. Li, Y. Zhang, X. Zhang, H. Shan, M. Li, S. Ju, Z. Zhao, Z. Zhang, W. Cong, *et al.*, "Ct super-resolution gan constrained by the identical, residual, and cycle learning ensemble (gan-circle)," *IEEE transactions on medical imaging*, vol. 39, no. 1, pp. 188–203, 2019.

[12] Y. Chen, Y. Xie, Z. Zhou, F. Shi, A. G. Christodoulou, and D. Li, "Brain mri super resolution using 3d deep densely connected neural networks," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pp. 739–742, IEEE, 2018.

[13] A. Aakerberg, K. Nasrollahi, and T. B. Moeslund, "Real-world super-resolution of face-images from surveillance cameras," *IET Image Processing*, vol. 16, no. 2, pp. 442–452, 2022.

[14] S. P. Mudunuri and S. Biswas, "Low resolution face recognition across variations in pose and illumination," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 5, pp. 1034–1040, 2015.

[15] D. Zhang, J. Shao, X. Li, and H. T. Shen, "Remote sensing image super-resolution via mixed high-order attention network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 6, pp. 5183–5196, 2020.

[16] J. Gu, X. Sun, Y. Zhang, K. Fu, and L. Wang, "Deep residual squeeze and excitation network for remote sensing image super-resolution," *Remote Sensing*, vol. 11, no. 15, p. 1817, 2019.

[17] NVIDIA, "Pixel perfect: Rtx video super resolution now available — nvidia blog." https://blogs.nvidia.com/blog/2023/02/28/rtx-video-super-resolution/, 2023.

[18] V. Vavilala and M. Meyer, "Deep learned super resolution for feature film production," in *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks*, pp. 1–2, 2020.

[19] SAMSUNG, "How to use the intelligent mode of samsung qled tv — samsung ca." https://www.samsung.com/ca/support/tv-audio-video/how-to-use-the-intelligent-mode-of-samsung-qled-tvs/.

[20] T. Köhler, M. Bätz, F. Naderi, A. Kaup, A. Maier, and C. Riess, "Toward bridging the simulated-to-real gap: Benchmarking super-resolution on real data," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 11, pp. 2944–2959, 2019.

[21] H. Chen, X. He, L. Qing, Y. Wu, C. Ren, R. E. Sheriff, and C. Zhu, "Real-world single image super-resolution: A brief review," *Information Fusion*, vol. 79, pp. 124–145, 2022.

[22] A. Liu, Y. Liu, J. Gu, Y. Qiao, and C. Dong, "Blind image super-resolution: A survey and beyond," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 5, pp. 5461–5480, 2022.

[23] W. Zhang, G. Shi, Y. Liu, C. Dong, and X.-M. Wu, "A closer look at blind super-resolution: Degradation models, baselines, and performance upper bounds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 527–536, 2022.

[24] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *Proceedings of the European conference on computer vision (ECCV) workshops*, pp. 0–0, 2018.

[25] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2472–2481, 2018.

[26] K. C. Chan, X. Wang, K. Yu, C. Dong, and C. C. Loy, "Basicvsr: The search for essential components in video super-resolution and beyond," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4947–4956, 2021.

[27] J. Liang, Y. Fan, X. Xiang, R. Ranjan, E. Ilg, S. Green, J. Cao, K. Zhang, R. Timofte, and L. V. Gool, "Recurrent video restoration transformer with guided deformable attention," *Advances in Neural Information Processing Systems*, vol. 35, pp. 378–393, 2022.

[28] X. Wang, L. Xie, C. Dong, and Y. Shan, "Real-esrgan: Training real-world blind super-resolution with pure synthetic data," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1905–1914, 2021.

[29] K. Zhang, J. Liang, L. Van Gool, and R. Timofte, "Designing a practical degradation model for deep blind image super-resolution," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4791–4800, 2021.

[30] S. Winkler, "Analysis of public image and video databases for quality assessment," *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 6, pp. 616–625, 2012.

[31] I. Rec, "P. 910: Subjective video quality assessment methods for multimedia applications," *International Telecommunication Union, Geneva*, vol. 2, 2008.

[32] Y. Liu, A. Liu, J. Gu, Z. Zhang, W. Wu, Y. Qiao, and C. Dong, "Discovering distinctive" semantics" in super-resolution networks," *arXiv preprint arXiv:2108.00406*, 2021.

[33] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[34] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[35] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[36] K. C. Chan, S. Zhou, X. Xu, and C. C. Loy, "Investigating tradeoffs in real-world video super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5962–5971, 2022.

[37] 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.

[38] A. Blattmann, R. Rombach, H. Ling, T. Dockhorn, S. W. Kim, S. Fidler, and K. Kreis, "Align your latents: High-resolution video synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22563–22575, 2023.

[39] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi, "Palette: Image-to-image diffusion models," in *ACM SIGGRAPH 2022 Conference Proceedings*, pp. 1–10, 2022.

[40] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding," *Advances in Neural Information Processing Systems*, vol. 35, pp. 36479–36494, 2022.

[41] C. Shen, M. Kasra, W. Pan, G. A. Bassett, Y. Malloch, and J. F. O'Brien, "Fake images: The effects of source, intermediary, and digital media literacy on contextual assessment of image credibility online," *New media & society*, vol. 21, no. 2, pp. 438–463, 2019.

[42] M. Zanardelli, F. Guerrini, R. Leonardi, and N. Adami, "Image forgery detection: a survey of recent deep-learning approaches," *Multimedia Tools and Applications*, vol. 82, no. 12, pp. 17521–17566, 2023.

[43] R. J. Chen, M. Y. Lu, T. Y. Chen, D. F. Williamson, and F. Mahmood, "Synthetic data in machine learning for medicine and healthcare," *Nature Biomedical Engineering*, vol. 5, no. 6, pp. 493–497, 2021.

[44] B. G. Matthew Ferraro, "The other side says your evidence is a deepfake. now what?." https://www.wilmerhale.com/insights/publications/20221221-the-other-side-says-your-evidence-is-a-deepfake-now-what, 2022.

[45] "Regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts." https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A52021PC0206, 2021.

[46] W. Sun, H. Duan, X. Min, L. Chen, and G. Zhai, "Blind quality assessment for in-the-wild images via hierarchical feature fusion strategy," in *2022 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 01–06, IEEE, 2022.

[47] Z. Yang, Y. Dong, L. Song, R. Xie, L. Li, and Y. Feng, "Native resolution detection for 4k-uhd videos," in *2020 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 1–5, IEEE, 2020.

[48] D. Ma, F. Zhang, and D. Bull, "Bvi-dvc: a training database for deep video compression," *IEEE Transactions on Multimedia*, 2021.

[49] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.

[50] M. Hall-Beyer, "Glcm texture: A tutorial v. 3.0 march 2017," 2017.

[51] R. R. Shah, V. A. Akundy, and Z. Wang, "Real versus fake 4k-authentic resolution assessment," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2185–2189, IEEE, 2021.

[52] V. Meshchaninov, I. Molodetskikh, and D. Vatolin, "Combining contrastive and supervised learning for video super-resolution detection," *arXiv preprint arXiv:2205.10406*, 2022.

[53] M. Montagnuolo and A. Messina, "Multimedia knowledge representation for automatic annotation of broadcast TV archives," *Journal of Digital Information Management*, vol. 5, no. 2, pp. 67–74, 2007.

[54] N. Chervyakov, P. Lyakhov, and N. Nagornov, "Analysis of the quantization noise in discrete wavelet transform filters for 3D medical imaging," *Applied Sciences (Switzerland)*, vol. 10, no. 4, 2020.

[55] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[56] I. Radiocommunication Bureau, *Recommendation ITU-R BT.500-14 Methodologies for the subjective assessment of the quality of television images BT Series Broadcasting service (television)*, vol. 14. 2020.

[57] ITU-T Recommendation P.913, "Methods for the subjective assessment of video quality, audio quality and audiovisual quality of Internet video and distribution quality television in any environment," *Recommendation ITU-T P.913*, 2021.

[58] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv:1301.3781*, 2013.

[59] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NeurIPS*, 2013.

[60] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL,*, 2019.

[61] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *ICCV*, 2015.

[62] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: Unsupervised learning using temporal order verification," in *ECCV*, 2016.

[63] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet, and A. Zisserman, "Temporal cycle-consistency learning," in *CVPR*, 2019.

[64] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, 2016.

[65] I. Misra and L. van der Maaten, "Self-supervised learning of pretext-invariant representations," in *CVPR*, 2020.

[66] O. J. Hénaff, A. Razavi, C. Doersch, S. M. A. Eslami, and A. van den Oord, "Data-efficient image recognition with contrastive predictive coding," *arXiv:1905.09272*, 2019.

[67] X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information clustering for unsupervised image classification and segmentation," in *ICCV*, 2019.

[68] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *CVPR*, 2020.

[69] Z. Wu, Y. Xiong, S. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance-level discrimination," *arXiv:1805.01978*, 2018.

[70] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *NIPS*, 2016.

[71] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *CVPR*, 2015.

[72] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv:1703.07737*, 2017.

[73] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *CVPR*, 2006.

[74] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.

[75] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv:1807.03748*, 2018.

[76] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," in *ICLR*, 2019.

[77] M. Tschannen, J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic, "On mutual information maximization for representation learning," *arXiv:1907.13625*, 2019.

[78] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020.

[79] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent: A new approach to self-supervised learning," in *NeurIPS*, 2020.

[80] A. Fetterman and J. Albrecht, "Understanding self-supervised and contrastive learning with bootstrap your own latent (BYOL)," *https://untitled-ai.github.io/understanding-self-supervised-contrastive-learning.html*, 2020.

[81] Y. Tian, L. Yu, X. Chen, and S. Ganguli, "Understanding self-supervised learning with dual deep networks," *arXiv:2010.00578*, 2020.

[82] P. H. Richemond, J.-B. Grill, F. Altché, C. Tallec, F. Strub, A. Brock, S. Smith, S. De, R. Pascanu, B. Piot, and M. Valko, "Byol works even without batch statistics," *arXiv:2010.10241*, 2020.

[83] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.

[84] Y. You, I. Gitman, and B. Ginsburg, "Large batch training of convolutional networks," *arXiv:1708.03888*, 2017.

[85] A. Siarohin, E. Sangineto, and N. Sebe, "Whitening and coloring transform for GANs," in *International Conference on Learning Representations*, 2019.

[86] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[87] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," 2015.

[88] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *ECCV*, 2020.

[89] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *AISTATS*, 2011.

[90] X. Chen, H. Fan, R. B. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv:2003.04297*, 2020.

[91] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *NeurIPS*, 2020.

[92] A. Ermolov, A. Siarohin, E. Sangineto, and N. Sebe, "Whitening for self-supervised representation learning," in *International Conference on Machine Learning*, 2021.

[93] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[94] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," *arXiv:2012.12877*, 2020.

[95] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z. Jiang, F. E. Tay, J. Feng, and S. Yan, "Tokens-to-token ViT: Training vision transformers from scratch on ImageNet," in *ICCV*, 2021.

[96] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *arXiv:2103.14030*, 2021.

[97] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "CvT: Introducing convolutions to vision transformers," *arXiv:2103.15808*, 2021.

[98] K. Yuan, S. Guo, Z. Liu, A. Zhou, F. Yu, and W. Wu, "Incorporating convolution designs into visual transformers," *arXiv:2103.11816*, 2021.

[99] Y. Li, K. Zhang, J. Cao, R. Timofte, and L. V. Gool, "LocalViT: Bringing locality to vision transformers," *arXiv:2104.05707*, 2021.

[100] W. Xu, Y. Xu, T. Chang, and Z. Tu, "Co-scale conv-attentional image transformers," *arXiv:2104.06399*, 2021.

[101] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*, 2020.

[102] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," in *ICLR*, 2021.

[103] Z. Dai, B. Cai, Y. Lin, and J. Chen, "UP-DETR: unsupervised pre-training for object detection with transformers," in *CVPR*, 2021.

[104] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *ICCV*, 2021.

[105] T. Meinhardt, A. Kirillov, L. Leal-Taixé, and C. Feichtenhofer, "TrackFormer: Multi-object tracking with transformers," *arXiv:2101.02702*, 2021.

[106] Y. Jiang, S. Chang, and Z. Wang, "TransGAN: Two transformers can make one strong GAN," *arXiv:2102.07074*, 2021.

[107] D. A. Hudson and C. L. Zitnick, "Generative Adversarial Transformers," in *ICML*, 2021.

[108] H. Zhao, L. Jiang, J. Jia, P. H. S. Torr, and V. Koltun, "Point transformer," *arXiv:2012.09164*, 2020.

[109] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.

[110] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018.

[111] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[112] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do vision transformers see like convolutional neural networks?," *arXiv:2108.08810*, 2021.

[113] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," in *ICLR*, 2020.

[114] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, 2016.

[115] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *ECCV*, 2020.

[116] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.

[117] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[118] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *CVPR*, 2019.

[119] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," *Int. J. Comput. Vis.*, vol. 128, no. 3, pp. 642–656, 2020.

[120] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, "Detnet: Design backbone for object detection," in *ECCV*, 2018.

[121] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *ICLR*, 2018.

[122] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "CutMix: Regularization strategy to train strong classifiers with localizable features," in *ICCV*, 2019.

[123] Y. Liu, E. Sangineto, W. Bi, N. Sebe, B. Lepri, and M. De Nadai, "Efficient training of visual transformers with small-size datasets," in *Neural Information Processing Systems*, 2021.

[124] T.-Y. Lin and S. Maji, "Improved bilinear pooling with cnns," *arXiv preprint arXiv:1707.06772*, 2017.

[125] P. Li, J. Xie, Q. Wang, and Z. Gao, "Towards faster training of global covariance pooling networks by iterative matrix square root normalization," in *CVPR*, 2018.

[126] Y. Song, N. Sebe, and W. Wang, "Why approximate matrix square root outperforms accurate svd in global covariance pooling?," in *ICCV*, 2021.

[127] L. Huang, D. Yang, B. Lang, and J. Deng, "Decorrelated batch normalization," in *CVPR*, 2018.

[128] L. Huang, Y. Zhou, F. Zhu, L. Liu, and L. Shao, "Iterative normalization: Beyond standardization towards efficient whitening," in *CVPR*, 2019.

[129] L. Huang, L. Zhao, Y. Zhou, F. Zhu, L. Liu, and L. Shao, "An investigation into the stochasticity of batch whitening," in *CVPR*, 2020.

[130] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, "Universal style transfer via feature transforms," in *NeurIPS*, 2017.

[131] W. Cho, S. Choi, D. K. Park, I. Shin, and J. Choo, "Image-to-image translation via group-wise deep whitening-and-coloring transformation," in *CVPR*, 2019.

[132] 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.

[133] C. Ionescu, O. Vantzos, and C. Sminchisescu, "Training deep networks with structured layers by matrix backpropagation," *arXiv preprint arXiv:1509.07838*, 2015.

[134] W. Wang, Z. Dang, Y. Hu, P. Fua, and M. Salzmann, "Backpropagation-friendly eigendecomposition," in *NeurIPS*, 2019.

[135] W. Wang, Z. Dang, Y. Hu, P. Fua, and M. Salzmann, "Robust differentiable svd," *TPAMI*, 2021.

[136] G. Schulz, "Iterative berechung der reziproken matrix," *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 13, no. 1, pp. 57–59, 1933.

[137] N. J. Higham, *Functions of matrices: theory and computation*. SIAM, 2008.

[138] A. Krizhevsky, "Learning multiple layers of features from tiny images," *Master's thesis, University of Tront*, 2009.

[139] Y. Song, N. Sebe, and W. Wang, "Fast differentiable matrix square root," in *International Conference on Learning Representations*, 2022.

[140] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *ICCV*, 2015.

[141] P. Li, J. Xie, Q. Wang, and W. Zuo, "Is second-order information helpful for large-scale visual recognition?," in *ICCV*, 2017.

[142] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother, "Dsac-differentiable ransac for camera localization," in *CVPR*, 2017.

[143] D. Campbell, L. Liu, and S. Gould, "Solving the blind perspective-n-point problem end-to-end with robust differentiable geometric optimization," in *ECCV*, 2020.

[144] Z. Dang, K. M. Yi, Y. Hu, F. Wang, P. Fua, and M. Salzmann, "Eigendecomposition-free training of deep networks for linear least-square problems," *TPAMI*, 2020.

[145] Z. Wang, L. Zhao, H. Chen, L. Qiu, Q. Mo, S. Lin, W. Xing, and D. Lu, "Diversified arbitrary style transfer via deep feature perturbation," in *CVPR*, 2020.

[146] Q. Wang, J. Xie, W. Zuo, L. Zhang, and P. Li, "Deep cnns meet global covariance pooling: Better representation and generalization," *TPAMI*, 2020.

[147] Y. Song, N. Sebe, and W. Wang, "Fast differentiable matrix square root," in *ICLR*, 2022.

[148] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS*, 2019.

[149] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "{TensorFlow}: A system for {Large-Scale} machine learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pp. 265–283, 2016.

[150] E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, *et al.*, *LAPACK Users' guide*. SIAM, 1999.

[151] E. Wang, Q. Zhang, B. Shen, G. Zhang, X. Lu, Q. Wu, and Y. Wang, "Intel math kernel library," in *High-Performance Computing on the Intel® Xeon Phi™*, pp. 167–188, Springer, 2014.

[152] J. J. Cuppen, "A divide and conquer method for the symmetric tridiagonal eigenproblem," *Numerische Mathematik*, vol. 36, no. 2, pp. 177–195, 1980.

[153] M. Gu and S. C. Eisenstat, "A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem," *SIAM Journal on Matrix Analysis and Applications*, vol. 16, no. 1, pp. 172–191, 1995.

[154] Y. Nakatsukasa and N. J. Higham, "Stable and efficient spectral divide and conquer algorithms for the symmetric eigenvalue decomposition and the svd," *SIAM Journal on Scientific Computing*, vol. 35, no. 3, pp. A1325–A1349, 2013.

[155] M. Ahues and F. Tisseur, "A new deflation criterion for the qr algorithm," *LAPACK Working Note*, vol. 122, 1997.

[156] K. Braman, R. Byers, and R. Mathias, "The multishift qr algorithm. part i: Maintaining well-focused shifts and level 3 performance," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 4, pp. 929–947, 2002.

[157] K. Braman, R. Byers, and R. Mathias, "The multishift qr algorithm. part ii: Aggressive early deflation," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 4, pp. 948–973, 2002.

[158] J. Wilkinson, "The algebraic eigenvalue problem," in *Handbook for Automatic Computation, Volume II, Linear Algebra*, Springer-Verlag New York, 1971.

[159] J. Xie, R. Zeng, Q. Wang, Z. Zhou, and P. Li, "So-vit: Mind visual tokens for vision transformer," *arXiv preprint arXiv:2104.10935*, 2021.

[160] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.

[161] Y. Song, N. Sebe, and W. Wang, "Batch-efficient eigen decomposition for small and medium matrices," in *European Conference on Computer Vision*, 2022.

[162] Z. Wang, J. Bao, W. Zhou, W. Wang, H. Hu, H. Chen, and H. Li, "Dire for diffusion-generated image detection," 2023.