Self-supervised Learning for different Downstream Tasks

Master Project in Artificial Intelligence

submitted
by

Vishnudev Krishnadas Vishnudev Krishnadas

born 24.02.1997 in Kerala, India

Written at

Lehrstuhl für Mustererkennung (Informatik 5)
Department Informatik
Friedrich-Alexander-Universität Erlangen-Nürnberg.

Advisor: Dr.-Ing. Vincent Christlein

Abstract

This project presents a comparison of self-supervised learning methods for different downstream tasks in the context of Medieval Handwriting in the Latin Script dataset. Self-supervised learning has shown promise in various computer vision and natural language processing applications, but its effectiveness on historical scripts has not been extensively explored. Three self-supervised learning methods, namely, SimCLR, MAE, and BYOL are compared in this work. The performance evaluation was conducted on one downstream tasks i.e. script type classification. The results indicate that the SimCLR method outperforms other methods in the downstream task for the Medieval Handwritings Script dataset. Additionally, insights were provided regarding the factors influencing the performance of self-supervised learning methods in this context, including the selection of pre-training data and the size of the pre-training dataset. In conclusion, this study showcases the potential of self-supervised learning for historical handwritten document classification tasks and emphasizes the significance of selecting suitable methods for specific downstream tasks.

Contents

Chapter 1

Introduction

The increasing digitization of historical documents has created demand for efficient and accurate methods to categorize and organize these valuable ancient scripts. The field of Deep Learning has met with great success in many problems involving complex data [Naj+15]. The majority of the available digitized corpus is unlabeled and lacks metadata. Consequently, an unsupervised approach to learning is necessary which takes advantage of the large amounts of unlabeled data. Hence, this master project evaluates the efficacy of Deep Learning based Self-supervised Learning approaches in the context of historical documents. Self-supervised learning models extract intrinsic patterns from the data, which is used for knowledge transfer to the target task.

This project focuses on the style classification of Medieval Handwritings in Latin Script and is the target task in this work. Style classification may also be referred to as script type classification in this report. The Script type classification refers to the task of categorizing Latin scripts according to morphological differences in the handwritten text [Clo+17]. Representations produced by the self-supervised model are employed to enhance the accuracy of script type prediction by utilizing the morphology present in the text. For this study, it was of interest to investigate Simple Framework for Contrastive Learning of Visual Representations (SimCLR) [Che+20], Masked Autoencoders (MAE) [He+21], and Bootstrap Your Own Latent (BYOL) [Gri+20] Self-supervised Learning (SSL) methods. The scope of this project is to compare the quality of generated representations by the mentioned self-supervised learning techniques.

The upcoming chapters of this study will consist of an overview of the dataset and the self-supervised algorithms employed. In addition, the methods utilized will be carefully examined. Subsequently, the results and analysis of the study will be presented.

Chapter 2

Materials and Methods

Self-supervised representation learning is used to obtain visual features from a large scale of unlabeled data. These visual features are discovered by learning the objective function of a pretext task. To assess the quality of features generated by the pretext task, a downstream task based on an application is selected. Knowledge transfer from the pre-trained model in the pretext task proves valuable for the downstream or target task [Eri+22]. In this project, the target task is defined as script-type classification for Medieval Latin scripts.

2.1 Dataset

The dataset used in this project comes from the International Conference on Document Analysis (ICDAR) 2017 competition on the Classification of Latin Medieval Manuscripts (CLaMM) [Clo+17]. All self-supervised models used were pre-trained on a dataset with 70% of 5,540 images combining the *Training dataset* and *Tasks 1 & 3 dataset* provided on the competition portal [ICD17]. Then, a linear evaluation protocol was followed to evaluate the model's performance and compare it with other models. The subsequent sections detail the setup for the state-of-the-art SSL methods used in this project.

2.2 Software and Hardware environment

All the methods were implemented with Pytorch Lightning [Fal19] and the Hydra framework [Yad19] for configurations and experiments. The experiments were executed on a High Performance Cluster with GPUs using SLURM jobs. The GPUs used for pre-training were Nvidia GeForce RTX3080 and Nvidia A100 Tensor Core.

2.3 Self-supervised learning techniques

2.3.1 Simple Framework for Contrastive Learning of Visual Representations

SimCLR has garnered considerable attention in the machine learning community due to its efficacy in learning meaningful representations from unlabeled data. The SimCLR architecture comprises several key components: a data augmentation module, a base encoder (ResNet-50), a compact projection head, and a contrastive loss function [Che+20]. In the following discussion, we delve into the configuration of these components.

- **Data augmentation** module plays a pivotal role and encompasses a sequential series of transformations applied probabilistically. These transformations include random resized crop, horizontal flip, gaussian blur, random rotation, random erasing, dilation, erosion, and normalization. By simulating various nuances found in historical handwritten documents — such as smudging, ink bleeding, discoloration, folds, stains, and others — the augmentations contribute significantly to the model's generalization capabilities.
- **Backbone architecture.** In order to extract representations from the augmented data, the ResNet-50 neural network encoder architecture is employed. This architecture has been proven effective in capturing relevant features from the input data.
- **Projection head** serves as a crucial component responsible for reducing the dimensionality of the data. It consists of a sequential neural network comprising two linear layers, with batch normalization and ReLU activation applied in between. This configuration enables the projection head to extract compact representations from the higher-dimensional space.
- **Loss function.** As recommended by Ting Chen et al. [Che+20], the Normalized Temperature-scaled Cross Entropy Loss is employed as the contrastive loss function. This loss function aids in maximizing agreement between differently augmented views of the same sample while minimizing agreement between views of different samples.

2.3.2 Masked Autoencoders

MAE can serve as powerful pretraining models for deep neural networks. By training an autoencoder on a large unlabelled dataset, it can learn general features or representations

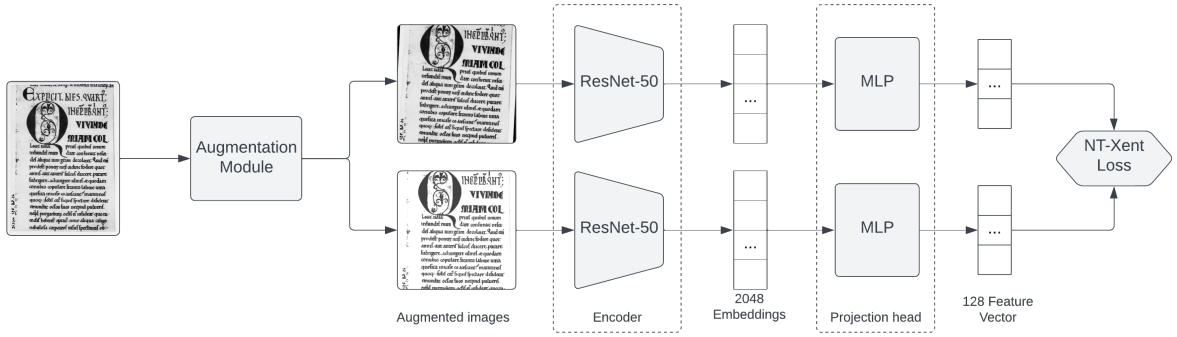


Figure 2.1: SimCLR Architecture

that capture the statistical regularities of the data. MAE comprise two key components: an encoder consisting of Vision Transformer (ViT) blocks in series and a lightweight decoder that reconstructs the input from the representation generated by the encoder. Another essential technique included is an optimal masking ratio of 75% to balance the accuracy and generalization of the model [He+21].

A set of five ViT architecture sizes from ViT-Tiny to ViT-Huge were evaluated for the encoder. The encoder operates on a visible subset of patches without the mask tokens and a shallow decoder reconstructs the original image from the representation generated by the encoder. Simple transformations of resize, crop and flip are applied. Xavier Uniform technique is used to initialize the weights for all the transformer blocks. The mean squared error at the pixel level between the input and the reconstructed image is used as the loss.

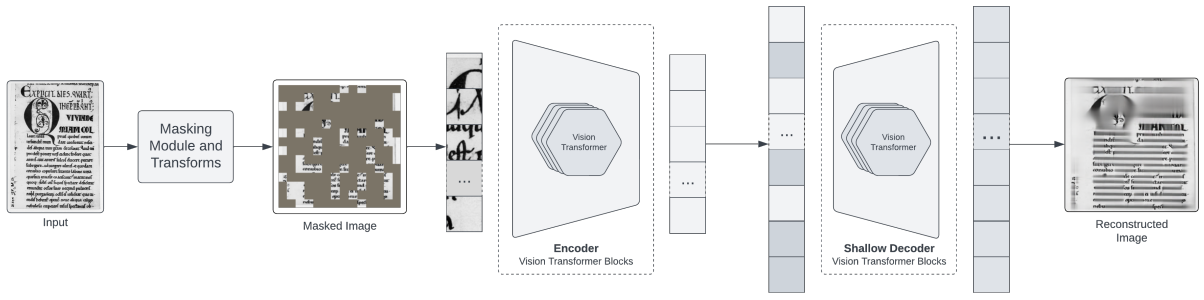


Figure 2.2: MAE Architecture

2.3.3 Bootstrap Your Own Latent

BYOL uses two neural networks, referred to as the online and target networks, which work together to learn from one another. Both the networks use a ResNet-50 architecture as the backbone and a multi-layer perceptron as the projection network. Additionally, a cosine based similarity loss between the online and target networks plays a vital role in effective learning [Gri+20]. The augmentations for images in the online and target networks are the same as in SimCLR.

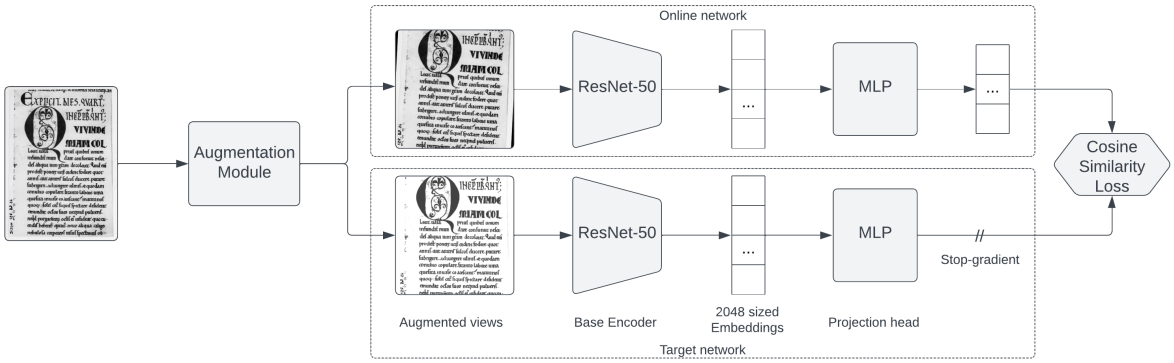


Figure 2.3: BYOL Architecture

Chapter 3

Experimental setup and results

The three self-supervised learning techniques mentioned were used in this study based on recommendations from the relevant research papers and with some customization. After training and evaluating these models using a linear evaluation protocol, it is concluded that the SimCLR model outperforms the other methods mentioned given the setup. The following sections describe the model hyperparameter customizations, image transformations used, and a comparative analysis of the results.

3.1 Model Configuration and Customizations

The SimCLR model includes erosion, dilation and random erasing in addition to transformations from Ying Chen et al. [Che+20]. Each transform was applied randomly with a probability of 0.5. A kernel size of 5 is utilized for dilation while a kernel size of 3 is employed for erosion. An identical approach was used in BYOL for preprocessing. The model configuration and hyperparameters passed for all the models are defined in Table 3.1.

Table 3.1: Pre-training configuration.

SimCLR		MAE		BYOL	
Variable	Value	Variable	Value	Variable	Value
Optimizer	Adam	Optimizer	AdamW	Optimizer	Adam
Base learning rate	1.5e-3	Base learning rate	3e-4	Base learning rate	3e-1
Weight decay	1e-6	Optimizer momentum	$\beta_1 = 0.9, \beta_2 = 0.95$	Weight decay	1.5e-6
Batch size	256	Weight decay	5e-2	Target decay rate	0.996
Scheduler	LambdaLR	Scheduler	Half-cycle cosine	Batch size	64
Scheduler Function	Linear warmup decay	Batch size	512	Scheduler	Cosine annealing
Warmup Epochs	10	Warmup Epochs	10	Warmup Epochs	10
Loss function	NT-Xent	Loss function	MSE	Loss function	Cosine similarity
Temperature	0.1	Backbone	Vision Transformer	Backbone	ResNet-50
Backbone	ResNet-50				

Evaluation of multiple ViT block sizes for backbone architecture in MAE revealed that ViT-Large demonstrated superior performance in terms of learning and numerical

stability. Whereas, both ViT-Base and ViT-Huge consistently resulted in NaN loss, while ViT-Tiny proved to be insufficient to learn features. Whereas, the added Erosion and Dilation augmentations to SimCLR were not performing well. Furthermore, extensive experimentation revealed that a large batch size of 512 and a learning rate scheduler are crucial for attaining good training results.

3.2 Pre-training progress and monitoring

The self-supervised methods in this study are pre-trained on the ICDAR-CLaMM dataset, which encompasses 12 classes for script type classification. To ensure consistency and reproducibility, a deterministic train, validation, and test split is performed, with a ratio of 70:10:20 percentages and a seed of 42. The test set remains untouched and is reserved for subsequent evaluation.

The models are pre-trained for a total of 500 epochs. The choice of a larger number of epochs is driven by the fact that the models are trained from scratch. This extended training duration allows the models to learn meaningful representations and capture intricate patterns present in the dataset.

By utilizing the powerful computational capabilities of GPUs and parallel data loading to accelerate the training process, resulting in the convergence of the models over 500 epochs. Over 150 experiments were conducted with varying configurations to find optimal parameters for convergence.

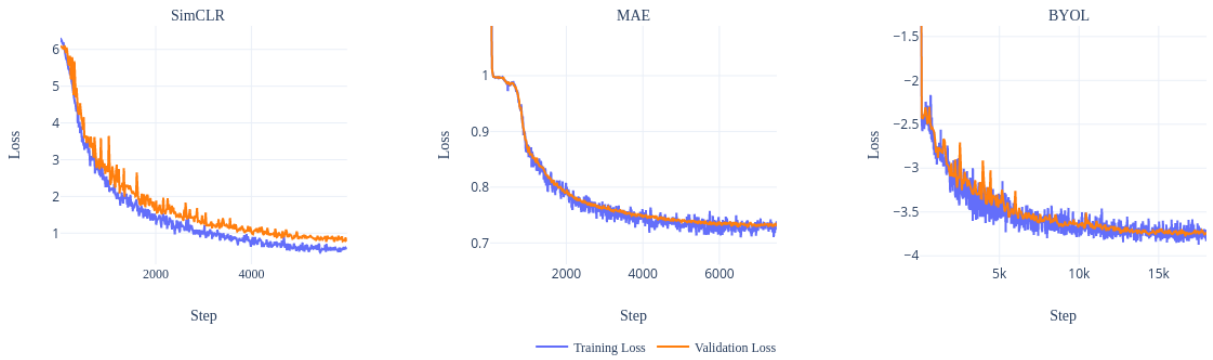
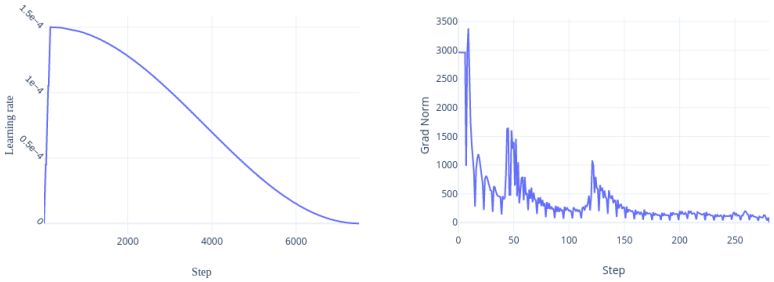


Figure 3.1: Loss curves for pre-trained models with batch size of 256, 256 and 64 from left to right.

Throughout the pretraining process, periodic evaluations are conducted to monitor the models' progress. Metrics such as training loss, validation loss, and grad norm are



(a) Learning rate (b) Normalized gradient

Figure 3.2: Metrics for MAE (a) Half-cycle cosine learning rate decay after warmup. (b) Normalized gradient after each step.

measured to assess the performance of the models at different stages of training. The loss curves for the best performing model generated during pre-training are shown in the Figure 3.1. These metrics played a crucial role in identifying and mitigating NaN loss caused by the issues of vanishing and exploding gradients.

3.3 Performance Evaluation

In order to assess the effectiveness of the learned representations through transfer learning, a downstream task of script type classification is performed using the pre-trained models. This evaluation serves to gauge the transferability and generalization capabilities of the representations acquired during the self-supervised pretraining phase.

3.3.1 Evaluation Metric

To assess the quality and transferability of the learned representations, a linear evaluation protocol is employed. This evaluation protocol serves as a standardized framework for evaluating the performance of the pre-trained models on a downstream task [Koc+22]. During the linear evaluation, the accuracy metric is employed to measure the performance of the linear classifier on the script type classification task. These metrics provide a quantitative assessment of the models' ability to generalize and make accurate predictions in the specific task context.

In addition, to acquire insights into the class distribution and the quality of the learned embeddings, a visualization technique based on t-distributed Stochastic Neighbor Embedding (t-SNE) is employed. t-SNE provides a powerful visualization method

for exploring high-dimensional data by reducing its dimensionality while preserving the underlying structure [Maa+08].

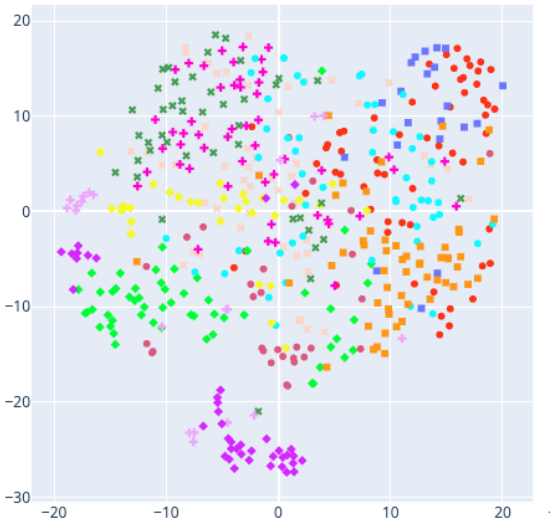


Figure 3.3: t-SNE visualization of embeddings with 12 classes for SimCLR

3.3.2 Quantitative Results

In this section, the results from the evaluations are presented and compared. The evaluation results for all the models are displayed in the Table 3.2. It is evident that, with a linear evaluation accuracy of 71.8% SimCLR outperforms MAE and BYOL.

Table 3.2: Evaluation results.

	Pre-training		Linear Evaluation	
Model Name	Epochs	Batch size	Training epochs	Top-1 accuracy
SimCLR	500	256	100	71.8%
MAE	500	256	100	36.1%
BYOL	500	64	100	45.2%

It is worth noting that MAE with a ViT backbone shows a lower accuracy of 36.1%. This observation can be attributed to its higher data requirements, as MAE with ViT demands a larger dataset for effective pre-training. The lower accuracy suggests that the available dataset size might not have been sufficient to fully leverage the potential of the MAE model with a ViT backbone. These findings highlight the importance of the choice of

model architecture and the impact it has on the performance of downstream tasks given the complexity of the dataset. SimCLR's success can be attributed to its ability to learn robust representations even with a comparatively smaller dataset.

3.3.3 Computational Efficiency

The models performance differs in terms of trainable parameters, pre-training time, checkpoint size for the same GPU acceleration. SimCLR has the fewest parameters and the smallest checkpoint size, while MAE has the shortest pre-training time and the largest parameters. This indicates that SimCLR is the least complex of all. Consideration of these factors can be crucial in selecting the most suitable model based on the available computational resources.

Furthermore, the implementation of effective learning techniques like gradient accumulation proved to be useful in training models with bigger batch sizes that exceed the memory capacity limitation of GPUs. Table 3.3 presents a comprehensive analysis of the metrics that impact computational efficiency, allowing for easy comparison.

Table 3.3: Comparison of computational efficiency metrics for models trained for 500 epochs.

Model Name	Trainable Params (millions)	Pre-training Time (hours)	GPU	Checkpoint Size (MB)
SimCLR	30	4.3	1 x A100	329
MAE	329	2.61	1 x A100	3700
BYOL	68	5.06	1 x A100	528

Chapter 4

Conclusion

Self-supervised image representation learning using deep convolutional neural networks and transformers for unlabelled data has shown great success. This project has extensively experimented on reviewing some of the state-of-the-art Self-supervised learning techniques on the ICDAR CLaMM dataset. Under the linear evaluation protocol on the dataset, SimCLR performs the best among MAE and BYOL, while using comparatively fewer parameters. In order to enhance accuracies, it is imperative to conduct further research and experimentation to delve into the intricacies of MAE and BYOL methodologies. Nevertheless, this study will be the foundation for implementing Self-supervised techniques for learning good representations in ancient handwritten script data.

List of Abbreviations

ICDAR International Conference on Document Analysis

CLaMM Classification of Latin Medieval Manuscripts

SSL Self-supervised Learning

SimCLR Simple Framework for Contrastive Learning of Visual Representations

MAE Masked Autoencoders

BYOL Bootstrap Your Own Latent

ViT Vision Transformer

t-SNE t-distributed Stochastic Neighbor Embedding

GPU Graphics Processing Unit

SLURM Slurm Workload Manager

NaN Not a Number

List of Figures

List of Tables

Bibliography

[Che+20] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A Simple Framework for Contrastive Learning of Visual Representations, June 2020. URL: http://arxiv.org/abs/2002.05709 (visited on 06/12/2023). arXiv:2002.05709 [cs, stat] (cited on pp. 2, 4, 7).

[Clo+17] F. Cloppet, V. Eglin, M. Helias-Baron, C. Kieu, N. Vincent, and D. Stutzmann. ICDAR2017 Competition on the Classification of Medieval Handwritings in Latin Script. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1371–1376, November 2017. DOI: 10.1109/ICDAR.2017.224. ISSN: 2379-2140 (cited on pp. 2, 3).

[Eri+22] L. Ericsson, H. Gouk, C. C. Loy, and T. M. Hospedales. Self-Supervised Representation Learning: Introduction, advances, and challenges. *IEEE Signal Processing Magazine*, 39(3):42–62, May 2022. ISSN: 1558-0792. DOI: 10.1109/MSP.2021.3134634. Conference Name: IEEE Signal Processing Magazine (cited on p. 3).

[Fal19] W. e. a. Falcon. Pytorch lightning. https://github.com/PytorchLightning/pytorch-lightning, 2019 (cited on p. 3).

[Gri+20] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu koray, R. Munos, and M. Valko. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. URL: https://proceedings.neurips.cc//paper/2020/hash/f3ada80d5c4ee70142b17b8192b2958e-Abstract.html (visited on 06/13/2023) (cited on pp. 2, 6).

[He+21] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked Autoencoders Are Scalable Vision Learners, December 2021. DOI: 10.48550/arXiv.2111.06377. URL: http://arxiv.org/abs/2111.06377 (visited on 06/17/2023). arXiv:2111.06377 [cs] (cited on pp. 2, 5).

[ICD17] ICDAR. ICDAR2017-CLaMM. fr-FR, 2017. URL: https://clamm.irht.cnrs.fr/icdar-2017/icdar2017-clamm/ (visited on 05/29/2023) (cited on p. 3).

[Koc+22] V. Kocaman, O. M. Shir, T. Bäck, and A. N. Belbachir. Saliency Can Be All You Need in Contrastive Self-supervised Learning. en. In G. Bebis, B. Li, A. Yao, Y. Liu, Y. Duan, M. Lau, R. Khadka, A. Crisan, and R. Chang, editors, *Advances in Visual Computing*, Lecture Notes in Computer Science, pages 119–140, Cham. Springer Nature Switzerland, 2022. ISBN: 978-3-031-20716-7. DOI: 10.1007/978-3-031-20716-7_10 (cited on p. 9).

[Maa+08] L. v. d. Maaten and G. Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL: http://jmlr.org/papers/v9/vandermaaten08a.html (cited on p. 10).

[Naj+15] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic. Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1):1, February 2015. ISSN: 2196-1115. DOI: 10.1186/s40537-014-0007-7. URL: https://doi.org/10.1186/s40537-014-0007-7 (visited on 07/12/2023) (cited on p. 2).

[Yad19] O. Yadan. Hydra - a framework for elegantly configuring complex applications. Github, 2019. URL: https://github.com/facebookresearch/hydra (cited on p. 3).