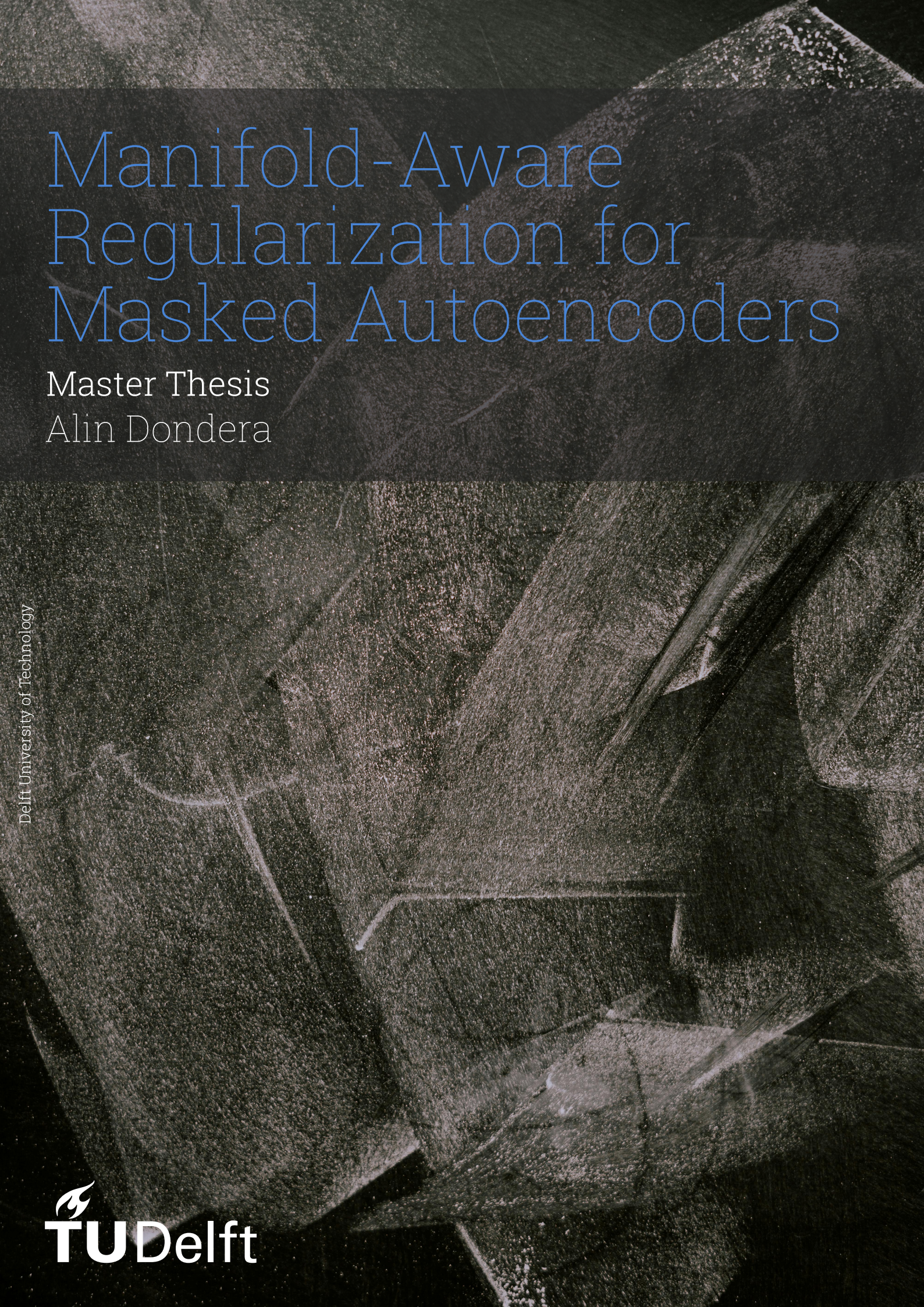
Manifold-Aware Regularization for Masked Autoencoders

Master Thesis

Alin Dondera



Delft University of Technology

TU Delft

Manifold-Aware Regularization for Masked Autoencoders

by

Alin Dondera

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday June 24, 2024.

Student number:	4934245	
Project duration:	April, 2023 – June 24, 2024	
Thesis committee:	Dr. ir. H. Jamali-Rad,	TU Delft and Shell, Daily Supervisor
	Dr. ir. J.C. van Gemert,	TU Delft, Advisor
	Dr. ir. G. Migut,	TU Delft, External Committee Member

TU Delft

Preface

This thesis marks the culmination of my six-year spell at TU Delft. It has been a privilege to work alongside and learn from brilliant people, and I hope that the knowledge I gained can be seen throughout this report. I am optimistic about the potential of this work and hope it can lead to a positive impact in the field of self-supervised learning. This research was conducted within the Computer Vision Lab at TU Delft, under the supervision of Dr. J.C. van Gemert and the daily guidance of Dr. H. Jamali-Rad.

First things first, I want to extend my gratitude to Dr. Jamali-Rad for just about everything he has done throughout this thesis. Your guidance, unwavering support, and the considerable amount of time you invested in this work have been invaluable. I've learned a lot from you. I also want to acknowledge the work of Anuj Singh, whose significant contributions to this research cannot be overstated. Our discussions were always a source of inspiration and clarity, providing a clear path forward at every turn. To Dr. J.C. van Gemert, although our interactions were less frequent, each conversation left me with new insights and perspectives. Thank you for that. Your guidance and expertise have been instrumental. I also want to thank Dr. Gosia Migut for her time, interest, and willingness to participate in my committee on such short notice. Your valuable feedback and expertise are greatly appreciated.

Finally, and most importantly, I want to extend a heartfelt thank you to my more-than-amazing family and friends. This would not have been possible without your unconditional support. To my parents, now that this chapter is closed, I promise I will use my newly freed-up schedule to call more often.

Alin Dondera
Delft, June 2024

Contents

1

Introduction

It is safe to say that the field of Artificial Intelligence (AI) has achieved new heights in recent years. Across a multitude of fields, from the ever-increasing family of Large Language Models (LLM) in Natural Language Processing (NLP) to Computer Vision breakthroughs enabling self-driving cars and creative image generation, these advancements rely upon the ability of AI systems to learn meaningful and effective data representations. This capacity to extract patterns from raw data, whether it is text, images, or sensor readings, lies at the heart of today's AI systems.

Traditionally, these representations were mostly learned via supervised learning. This means that a model would learn from datasets carefully annotated by humans for specific downstream tasks. A seminal example of such a dataset is ImageNet [11], a massive collection of images meticulously labeled with their corresponding object categories. While effective, this approach suffers from an important limitation: the expensive and time-consuming nature of the annotation process. The need for extensive human labeling can create bottlenecks in AI development, limiting the scalability and adaptability of models to new domains. Additionally, supervised learning can introduce biases inherent in the annotation process, potentially leading to discriminatory or inaccurate outcomes. Last but not least, this process of learning from a vast amount of labeled data drastically differs from how humans learn. A child does not need to see thousands of meticulously labeled cats and dogs to distinguish them. Instead, through exploration, interaction, and limited guidance, they develop a deeper understanding of the world. This ability to learn from (mostly) unlabeled data and build rich representations with minimal supervision is where self-supervised learning outperforms its supervised counterpart.

Self-supervised learning eliminates the need for manual annotations by relying solely on unlabeled data for training a machine learning system. The system derives supervisory signals from the unlabeled data using so-called "pretext tasks" [13]. This method has gained significant traction in the field of NLP. A prime example is one of the LLMs, called BERT [12]. The pretext task in this case is defined as masking a part of a sequence of words, then asking the model to predict the missing words. The effectiveness of today's LLM models is in large part thanks to this kind of self-supervised pretraining on vast amounts of unlabeled text data.

Advancements in self-supervised learning have also quickly expanded into the domain of image representation learning. Here, models learn by solving carefully designed pretext tasks to exploit the intrinsic structure of image data. While initial approaches artificially design their pretext tasks (e.g. predicting rotation [16], patch ordering [13] or jigsaw puzzles [26]), the latest approaches focus on discriminating between different instances of images obtained through various augmentations. One example is contrastive learning (such as with SimCLR [6] and MoCo [21]). Here, the goal is to learn representations that bring similar examples (e.g., different augmentations of the same image) closer together in the representation space while pushing dissimilar examples apart. Other techniques include knowledge distillation (like in BYOL [18] and DINO [4]), where a "teacher" network learns from self-supervised signals and guides the learning of a 'student' network, and clustering-based methods (e.g., SwAV [5]), where models learn to group similar image features by predicting cluster assignments. Finally, InfoMax

methods like BarlowTwins [32] and VICReg [1] focus on maximizing the mutual information between different views of an image, but with an emphasis on reducing redundancy within the learned representations. Discriminative methods are not without limitation. These methods can be sensitive to the choice of data augmentations, prone to collapsing representations, and sometimes overemphasize low-level visual features rather than capturing higher-level semantic understanding.

Generative methods, on the other hand, specifically Masked Image Modelling (MIM) methods, with Masked Autoencoders (MAE) [20] as a prime example, do not require any augmentations and use the same framework for SSL in Computer Vision as in NLP. Currently, most state-of-the-art approaches use MIM in one way or another in their pretraining. This type of approach is not without blame however, as they can struggle with capturing the underlying manifold structure of the data, leading to suboptimal representations and limiting their generalization capabilities.

Manifold-aware regularization techniques aim to address this limitation by explicitly incorporating the geometric structure of the data manifold into the learning process. These techniques encourage the model to learn representations that are faithful to the intrinsic geometry of the data, thereby improving the quality of the learned representations and enhancing their generalization ability to unseen data.

This thesis is organized as follows: We begin by introducing `MAGMA`, a novel manifold-aware regularization for MAEs, which has been submitted to the “18th European Conference on Computer Vision (ECCV 2024)”. We then delve into the background knowledge necessary to fully appreciate our contribution, including an overview of deep learning, self-supervised learning, and manifold regularization. Finally, we conclude by summarizing our findings and highlighting potential future avenues of research.

2

Scientific Paper (MAGMA)

MAGMA: Manifold Regularization for MAEs

Alin Dondera[1], Anuj Singhe[1,2] and Hadi Jamali-Rad[1,2]

[1]Computer Vision Lab, Delft University of Technology, The Netherlands
[2]Shell Global Solutions International B.V., Amsterdam, The Netherlands

Abstract

Masked Autoencoders (MAEs) represent a significant shift in self-supervised learning (SSL) due to their independence from augmentation techniques for generating positive (and/or negative) pairs as in contrastive frameworks. Their masking and reconstruction strategy also aligns well with SSL approaches in natural language processing. Most MAEs are built upon Transformer-based architectures where visual features are not regularized as opposed to their convolutional neural network (CNN) based counterparts, which can potentially limit their effectiveness. To address this, we introduce MAGMA, a novel batch-wide layer-wise regularization loss applied to representations of different Transformer layers. We demonstrate that by plugging in the proposed regularization loss, one can significantly improve the performance of MAE-based baselines. We further demonstrate the impact of the proposed loss on optimizing other generic SSL approaches (such as VICReg and SimCLR), broadening the impact of the proposed approach. Our code base is available and can be accessed here.

1 Introduction

Self-supervised learning has made significant progress over the recent years by producing results on par with supervised baselines (Bardes et al., 2021; Grill et al., 2020; Zbontar et al., 2021; Chen et al., 2020; Caron et al., 2020; 2021), thus rendering it as a promising paradigm for learning representations without access to labels. Many notable approaches in self-supervised learning such as contrastive learning (Chen et al., 2020), clustering-based methods (Caron et al., 2020), redundancy minimization (Bardes et al., 2021; Zbontar et al., 2021) and distillation-based methods (Grill et al., 2020) aim to learn representations that generalize well by avoiding degenerate solutions and representational collapse by utilising a joint embedding architecture to enforce consistency between representations of different image-views. Inspired by natural language processing (NLP), Masked Autoencoders (MAE) approach the task of self-supervised pre-training by a conceptually simple idea of masking a portion of the input data to then learn to predict the removed content. Specifically, this is applied to images by masking a very large portion (eg. 75%) of their content by replacing it with random patches. This creates a challenging pretext task for image representation learning that requires the neural network to develop a holistic understanding beyond low-level image statistics (He et al., 2022). By masking a large part of the image and processing only the unmasked region, MAEs provide a computationally efficient way of pre-training large-scale vision transformers such as ViT-B/H/S (He et al., 2022; Dosovitskiy et al., 2020). However, due to the lack of an objective that optimizes for contrasting negative pairs of images, the features learnt by MAE pre-training require large amounts of labeled data to be fine-tuned for satisfactory downstream task performance (Lehner et al., 2023). Moreover, deep architectures such as convolutional neural networks are designed with inherent regularization characteristics such as translation invariance, equivariance, and parameter sharing that are relevant to learning information-rich features from images for multiple vision-oriented tasks. On the other hand, ViT-based architectures operate on patches of images and lack these aforementioned regularization characteristics in their feature extraction process. In an ideal scenario, a well-trained network should exhibit a crucial property: if two similar inputs are fed into the network, their resulting outputs should also be close together. This principle ensures that the network learns robust representations that capture the underlying structure of the data, not just random noise or specific details. Deviations from this principle can indicate the network is overly sensitive to small input variations, leading to poor generalization performance on unseen data. One way to enforce this behavior is through manifold regularization, which aims to guide the model toward learning smoother representations aligned

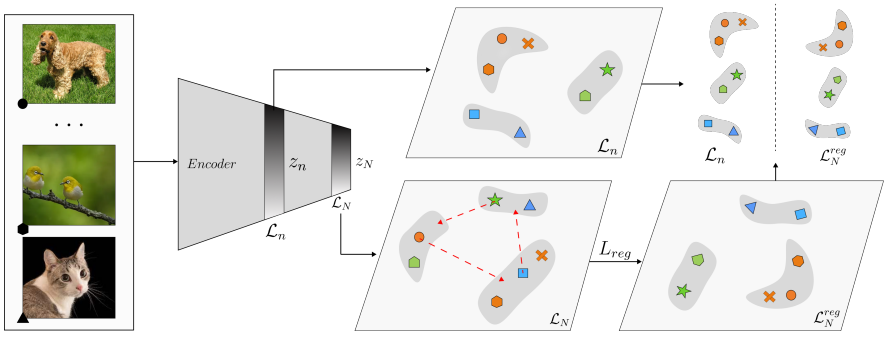


Figure 1: Visualization for the proposed regularization loss `MAGMA` with MAE: `MAGMA` penalizes representations that are close in the latent space of intermediate layer k but far apart in layer l latent space. This induces a regularization effect across different layers that preserves inter-sample and intra-batch relationships thus enforcing consistency in the latent representation space. Note that we demonstrate this for MAE based pre-training with a transformer encoder-decoder architecture such as ViT.

with the intrinsic data geometry (Belkin et al., 2006). To this end, we introduce `MAGMA`, a novel batch-wide loss that regularizes representations across multiple different layers of a feature extractor. Our extensive experiments, ablations, and analyses empirically demonstrate improved downstream image classification performance on MAE-based baselines by simply plugging in the proposed regularization loss during the pre-training phase. To corroborate the general applicability and broader impact of `MAGMA`, we demonstrate improved and on-par performance of other generic SSL approaches such as VICReg (Bardes et al., 2021) and SimCLR (Chen et al., 2020) when pre-trained with our proposed loss.

2 Related Work

Self-supervised learning. Self-supervised learning is crucial for overcoming the limitations of traditional supervised learning, which requires vast amounts of expensive, hand-labeled data. By automatically generating labels from the data itself, self-supervised techniques enable models to learn meaningful representations from unlabeled data, reducing our reliance on manual annotation.

Masked Autoencoders. The success of self-supervised learning in Natural Language Processing (NLP), particularly with masked language modeling techniques in models like BERT (Devlin et al., 2018) has inspired analogous developments within computer vision. Masked Autoencoders (MAEs) (He et al., 2022) take the idea of masking and apply it to an autoencoder structure with a pixel-level reconstruction loss. This results in impressive performance across various downstream tasks (Pang et al., 2022; Zhang et al., 2022; Chen et al., 2023; Zhou et al., 2022). Other similar works include BEiT (Bao et al., 2021), SimMIM (Xie et al., 2022), and iBOT (Zhou et al., 2021), with close connections to contrastive learning (Kong et al., 2019; Zhang et al., 2022).

Manifold regularization. At the core of `MAGMA` lies the seminal piece of work of Belkin et al. (2006). The authors provide a geometrically intuitive and novel semi-supervised learning framework that leverages the underlying geometry of data distributions under the assumption that two points close together on the manifold (i.e., similar in the true underlying structure of the data), should have their corresponding target outputs also be similar. This idea has been successfully applied in deep learning across of variety of tasks, such as speech recognition (Tomar and Rose, 2014; 2016), NLP (Yonghe et al., 2019; Li et al., 2021) and vision (Jie et al., 2015; Jin and Rinard, 2020; Shaham et al., 2018; Hu et al., 2018), showcasing its usefulness in the general setting. `MAGMA` extends the concept of manifold regularization to the self-supervised setting, guiding internal network transformations to promote smoother, more generalizable representations. While Shaham et al. (2018) explores a similar direction, their approach relies on Siamese networks to explicitly calculate similarity measures between input images. In contrast, our regularization operates directly on the representations generated within the network, offering a more tightly integrated self-supervised mechanism.

3 Method: MAGMA

Given an unlabeled dataset $\mathcal{D}_u$ with samples $x \in \mathcal{D}_u$ our goal is to train an encoder f_θ with L layers to produce information-rich representations in a self-supervised fashion. During inference, the parameters of the encoder are frozen θ and a linear layer is trained in a supervised fashion. This procedure is known as linear probing and is the commonly adopted setup in self-supervised learning (SSL) literature. We denote a batch of B samples as $\mathcal{B}$. In this setting, our goal is to apply a batch-level regularization loss in a layer-wise fashion on a set of layers $\mathcal{K} \subseteq [L]$:

$$\mathcal{L}(\mathcal{B}, \mathcal{K}; \theta) = \mathcal{L}_{SSL}(\mathcal{B}; \theta) + \lambda \mathcal{L}_{Reg}(\mathcal{B}, \mathcal{K}; \theta), \tag{1}$$

where the first term denotes a standard self-supervised learning loss, and λ is a weighting parameter between the two terms. While any set of layers can in practice be adopted for such a regularization, we demonstrate later on that applying this on an intermediate and the last layer $\mathcal{K} = \{l, L\}$ would yield the maximum impact. Notably, this is applied only at the pretraining phase.

3.1 Batch-Wide Layer-Wise Manifold Regularization

We denote the representation output of layer $l \in [L]$ of f_θ for input image $i \in \mathcal{B}$ as $Z_i^{(l)}$. Inspired by Belkin et al. (2006), we propose to apply the following batch-wide layer-wise regularization term to enforce consistency among the output representations of the selected layers:

$$\mathcal{L}_{\text{Reg}}(\mathcal{B}, \mathcal{K}; \theta) = \frac{1}{B^2} \sum_{k,l \in \mathcal{K}} \sum_{i,j \in [B]} w(Z_i^{(k)}, Z_j^{(k)}) \cdot ||Z_i^{(l)} - Z_j^{(l)}||^2 \tag{2}$$

$$\mathcal{L}_{\text{Reg}^*}(\mathcal{B}, \mathcal{K}; \theta) = \frac{1}{B^2} \sum_{k,l \in \mathcal{K}} \sum_{i,j \in [B]} w^*(Z_i^{(k)}, Z_j^{(k)}) \cdot ||Z_i^{(l)} - Z_j^{(l)}||^{-2}, \tag{3}$$

where $w(.) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ can be any similarity kernel, with D being the size of the vectorized version of Z. In our study, we employ the Radial Basis Function (RBF) kernel due to its favorable properties as discussed in (Belkin et al., 2006). Thus, we have:

$$w(Z_i^{(k)}, Z_j^{(k)}) = \exp\left(\frac{-||Z_i^{(k)} - Z_j^{(k)}||^2}{2\sigma}\right) \tag{4}$$

$$w^*(Z_i^{(k)}, Z_j^{(k)}) = \exp\left(\frac{||Z_i^{(k)} - Z_j^{(k)}||^2}{2\sigma}\right), \tag{5}$$

where σ is a free parameter. We choose $\sigma^2 = \texttt{var}(d_{ij})$, with $d_{ij} = ||Z_i^{(k)} - Z_j^{(k)}||^2$, following the approach in Rodríguez et al. (2020) for enhanced training stability. Dynamically adjustment of σ this way ensures our regularization adapts to the spread of features inside a batch: the more spread out the features are (i.e. higher σ) the wider the influence of the RBF kernel. Conversely, a lower spread would result in a more focused kernel (focusing on finer, more local distinctions). Note that in Eq. 2, layer k is considered as the reference layer and layer l is regularized accordingly. More concretely, if two instances (Z_i and Z_j) have closer representations in the manifold space of layer k (leading to higher $w(Z_i^{(k)}, Z_j^{(k)})$), but are far apart in the manifold space of layer l, $\mathcal{L}_{\text{Reg}}$ would heavily penalize them, as a result pulling them closer in the regularized manifold. We illustrate later on that in practice this would not only regularize layer l but also all the previous layers.

The regularization loss in Eq. 2 can be reformulated in terms of the Laplacian matrix L determined by all pairs of instances $(Z_i^{(k)}, Z_j^{(k)})$ in a batch, and is defined as follows:

$$\mathcal{L}_{\text{Reg}}(\mathcal{B}, \mathcal{K}; \theta) = \frac{1}{B^2} \texttt{Trace}(Z^{(l)T} L Z^{(l)}), \tag{6}$$

We make use of the normalized Laplacian for better stability during training, defined as follows:

$$L = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}, \quad D_{ii} = \sum_j W_{ij} = \sum_j w(Z_i^{(k)}, Z_j^{(k)}), \tag{7}$$

Application to Transformers. For the sake of generality, we have so far formulated the problem so that it would be readily applied to any layered neural network architecture. Even though, we have only observed significant impact on ViT based architectures. The only difference for ViT based architectures is that per layer l we would have P patches each of which returning a representation $Z_{i,p}^{(l)}$, $\forall p \in [P]$, where the image level representation would simply be the average of all those representations $Z_i^{(l)} = \sum_p Z_{i,p}^{(l)}$. The reason behind this averaging strategy is that applying the regularization over the representations of individual patches across different images is not ideal due to patch noise and lack of global context. This may result in irrelevant computations since similar patches within an image already share context through the self-attention mechanism.

4 Impact of Architectures and Pretraining Methods

The proposed regularization can be seamlessly incorporated into various self-supervised methods, with the caveat that the chosen architecture and pretraining approach play an important role in determining the efficacy of the regularization. The inherent characteristics of CNN-based architectures can diminish the impact of regularization. For instance parameter sharing, translation invariance and equivariance in CNNs, which facilitates the reuse of learned features across various input regions, can result in reduced regularization impact. In contrast, Transformers lack these specific characteristics, potentially making them more suitable for this regularization.

The nature of the pretraining method significantly influences the impact of regularization. Contrastive methods (e.g., SimCLR, MoCo), clustering-based approaches (SWaV), distillation-based techniques (DINO, BYOL), and InfoMax/Dimension Contrastive methods all aim to bring representations of augmented views of the same image closer together, essentially performing a task related to our proposed regularization. Therefore, the proposed regularization will have a diminished impact on these methods. On the other hand, Masked Autoencoders (MAE)'s exhibits a generative nature, by randomly masking large portions of an image and reconstructing the missing pixels. Since this process is applied individually, it is also not sharing any information between representations within a batch. These characteristics make it better suited for the regularization term. As a result, our study will primarily focus on MAEs as they align well with the objectives of our proposed regularization approach.

5 Experiments

Our goal in this section is to evaluate the impact of adding our regularization term on top of pre-existing SSL methods, both quantitatively and qualitatively. We aim at addressing the following questions:
[Q1] How does $\mathcal{L}_{\text{Reg}}$ influence downstream image classification?
[Q2] What is the effect of $\mathcal{L}_{\text{Reg}}$ on the training dynamics?
[Q3] What are the important hyperparameters of the proposed regularization?
[Q4] Is the impact of $\mathcal{L}_{\text{Reg}}$ on representations qualitatively noticeable?

Benchmark Datasets. We evaluate our proposed regularization on commonly adopted datasets for the downstream task of image classification, namely, CIFAR100 (Krizhevsky et al., 2009), STL-10 (Coates et al., 2011), Tiny-Imagenet (Le and Yang, 2015), and Imagenet-100 (Tian et al., 2020). This selection of datasets provides various challenges in terms of data resolution, number of classes, and overall complexity of context presented in the sample image. By testing across diverse datasets, we aim to showcase the robustness and generalizability of our proposed regularization.

Baseline methods. We evaluate the efficacy of our proposed regularization on several SSL methods (to demonstrate its versatility), with an emphasis on MAE for the reasons discussed in Section 4. This includes U-MAE (Zhang et al., 2022), an improvement over the baseline MAE addressing dimensional collapse with

Table 1: Linear probing accuracy and k-nn accuracy (k=10) of models pretrained and evaluated on the given datasets. Adding our proposed regularization term to the baseline method generally increases performance.

	CIFAR-100		STL-10		Tiny-Imagenet		Imagenet-100	
Method	linear	knn	linear	knn	linear	knn	linear	knn
MAE	38.2	36.6	66.5	62.0	17.8	17.7	58.0	47.5
M-MAE (ours)	**43.3**	**40.7**	**71.0**	**65.9**	**20.9**	**20.5**	**69.0**	**49.8**
U-MAE	45.3	45.9	74.9	72.1	21.5	19.0	69.5	56.8
MU-MAE (ours)	**46.4**	**46.4**	**75.6**	**73.0**	**25.2**	**23.9**	**73.4**	**60.1**
SimCLR	62.8	58.7	**90.4**	**86.9**	**50.9**	43.5	67.8	65.3
M-SimCLR (ours)	**63.2**	**59.4**	**90.5**	**86.9**	**51.0**	**44.6**	**68.7**	**65.6**
VICReg	63.6	60.8	**87.4**	**84.5**	45.2	**40.5**	68.4	62.1
M-VICReg (ours)	**64.7**	**61.9**	**87.4**	**84.5**	**45.8**	**40.5**	**70.4**	**65.1**

an additional regularization term. Additionally, we investigate the impact on two other widely adopted SSL baselines: SimCLR (Chen et al., 2020) and VICReg (Bardes et al., 2021).

Implementation details. We focus on the impact of regularization by keeping the architectural and hyperparameter choice intact throughout the experimentation, except for the ablation studies. For low(er)-resolution datasets (CIFAR100, STL-10, and Tiny-Imagenet) we use a ViT-Tiny backbone, while for Imagenet-100, we use ViT-Base. We select the best-performing hyperparameter setting for each baseline method and add our regularization on top of it. For our regularization, we tune three parameters: the regularization weight λ, the number of warmup epochs e_{st}, and the duration of the regularizer e_{dur}. More details on the choice of (hyper)parameters can be found in the supplementary material. Notably, for all the other baselines, we present the reproduction results in one optimized pipeline, which in almost all cases leads to performances over the originally reported results in Bardes et al. (2021); Chen et al. (2020); He et al. (2022); Zhang et al. (2022).

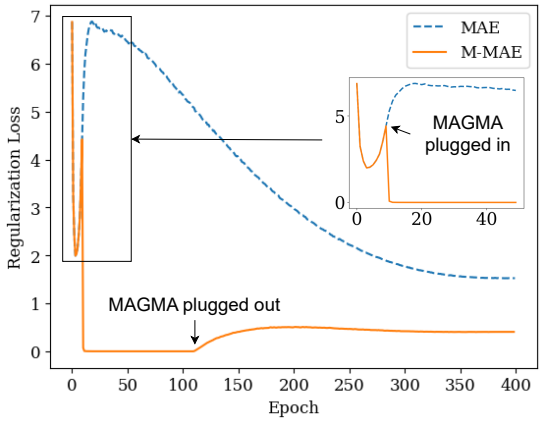
Evaluation protocol. For our main results, we follow the commonly adopted protocol in SSL, based on freezing the network encoder after the pretraining phase and training a linear layer on top of it in a supervised fashion. For all baselines, we train for 100 epochs using SGD, using a learning rate of 0.1 with decay at steps 60 and 80, and a batch size of 256. In addition, we also evaluate the k-nearest neighbours (kNN) classification accuracy using $k = 10$ and a Euclidean distance measure.

5.1 [AQ1] Comparison Against Other Baselines

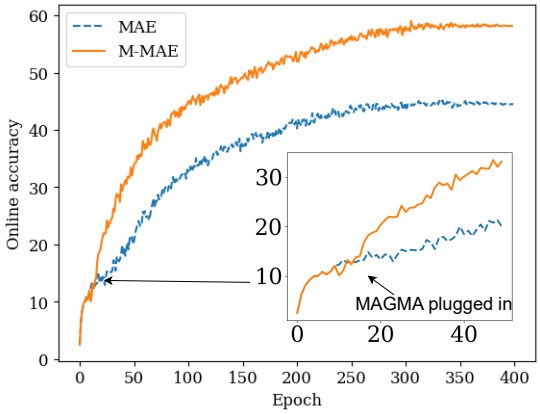
Table 1 summarizes the results of applying `MAGMA` on top of the aforementioned four baselines (MAE, U-MAE, SimCLR, and VICReg). We pretrain and evaluate on the same datasets to showcase the robustness of `MAGMA` over various pretraining scenarios. As can be seen, our proposed approach offers significant improvements across all four datasets by outperforming baseline MAE, both in the linear setting (+5.1% on CIFAR-100, +4.5% on STL10, +2.9% on Tiny-Imagenet and +11% on Imagenet-100), as well for kNN one (**+4.1**% on CIFAR-100, **+3.9**% on STL-10, **+2.8**% on Tiny-Imagenet, and **+2.3**% on ImageNet 100). For U-MAE, while the improvements are still significant, except for Tiny-Imagenet, they are smaller in magnitude. Going beyond MAEs, the results on SimCLR and VICReg, show some marginal improvement opening the door for further investigation and broader impact.

5.2 [AQ2] Training Dynamics

Figure 2a illustrates the value of the proposed loss term ($\mathcal{L}_{Reg}$) throughout training epochs. The dashed line illustrates the scenario in which $\mathcal{L}_{Reg}$ is evaluated but not backpropagated. This curve manifest signs of instability (lack of consistency) in the manifold space of representations (for selected layers 11 and 12). The solid curve shows the impact of backpropagating $\mathcal{L}_{Reg}$ (applying `MAGMA` at epoch 10) where a sudden change of behavior is apparent upon the introduction of $\mathcal{L}_{Reg}$ in the optimization. The fact that the $\mathcal{L}_{Reg}$ drops drastically instead of ascending (dashed line) after being introduced, together with the stability of the



(a) Regularization loss tracked throughout pretraining for MAE and M-MAE, on Imagenet-100



(b) Online accuracy tracked throughout the pretraining phase for MAE and M-MAE, on Imagenet-100.

Figure 2: (a) The regularization loss showed for MAE and M-MAE. For MAE we calculate the loss without backpropagating. For M-MAE, we apply the loss after 10 warmup epochs, and take it out after 100 epochs. (b) The online accuracy was obtained by training a linear layer on the representations produced by the encoder throughout pretraining. The accuracy slightly drops for M-MAE when the regularization kicks in but increases at a significantly higher rate compared to MAE.

loss after removal (at epoch 110), as well as the consistently better online accuracy of M-MAE as seen in Figure 2b, could potentially suggest that the optimization is now steered in a different direction, leading to an overall significantly better performance. Based on this, we hypothesize that the e_{st} parameter is best set around the point when the $\mathcal{L}_{\text{Reg}}$ loss would start increasing.

Which layers to regularize on? We have run extensive experimentation to effectively select the target layers for applying `MAGMA`. It turns out regularizing the last layer with respect to the penultimate layer seems to have the maximum impact, in ViT based architecture. In Figure 3, we demonstrate that choosing $k = 10$ (11-th later in ViT base architecture) as the reference and $l = 11$ (last year) not only leads to regularizing loss across the two layers, but also results in percolated impact through all the previous layers.

5.3 [AQ3] Ablations on Important (Hyper)Parameters.

We evaluate the impact of three pivotal regularization parameters (i) λ in Equation (1), (ii) the epoch at which `MAGMA`is applied, e_{st}, and (iii) the duration over which the regularization is applied before being plugged out, e_{dur}. The results for the first three parameters are summarized in Table 2.

The regularization weight λ directly controls the strength of the regularization effect in the overall optimization loss Equation (1). Intuitively, lower weight for $\mathcal{L}_{\text{Reg}}$ might not significantly impact the overall optimization, whereas higher weight could lead to an over-regularized optimization and a degraded performance. The results show a similar trend: lowering the weight to 0.1 leads to a performance similar to the baseline (+2%). Increasing the weight by a factor of 10 reduces the gain slightly by 1%. Interestingly, reducing the weight to 0.01 leads to lower downstream classification performance than the baseline. We hypothesize that this is because the regularization introduces a competing gradient signal which inadvertently hinders training performance.

The warm up period e_{st} allows the model to train for a few epochs without $\mathcal{L}_{\text{Reg}}$ to help it establish a reasonable foundation for learning basic representations. This could prevent the regularization from overly restricting the model too early in the training process. As can be seen from Table 2, a small number of epochs for e_{st} would already be enough for a maximal impact. Delaying this further seems to have an increasingly negative impact.

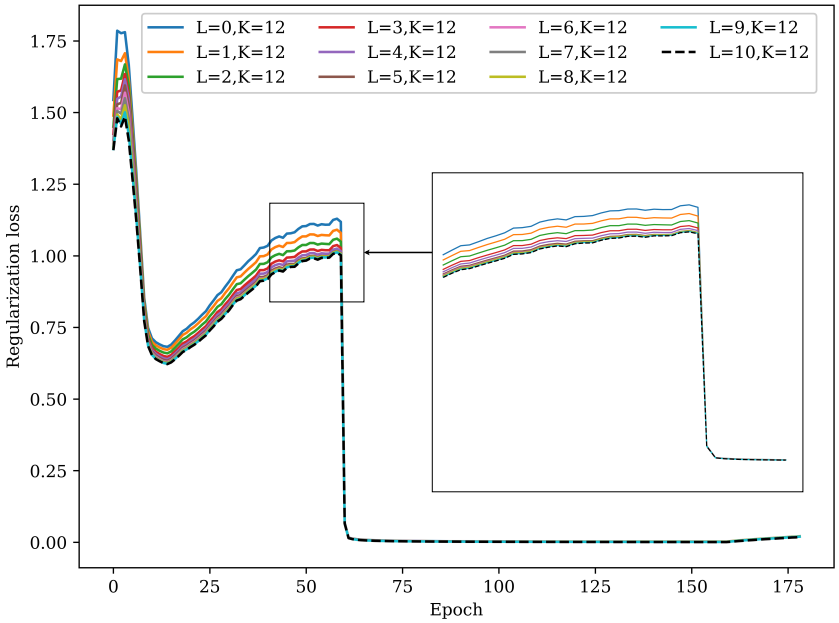


Figure 3: Effect of regularization. Implication: if the representations from any two layers are close, then the output representation will also be close.

Table 2: Linear accuracy performance using different choices of hyperparameters for regularization. Results are computed on ImageNet-100.

λ	*1*	*0.1*	*0.01*	*10*	1	1	1	1	1	1	1	1
e_{st}	10	10	10	10	*0*	*2*	*20*	*50*	10	10	10	10
e_{dur}	100	100	100	100	100	100	100	100	*10*	*50*	*200*	*390*
Accuracy	69.0	60.0	54.6	67.9	68.2	69.0	65.5	62.7	68.5	68.8	69.0	68.5

Lastly, the duration parameter e_{dur}, determines the amount of pressure put on the model to develop smooth and aligned representations across layers. We experiment with different values ranging from only 10 epochs, up until the end of training (i.e. a duration of 390 epochs). The results show that the impact of this parameter is less pronounced. There is a slight decrease in performance (by about 0.5%), for significantly lower or higher duration periods. It seems that applying `MAGMA` for a number of epochs already regularizes the representations across the network with a lingering impact from which point onward it can be plugged out without hampering the overall performance. As discussed earlier in Section 5.2, we hypothesize that this lingering impact is related to the adjusted optimization landscape as a result of applying the proposed regularization.

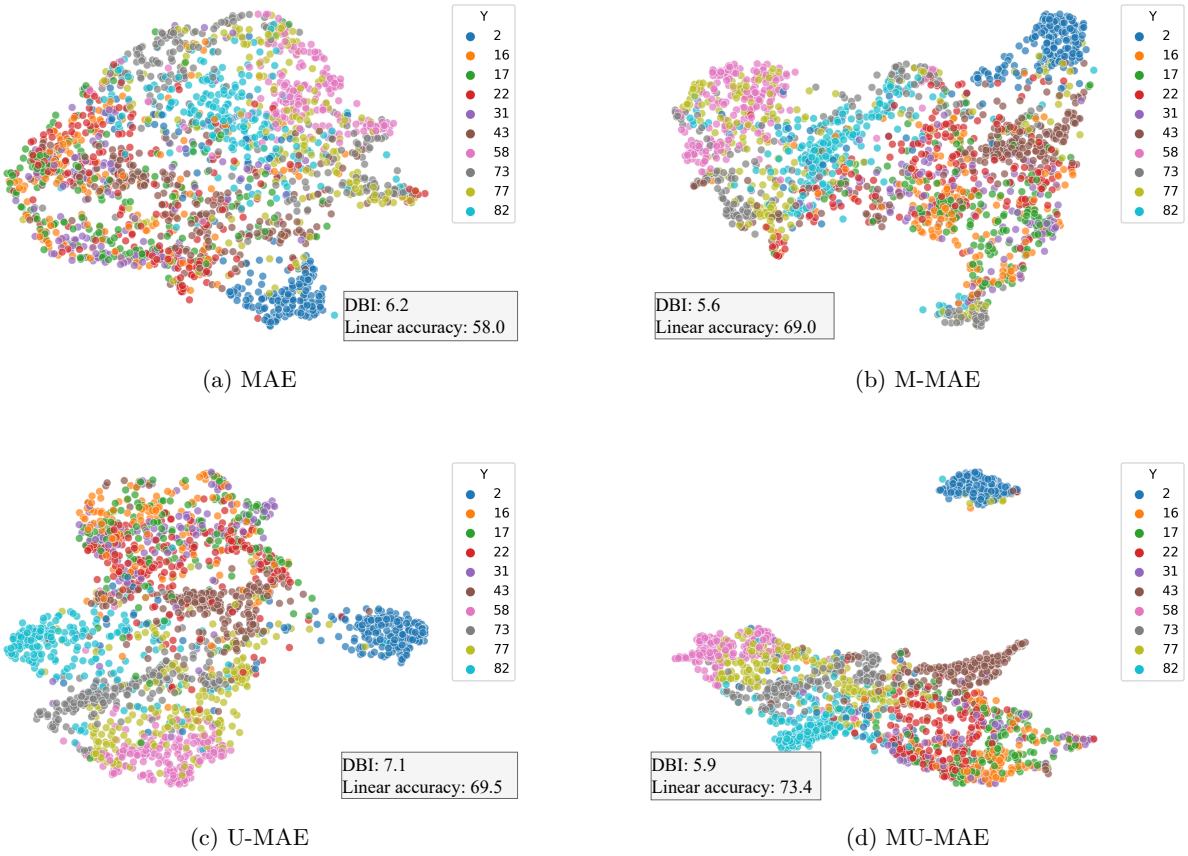
To further investigate the sensitivity of `MAGMA`, we evaluate the performance by changing the backbone architecture starting from small to larger (ViT-S to ViT-L). As can be seen in Table 3, increasing the capacity of the backbone results in considerable performance improvement in the baseline approaches (MAE and U-MAE) where the performance boosts decreases for changing the backbone from ViT-B to ViT-L. Interestingly, similarly significant boost can be observed on the `MAGMA` optimized baselines (M-MAE, MU-MAE), offering consistent improvement over the baselines.

5.4 [AQ4] Qualitative Analysis

To qualitatively assess the impact of our regularization, we visualize the representations of MAE, U-MAE, as well as their regularized version, M-MAE, and MU-MAE, on a random sample of 10 classes from Imagenet100. We use PacMAP (Wang et al., 2021) for dimensionality reduction. PaCMAP outperforms t-SNE (Van der

Table 3: Linear probing results using different backbones on Imagenet-100.

Method	ViT-S	ViT-B	ViT-L
MAE	46.8	57.9	60.6
M-MAE (ours)	**61.2**	**69.2**	**73.9**
U-MAE	57.6	69.5	**78.2**
MU-MAE (ours)	**62**	**73.4**	**78.4**



(a) MAE (b) M-MAE

(c) U-MAE (d) MU-MAE

Figure 4: PaCMAP plots for MAE-based methods. Applying `MAGMA` on top of U-MAE leads to compact and well-defined clusters.

Maaten and Hinton, 2008) and UMAP (McInnes et al., 2018) in preserving the global structure of high-dimensional data within visualizations. This means it more accurately reflects the large-scale relationships and patterns present in the original dataset. We include the linear accuracy, as well as the Davies-Bouldin Index (DBI) (Davies and Bouldin, 1979) alongside the visualizations. DBI is a common metric used to evaluate clustering algorithms, where lower DBI scores indicate better separation between clusters and tighter groupings within clusters.

Results are shown in Figure 4. We notice that after applying `MAGMA`, the visualized representations appear more compact as compared to the baseline MAE representations. This is quantitatively verified by the drop in DBI scores when applying `MAGMA` and MU-MAE.

6 Concluding remarks

We propose MAGMA a novel regularization technique that regularizes the representations and enforces consistency across different layers of a transformer-based MAE. We demonstrate the efficacy of the proposed approach through a suite of experimentations resulting in significant performance gain over MAE-based baselines.

Broader impact. As we have shown earlier, MAGMA can be rather straightforwardly applied to any kind of SSL approach irrespective of the backbone architecture. As we discuss in Section 3 this applies essentially to any layered deep neural networks, irrespective of operating on an encoder-decoder architecture. This can potentially broaden the application of the proposed approach to contexts even beyond computer vision. This is an avenue for future work

Limitations. Our initial results on Imagenet-1K (Deng et al., 2009) do not provide any notable performance above the baseline MAE. We hypothesize that this might be related to our computational constraints (of having 2 A100 GPUs) allowing us to only experiment with a batch size of maximum 256 per GPU. This is considerably smaller than the standard batch size reported in the literature (e.g. 4096 and above). Another limitation we observed when going beyond ViT-based architectures, to the CNN counterparts, is the considerably smaller impact of the proposed approach. We argue that standard operations in modern CNN-based architectures (such average pooling, weight sharing, etc.) might already serve as a regularize and minimize the impact of MAGMA. Both topics call for more extensive experimentation to further substantiate our understanding.

References

Amir, S., Gandelsman, Y., Bagon, S., and Dekel, T. (2021). Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2(3):4.

Bao, H., Dong, L., Piao, S., and Wei, F. (2021). Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*.

Bardes, A., Ponce, J., and LeCun, Y. (2021). Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*.

Belkin, M., Niyogi, P., and Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(11).

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Chen, Z., Agarwal, D., Aggarwal, K., Safta, W., Balan, M. M., and Brown, K. (2023). Masked image modeling advances 3d medical image analysis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1970–1980.

Coates, A., Ng, A., and Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings.

da Costa, V. G. T., Fini, E., Nabi, M., Sebe, N., and Ricci, E. (2022). solo-learn: A library of self-supervised methods for visual representation learning. *Journal of Machine Learning Research*, 23(56):1–6.

Davies, D. L. and Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. (2022). Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009.

Hu, H., Ma, B., Shen, J., Sun, H., Shao, L., and Porikli, F. (2018). Robust object tracking using manifold regularized convolutional neural networks. *IEEE Transactions on Multimedia*, 21(2):510–521.

Jie, B., Zhang, D., Cheng, B., Shen, D., and Initiative, A. D. N. (2015). Manifold regularized multitask feature learning for multimodality disease classification. *Human brain mapping*, 36(2):489–507.

Jin, C. and Rinard, M. (2020). Manifold regularization for locally stable deep neural networks. *arXiv preprint arXiv:2003.04286*.

Kong, L., d'Autume, C. d. M., Ling, W., Yu, L., Dai, Z., and Yogatama, D. (2019). A mutual information maximization perspective of language representation learning. *arXiv preprint arXiv:1910.08350*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Le, Y. and Yang, X. (2015). Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3.

Lehner, J., Alkin, B., Fürst, A., Rumetshofer, E., Miklautz, L., and Hochreiter, S. (2023). Contrastive tuning: A little help to make masked autoencoders forget. *arXiv preprint arXiv:2304.10520*.

Li, X., Wang, Y., Ouyang, J., and Wang, M. (2021). Topic extraction from extremely short texts with variational manifold regularization. *Machine Learning*, 110:1029–1066.

McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Pang, Y., Wang, W., Tay, F. E., Liu, W., Tian, Y., and Yuan, L. (2022). Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pages 604–621. Springer.

Rodríguez, P., Laradji, I., Drouin, A., and Lacoste, A. (2020). Embedding propagation: Smoother manifold for few-shot classification. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*, pages 121–138. Springer.

Shaham, U., Stanton, K., Li, H., Nadler, B., Basri, R., and Kluger, Y. (2018). Spectralnet: Spectral clustering using deep neural networks. *arXiv preprint arXiv:1801.01587*.

Tian, Y., Krishnan, D., and Isola, P. (2020). Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer.

Tomar, V. S. and Rose, R. C. (2014). Manifold regularized deep neural networks. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Tomar, V. S. and Rose, R. C. (2016). Graph based manifold regularized deep neural networks for automatic speech recognition. *arXiv preprint arXiv:1606.05925*.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Wang, Y., Huang, H., Rudin, C., and Shaposhnik, Y. (2021). Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *The Journal of Machine Learning Research*, 22(1):9129–9201.

Xie, Z., Zhang, Z., Cao, Y., Lin, Y., Bao, J., Yao, Z., Dai, Q., and Hu, H. (2022). Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9653–9663.

Yonghe, C., Lin, H., Yang, L., Diao, Y., Zhang, S., and Xiaochao, F. (2019). Refining word reesprentations by manifold learning. In *Proc. 28th Int. Joint Conf. Artif. Intell*, pages 5394–5400.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR.

Zhang, Q., Wang, Y., and Wang, Y. (2022). How mask matters: Towards theoretical understandings of masked autoencoders. *Advances in Neural Information Processing Systems*, 35:27127–27139.

Zhou, J., Wei, C., Wang, H., Shen, W., Xie, C., Yuille, A., and Kong, T. (2021). ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*.

Zhou, L., Liu, H., Bae, J., He, J., Samaras, D., and Prasanna, P. (2022). Self pre-training with masked autoencoders for medical image analysis. *arXiv preprint arXiv:2203.05573*, 1(3).

A Experimental setup

Environment details. `MAGMA` builds upon the solo-learn (da Costa et al., 2022) library of self-supervised methods for unsupervised visual representation learning. All methods are implemented using PyTorch 1.13 and PyTorch Lightning 1.7.7. The following GPUs are used, depending on availability: NVIDIA GeForce RTX 2080 Ti, NVIDIA Tesla V100, and NVIDIA A40.

Datasets. We conduct our experiments on the following four benchmark datasets:

- **CIFAR-100** (Krizhevsky et al., 2009) consists of 60,000 color images (32x32 pixels) divided into 100 classes, with 500 training images and 100 test images per class. This large number of classes with relatively few images per class pushes models to learn nuanced, discriminative representations for robust classification.

- **STL-10** (Coates et al., 2011) Contains 5,000 labeled training images, 8,000 test images, and 100,000 unlabeled images (96x96 pixels) across 10 classes. This setting of abundant unlabeled data allows the exploration of self-supervised representation learning techniques, offering a valuable testbed for scenarios where labeled data is scarce.

- **Tiny-ImageNet** (Le and Yang, 2015) is a downsized version of ImageNet with 200 classes, featuring 100,000 training images, 10,000 validation images, and 10,000 test images (64x64 pixels). This dataset bridges the gap between smaller benchmarks and full ImageNet, allowing experimentation with larger-scale image recognition tasks while maintaining computational feasibility.

- **ImageNet-100** (Tian et al., 2020) is a curated subset of the full ImageNet with approximately 130,000 images (variable resolutions) across 100 classes. It provides a standard train/test split, offering a manageable platform to test the scalability and efficiency of models before moving to the full complexity of ImageNet.

This collection of datasets provides a larger range of image classification challenges by varying scales, class complexities, and train/test splits. This suite enables a robust evaluation of the effectiveness of representation learning methods and their generalization across diverse scenarios.

Pretraining hyperparameters. We split the parameters into three categories: (i) common parameters across all methods and datasets, (ii) parameters used for the MAE-based methods (MAE (He et al., 2022), M-MAE, U-MAE (Zhang et al., 2022), and MU-MAE), (iii) parameters used for SimCLR Tian et al. (2020), M-SimCLR, VICReg (Bardes et al., 2021), and M-VICReg. The complete configuration files for all combinations of datasets and methods can also be found in the attached code archive.

(i) Common parameters. All methods use AdamW as an optimizer, with an initial warmup phase of 10 epochs, and an initial learning rate of $3e-5$ decaying to 0 via cosine annealing. Normalization is applied using the specific mean and standard deviation computed across each given dataset.

(ii) MAE-based methods. Mask ratio for all parameters is 0.75, following He et al. (2022). For U-MAE and MU-MAE, the uniformity weight is set to 0.01, following Zhang et al. (2022). The weight for the `MAGMA` loss is set to 1. For augmentations, we use a random resized crop (scale ranging between 0.08 and 1), followed by a random horizontal flip with a probability of 0.5. The crop is resized to 32×32 for CIFAR-100, 64×64 for Tiny-ImageNet, 96×96 for STL-10, and 224×224 for ImageNet-100. All other parameters unrelated to the regularization terms are shared between all methods, and only depend on the dataset. These can be seen in Table 4.

(iii) Non-generative SSL methods. For SimCLR and M-SimCLR we use a temperature of 0.2. For VICReg and M-VICReg, we use the best weights from Bardes et al. (2021) for the similarity, variance, and covariance loss terms (25, 25, and 1). The hidden dimensionality of the projector is equal to 2048 for all. For augmentations, each method follows the parameters described in the original paper. The rest of the relevant parameters can be found in Table 5.

Table 4: Sets of differing parameters for MAE, M-MAE, U-MAE, and MU-MAE across the given datasets

Dataset	Backbone	Patch Size	Epochs	Batch Size	lr	e_{st} (Reg. warmup)
CIFAR-100	ViT-Tiny	4	2000	256	$1.5e^{-4}$	60
Tiny-ImageNet	ViT-Tiny	8	800	512	$1.0e^{-3}$	10
STL-10	ViT-Tiny	12	800	512	$3.0e^{-4}$	10
ImageNet-100	ViT-Base	16	400	256	$1.5e^{-4}$	10

Table 5: Sets of differing parameters for SimCLR, M-SimCLR, VICReg, and M-VICReg across the given datasets

Dataset	Backbone	Patch Size	Epochs	Batch Size	lr	e_{st} (Reg. warmup)
CIFAR-100	ViT-Tiny	4	1000	256	$1.0e^{-3}$	10
Tiny-ImageNet	ViT-Tiny	8	1000	256	$1.0e^{-3}$	10
STL-10	ViT-Tiny	12	1000	256	$1.0e^{-3}$	10
ImageNet-100	ViT-Tiny	16	200	256	$1.0e^{-3}$	10

B Additional evaluations

We further demonstrate the effectiveness of `MAGMA` when evaluated on unseen datasets in table 6, offering substantial improvements over MAE.

Table 6: Linear probing accuracy of MAE and M-MAE models pretrained on ImageNet-100 across various datasets. Adding `MAGMA` significantly improves results when evaluated on unseen datasets.

Method	CIFAR-100	STL-10	Tiny-ImageNet	ImageNet-100
MAE	31.5	67.8	27.8	58.0
M-MAE (ours)	**51.6**	**84.8**	**43.1**	**69.0**

`MAGMA` is designed to enhance self-supervised representation learning at the pretraining phase. To demonstrate the impact, we keep our supervised fine-tuning strategy as simple as linear probing. Full fine-tuning (especially in low-data regimes) can lead to overfitting to the target dataset, completely defeating the purpose and ruling out the impact of the regularization. That is what we also observe in the new Table 7, with results between the baseline method and ours being close to identical. Zhang et al. (2022) notice the same effect of fine-tuning on their regularization method as well.

Table 7: Finetuning accuracy of MAE and M-MAE models pretrained on ImageNet-100. No significant differences can be seen.

Method	CIFAR-100	STL-10	Tiny-ImageNet	ImageNet-100
MAE	**76.9**	82.2	63.1	79.8
M-MAE (ours)	75.6	**85.5**	63.1	**80.6**

We acknowledge the importance of evaluating on ImageNet-1K, and therefore we include in Table 8 preliminary results of applying `MAGMA` to it. It's important to note that the results use a suboptimal batch size of 256 due to computational constraints, and as such are below baselines reported in the literature. Still, we notice a significant improvement when combining `MAGMA` with the uniformity loss of U-MAE, resulting in a 6.2% increase over MAE, and a 4.7% over U-MAE, showcasing the applicability of our regularization.

Table 8: Linear probing accuracy pretrained on ImageNet-1K. While MU-MAE offers significant improvements, M-MAE does not improve upon the baseline.

Dataset	MAE	M-MAE (ours)	U-MAE	MU-MAE (ours)
ImageNet-1K	47.1	46.5	48.6	**53.3**

C Additional visualizations

PCA. Inspired by Amir et al. (2021), we take our pretrained MAEs and extract features from each layer (in this case, we isolate the *key* features from the self-attention mechanism), apply PCA, and show the leading component. This provides a qualitative analysis of the quality of the intermediate representations learned by the models, showcasing the impact of the added regularization term. One visible pattern is the reduction in noise, specifically in the first and last layers, that M-MAE exhibits when compared to its MAE counterpart

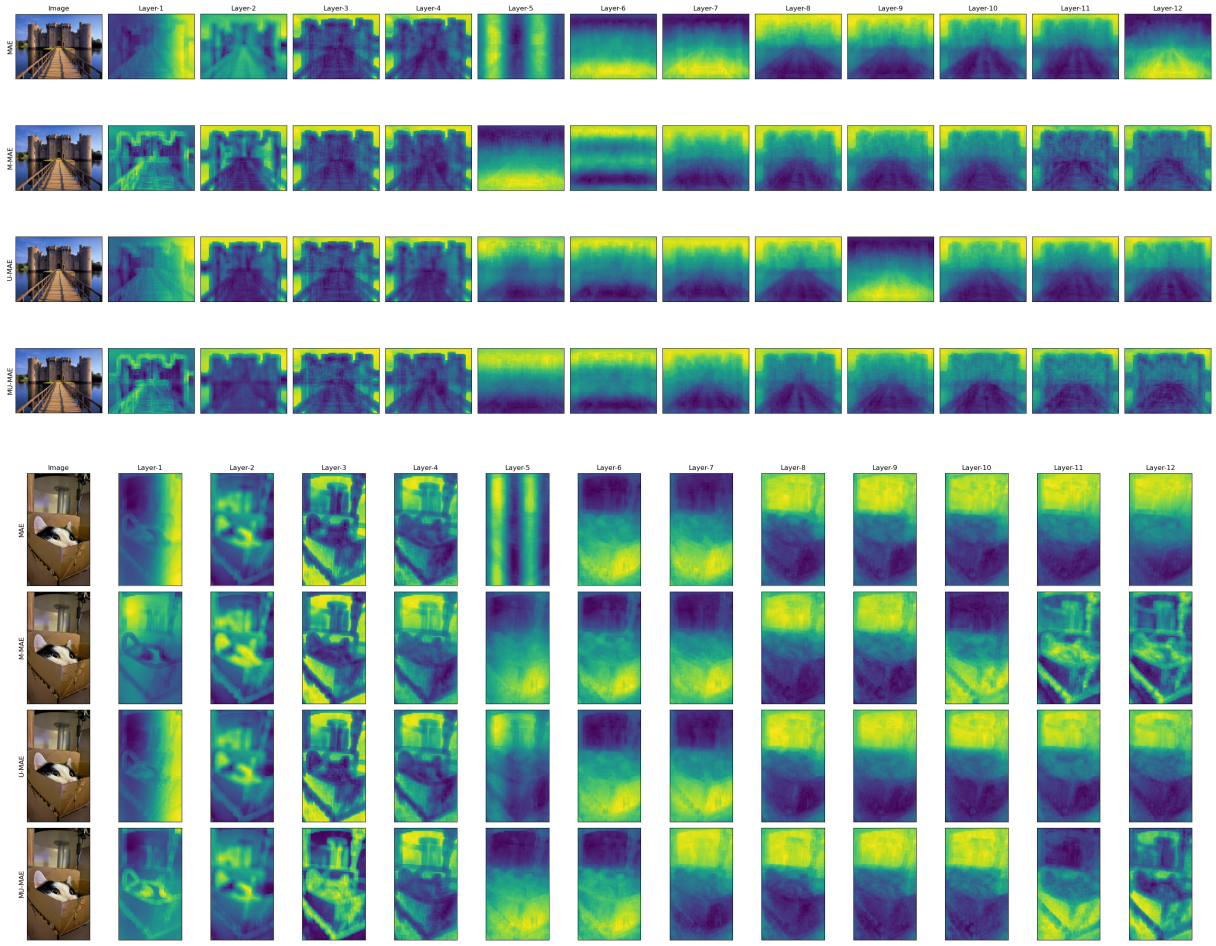


Figure 5: Visualization of PCA's leading component for features extracted from different layers of a ViT-B pretrained using MAE, M-MAE (ours), U-MAE, and MU-MAE (ours).

Attention maps. We investigate the impact of the different regularizations on the self-attention maps of the ViT-B architecture's last layer. To this end, we randomly select images from the ImageNet-1K validation set and visualize their corresponding attention maps in Figure Figure 6. Our observations reveal that the baseline MAE model often tends to attend to the background of the image, in line with findings from prior work (Lehner et al., 2023). In contrast, we notice differences when applying `MAGMA`: it appears to promote a semantic separation of the attention focus, where the model tends to attend primarily to either the background or the central object, but rarely both simultaneously. This suggests that `MAGMA` guides the model towards learning more specialized and semantically coherent representations, improving its ability to distinguish between foreground and background elements.

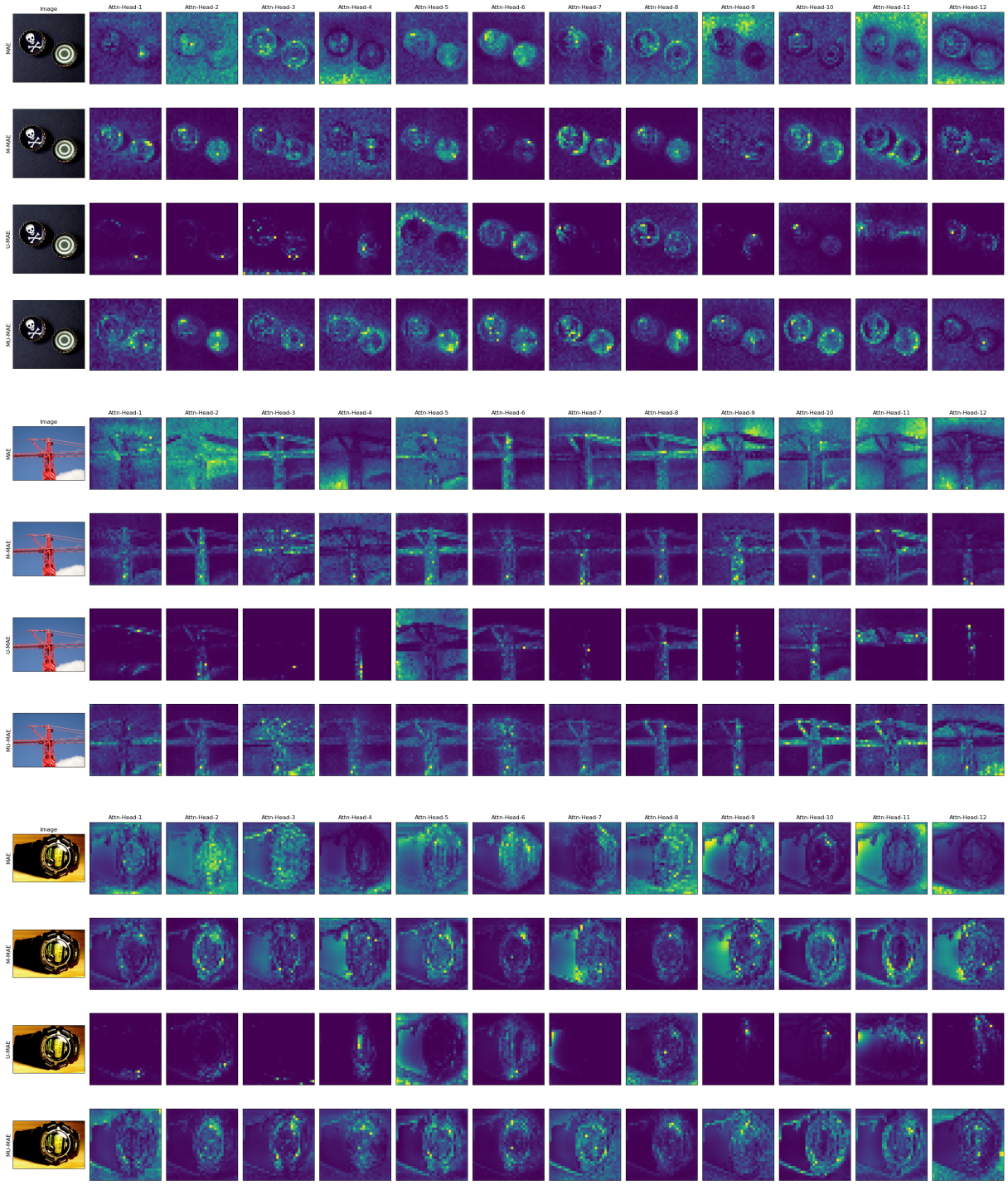


Figure 6: Attention maps from the 12 attention heads of the last layer of a ViT-B. The attention maps come from three different images, and for each image, we extract them over the four MAE-based methods evaluated: MAE, M-MAE (ours), U-MAE, MU-MAE (ours)

3

Deep Learning

Deep learning, a subfield of machine learning, has fueled the biggest breakthroughs in the field of Artificial Intelligence for the past decade. Inspired by the biological structure of **neurons** in our brain, deep learning uses **Artificial Neural Networks** (ANNs) as the backbone algorithm, to learn hierarchical representations of data. Although the idea was introduced more than half a century ago (1943) by McCulloch and Pitts [25], it wasn't until 1986, with the advent of the backpropagation algorithm by Rumelhart, Hinton, and Williams [30], that training deep, multi-layered networks became practical. Later, in 1998, the field gained momentum, with the introduction of the **Convolutional Neural Network** (CNN) by LeCun and Bengio [24], now a cornerstone of computer vision applications. Today, deep learning is redefining what's possible, enabling breakthroughs in self-driving cars, art, 3D modeling, medical imaging, and beyond. This chapter will provide an overview of the core concepts and key architectures of deep learning, particularly those relevant to computer vision. For a comprehensive exploration, readers are encouraged to consult Goodfellow et al. [17].

3.1. Deep Feedforward Networks

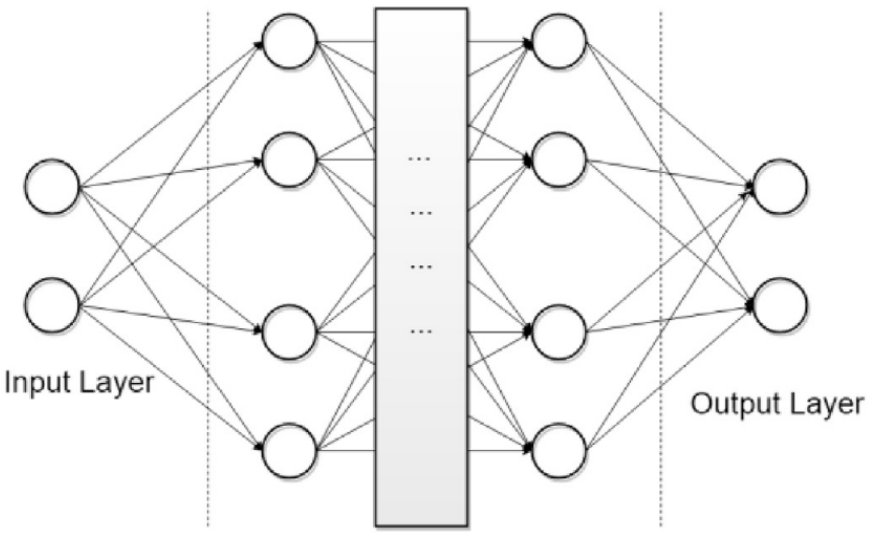


Figure 3.1: A typical architecture of a feedforward neural network.

Inspired by the biological neuron, a feedforward network is comprised of layers of artificial **neurons**. Each neuron receives an input vector $\mathbf{x} = (x_1, x_2, ..., x_n)$, where each element is associated with a weight w_i. The neuron calculates a weighted sum of its inputs, adds a bias term b, and produces a

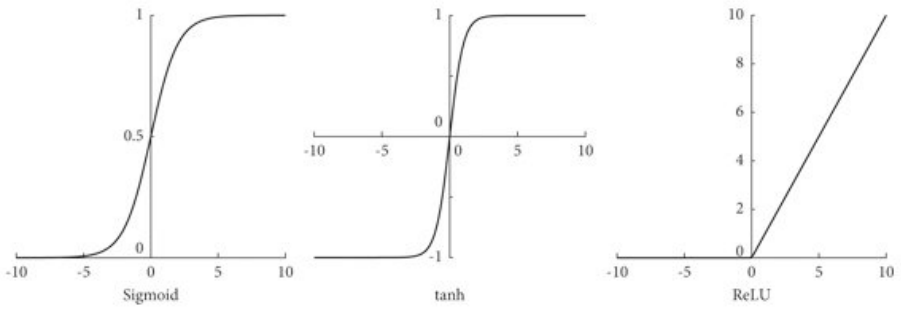


Figure 3.2: Three examples of activation functions. Taken from [23]

.

single output $y = \sum_{i=1}^{n} w_i x_i + b = w^T x + b$, where w is a vector comprised of the neuron's weights. Multiple neurons can be stacked together to form a **layer**. A layer can be represented as a function $f_l(\mathbf{x})$, where **x** is the input to the layer, and f_l encapsulates the computations performed by all the neurons in that layer. A feedforward neural network is organized as a sequence of layers, with information flowing in a forward manner, as can be seen in Figure 3.1. Given layers $f_1, f_2, \ldots, f_n$, the network f can be written as $f = f_1 \circ f_2 \circ \cdots \circ f_n$. The process of propagating input data through the network to obtain an output is called a **forward pass**.

Given a set of input data **X** and corresponding target outputs **Y**, the goal of training a neural network is to find the optimal weights and biases θ (also called the network parameters) that best approximate an unknown function $f^*(\mathbf{x})$, such that the network's output $\hat{y} = f(\mathbf{x}; \theta)$ closely matches the desired output y for all inputs. However, in their current form, each layer can only model linear relationships due to the weighted sum operation. In contrast, most real-world phenomena exhibit non-linear behavior. Therefore, we need to introduce **non-linearity** into our model to capture the complexities of these relationships. It turns out that with the added non-linearity, a deep feedforward network with a single hidden layer can approximate any continuous function on a compact subset of $\mathbb{R}^n$, as stated by the **Universal Approximation Theorem** [22].

Activation functions

Activation functions introduce non-linearities in a deep network which allows it to model non-linear relationships. These functions are applied to the output of each neuron before passing it to the next layer. The output y of a neuron now becomes $y = g(w^T x + b)$, where g can be any non-linear differentiable function.

The three most common examples of activation functions can be seen in Figure 3.2. Historically significant, the **sigmoid function** (logistic function) squashes its input into the range (0, 1). This makes it suitable for binary classification tasks, where the output can be interpreted as a probability. Mathematically, it's defined as: $\sigma(x) = \frac{1}{1+e^{-x}}$. However, while initially popular, this activation function did not scale well to deeper networks.

Similar to the sigmoid function, the **hyperbolic tangent** (tanh) function squashes its input, but into the range (-1, 1). Centering the output range around zero can sometimes lead to faster convergence during training. Mathematically, it's defined as: $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

The **Rectified Linear Unit** (ReLU), the most popular activation function in modern deep learning, is simple yet effective. It outputs zero for negative inputs and the input itself for positive inputs. Mathematically, it's defined as: $\text{ReLU}(x) = \max(0, x)$. ReLU mitigates some of the issues encountered in the sigmoid and hyperbolic tangent functions, but it can also suffer from its own issues (see "dying ReLU").

3.2. Optimization

Having established that a neural network with sufficient depth and non-linear activation functions can approximate any continuous function, a critical question arises: How do we find the optimal parameters

of the network to best approximate our target function? This process is known as optimization.

Loss functions

The goal of optimization is to minimize the discrepancy between the network's predictions and the ground truth labels in our training data. This discrepancy is quantified by a loss function, (sometimes also called a cost or objective function). The loss function measures how well the network fits our desired goal, with lower values indicating better performance.

The choice of loss function depends on the specific task at hand. For regression tasks (predicting continuous values, e.g. predicting the price of a stock on a given day), the standard loss function is the mean squared error (MSE). MSE is defined as:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.1}$$

In classification tasks (predicting discrete classes, e.g. predicting the type of object in an image), the cross-entropy (CE) loss is frequently employed. In this scenario, the neural network's output layer often uses a softmax activation function to convert raw scores into a probability distribution over the classes. Whereas MSE quantified the distance between two vectors, the CE loss tells how dissimilar the predicted probability distribution is from the "real" one. Given a single sample and C classes to predict, the loss is defined as:

$$CE = -\sum_{c=1}^{C} y_c \log(\hat{y}_c) \tag{3.2}$$

The total cross-entropy loss over a dataset of N samples is then computed by averaging the loss for each sample:

Gradient Descent

The loss function provides a scalar value that guides the optimization process, indicating the direction in which the parameters should be adjusted to improve performance. By looking at the gradients of the loss function with respect to the network parameters, we can find the direction pointing towards the configuration of parameters that would minimize said loss. This simple idea is at the core of the **Gradient Descent** algorithm, upon which neural network optimization relies upon.

Gradient descent starts at an initial point in the optimization landscape and iteratively takes steps in the direction of the steepest descent, guided by the negative gradient of the loss function. This gradient represents the direction in which the loss function decreases most rapidly. An illustration of gradient descent on the function $f(x) = \frac{1}{2}x^2$ is shown in Figure 3.3.

The parameters are updated proportionally to the magnitude of the gradient and a hyperparameter called the learning rate (η). The learning rate controls the step size and is a crucial factor in the effectiveness of the algorithm. Too small a learning rate can lead to slow convergence, while too large a learning rate can cause the algorithm to overshoot the minimum and even diverge. Mathematically, the update rule for gradient descent is:

$$\theta' = \theta - \eta \nabla_\theta L(\theta) \tag{3.3}$$

While seemingly straightforward, gradient descent can be computationally expensive when dealing with large datasets. Calculating the gradient over the entire training set at each iteration can be time-consuming. To address this, mini-batch gradient descent is often employed. In this approach, the dataset is divided into small batches, and the gradient is computed and parameters updated for each batch. This introduces some noise into the gradient estimate but significantly speeds up the optimization process.

One important consideration in gradient descent is the possibility of converging to a local minimum instead of the global minimum. A local minimum is a point where the loss function is lower than all

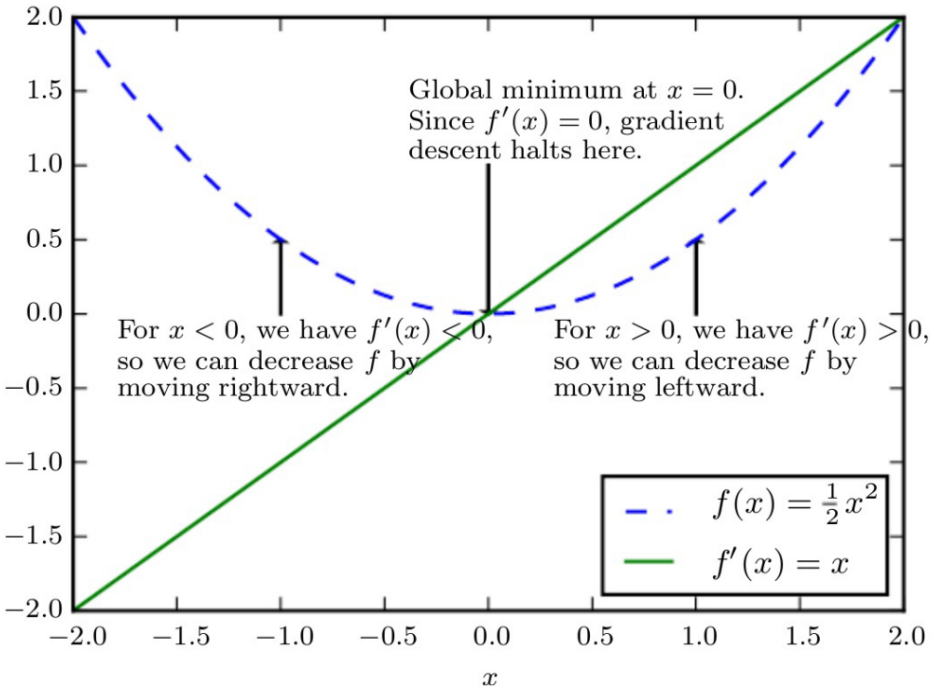


Figure 3.3: Illustration of the gradient descent algorithm. Taken from [17]

.

neighboring points, but there might exist other points with even lower loss values elsewhere in the optimization landscape. Gradient descent, in its basic form, cannot escape these local minima, potentially leading to suboptimal solutions. Various techniques, such as momentum and adaptive learning rates, can be employed to overcome this limitation and improve convergence to the global minimum.

Backpropagation

Gradient descent is just the general framework for optimizing neural networks. Applying it naively, however, leads to impractical solutions. **Backpropagation**, introduced by Rumelhart, Hinton, and Williams [30], is an algorithm that can efficiently calculate the gradients of the loss function with respect to each parameter in the network. It does so using the chain rule of derivatives.

Backpropagation works by propagating the error backward through the network, starting from the output layer. At each layer, the gradient of the loss function with respect to that layer's output is computed. This gradient is then used, along with the local gradient of the activation function, to compute the gradients with respect to the weights and biases of that layer. These gradients are then used to update the parameters via gradient descent.

3.3. Regularization

The goal of training a neural network is to generalize well to new, unseen data. If the network performs well on the training data alone, we are in the overfitting regime, meaning the model learns to memorize the training data at the expense of generalizability. Combating overfitting is typically done via various regularization techniques. On the other hand, if the regularization is too strong, we end up with a model that is not aligned well enough with the training data because of over-generalization. An illustration of these phenomena can be seen in Figure 3.4.

Two common techniques are L1 and L2 regularization, which add penalty terms to the loss function. L1 regularization adds a penalty proportional to the absolute value of the weights, encouraging sparsity (many weights becoming zero). L2 regularization, or weight decay, adds a penalty proportional to the square of the weights, favoring smaller weights in general.

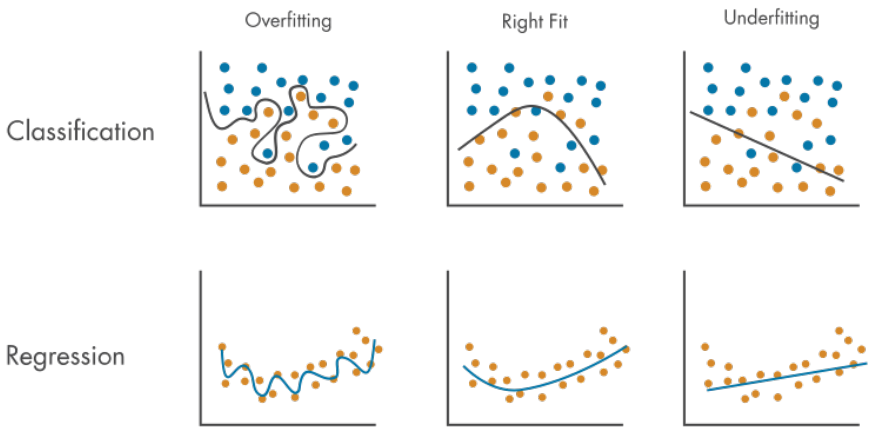


Figure 3.4: Overfitting and underfitting illustrated for regression and classification tasks. Taken from https://www.mathworks.com/discovery/overfitting.html

.

Another widely used method is dropout regularization. During training, dropout randomly deactivates a fraction of neurons in each layer, forcing the network to learn redundant representations and preventing any single neuron from becoming too specialized.

Data augmentation is another strategy to combat overfitting. It involves creating new training samples by applying random transformations to the existing data, such as rotations, flips, or translations. This artificially expands the dataset and exposes the model to a wider range of variations, improving its robustness to common perturbations found in practice.

In addition to these techniques, various other methods can be used for regularization. For example, early stopping involves monitoring the validation loss during training and stopping when it starts to increase, preventing the model from overfitting to the training data. Furthermore, certain loss functions, such as the focal loss, can be designed to inherently address class imbalance and improve generalization.

3.4. Convolutional Neural Networks

There are a multitude of neural network architectures in deep learning, but in the field of computer vision, **convolutional neural networks** (CNNs) are ubiquitous. Introduced by LeCun and Bengio [24] they are designed to process data with a grid-like topology (e.g. images). The main component of the CNN is the convolutional layer. Unlike traditional fully connected layers where every neuron connects to all neurons in the preceding layer, convolutional layers employ a more localized connectivity pattern. They utilize filters, also known as kernels, filters or feature detectors, which are small matrices of learnable parameters denoted as W. Given an input image denoted as X, the resulting output Y can be represented as:

$$Y_{i,j} = \sum_m \sum_n W_{m,n} \cdot X_{i+m,j+n} + b \tag{3.4}$$

An example of this operation can also be seen in Figure 3.5. Typically, multiple kernels are stacked together, resulting in the computation of multiple **channels**. Three desirable properties arise from this operation [17]:

- **Sparse interactions:** By having the kernel smaller than the input, only a neighborhood of pixels will interact with a specific output neuron. In contrast, in traditional neural networks, each input interacts with each output. Thus, CNNs reduce the number of parameters compared to fully connected networks, making them more computationally efficient and less prone to overfitting.
- **Parameter sharing:** Traditional neural networks use each weight only once when computing

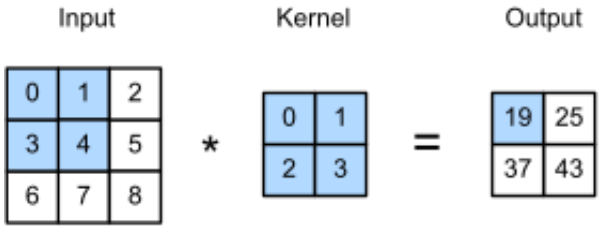


Figure 3.5: Example of a convolutional operation. Courtesy of https://www.d2l.ai

.

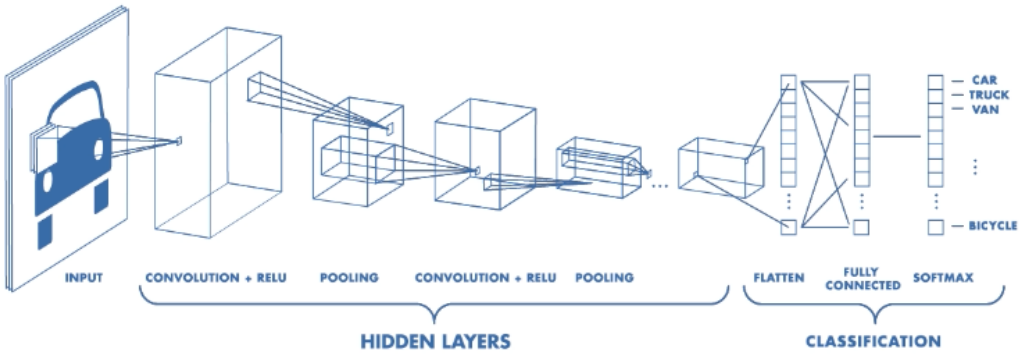


Figure 3.6: Example of a convolutional architecture

.

an output. In CNNs however, each kernel weight is used at (roughly) all positions in the input image. This weight sharing is drastically more efficient than dense matrix multiplication in terms of memory requirements.

- **Translation equivariance:** Convolutional layers exhibit equivariance to translation, which means that if the input is translated, the output will also be translated by the same amount. This is a desirable property in image processing, as it allows the network to detect features regardless of their position within the image. For example, a filter trained to detect a vertical edge will still detect it if the edge is shifted to a different location in the image.

To further enhance computational efficiency and achieve invariance to minor translations in the input, pooling layers are often inserted between convolutional layers. A pooling layer replaces the output of a convolution with a summary statistic of its nearby outputs. Common pooling operations include max pooling and average pooling. Max pooling extracts the maximum value within a local region of the feature map, while average pooling calculates the average value. An example of the full CNN architecture can be seen in Figure 3.6.

Deep Residual Networks

As neural networks grow deeper, a phenomenon known as the **vanishing gradient** problem arises. As more and more layers are added to the network, as the error signal travels back through the layers, the gradient shrinks to almost 0, resulting in the early layer receiving no updates.

Deep Residual Networks (ResNets) [19], offer a solution to this. The authors introduce skip connections, also known as shortcut connections. These connections allow the input of a layer to bypass one or more subsequent layers and be added directly to the output of those layers. This creates a ”shortcut” path for information flow, allowing the network to learn residual mappings rather than just the original mapping. This skip-connection can also be visualized in Figure 3.7.

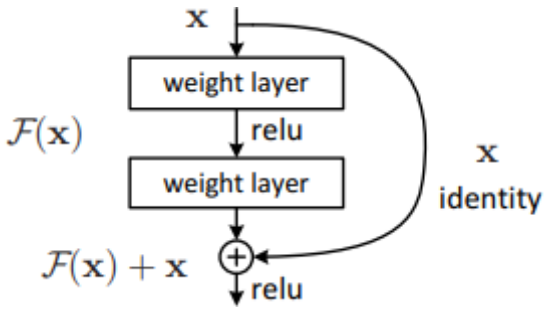


Figure 3.7: Building block of residual learning. Taken from [19]

.

ResNets have become immensely popular due to their ability to train deep networks without sacrificing performance. They have achieved state-of-the-art results on various image recognition tasks and have become a standard architecture in many computer vision applications. As such, they have been used as part of our experimentation on manifold-aware regularization.

3.5. Vision Transformers

In NLP, the Transformer [31] architecture has emerged as the default approach across various tasks. The Transformer is a neural network architecture, which, at its core, relies on the **self-attention** mechanism. Self-attention enables a model to weigh the importance of different elements within an input sequence when generating an output. Given an input matrix $X \in \mathbb{R}^{N \times D}$, where N is the number of inputs and D is the dimensionality of each input vector, self-attention maps X based on three matrices:

- $Q = XW^Q \in \mathbb{R}^{N \times d}$, where W^Q is a learnable weight matrix, and d is the dimensionality of the query vectors
- $K = XW^K \in \mathbb{R}^{N \times d}$, where W^K is a learnable weight matrix, and d is the dimensionality of the key vectors
- $V = XW^V \in \mathbb{R}^{N \times d_v}$, where W^V is a learnable weight matrix, and d_v is the dimensionality of the value vectors

Based on these, the attention mechanism computes an attention score for each pair of query and key elements. These scores are normalized using a softmax and finally used for a weighted sum of the value vectors. This can be summarized as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V \tag{3.5}$$

The resulting outputs are context-aware representations, incorporating information from all other inputs in the sequence. The Transformer architecture also contains a feedforward network (FFN) applied independently to each position and residual connections around both the attention and FFN layers, followed by layer normalization.

Vision transformers [14] adapt this architecture to image data. A ViT divides an image into fixed-size patches. These patches are then flattened into vectors and linearly projected into a higher-dimensional space. Positional encodings are added to retain spatial information, as self-attention is inherently permutation-invariant. The resulting vectors are passed through several transformer layers. An overview of the architecture can be seen in Figure 3.8.

The self-attention mechanism in ViTs allows the model to weigh the importance of different patches and their relationships with each other, regardless of their spatial distance in the image. This provides an advantage over CNNs, which are limited to more local receptive fields. However, ViTs have their own challenges. Perhaps most notable is their reliance on considerable amounts of training data. **Self-supervised learning** techniques have emerged as a promising solution to alleviate this issue. These techniques leverage unlabeled data to learn useful representations, which can then be fine-tuned on

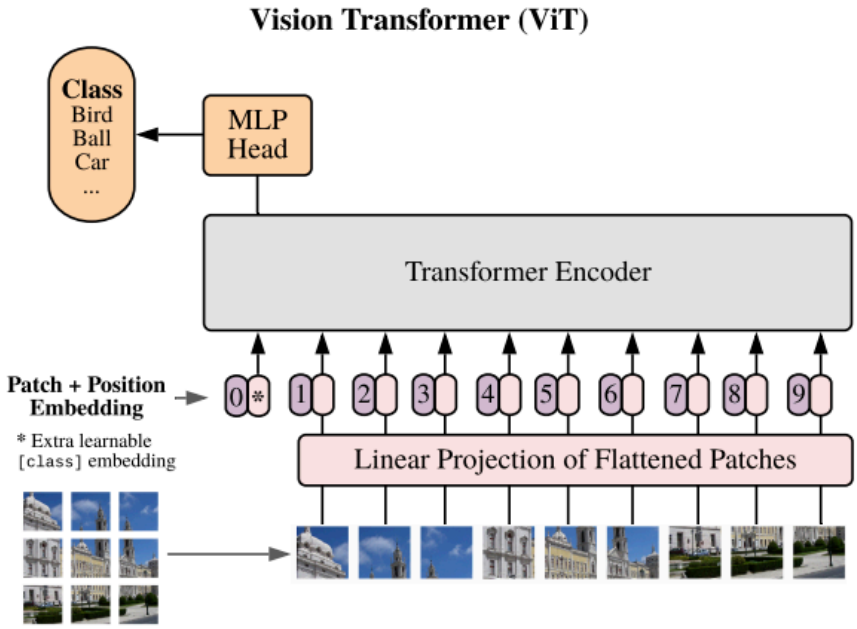


Figure 3.8: Overview of the vision transformer architecture. Taken from [14]

.

smaller labeled datasets. One such technique is the Masked Autoencoder (MAE) [20], which learns to reconstruct masked image patches, forcing the model to understand the underlying structure of images and capture meaningful features. This approach has shown significant promise in improving the data efficiency of ViTs, enabling them to achieve comparable performance to CNNs with significantly less labeled data.

4

Self-Supervised Learning

Deep learning models have revolutionized various fields, yet their success traditionally hinges on vast amounts of labeled data. Acquiring such data is often labor-intensive, expensive, and sometimes impossible due to inherent scarcity (e.g. medical data of rare diseases). Furthermore, the improvements achieved through supervised learning alone have begun to plateau, as evidenced in Figure 4.1.

Self-supervised learning (SSL) emerges as a promising solution to this challenge, enabling models to learn robust, transferable representations from unlabeled data, which is abundant and diverse. This approach circumvents the need for explicit labels by devising supervisory signals directly from the data itself in the form of a **pretext task**. The pretext task, whether it involves predicting missing words in a sentence, reconstructing parts of an image, or discriminating between transformed versions of the same image, instructs the model to capture generalizable representations of its inputs during a **pretraining** phase. These learned representations can then be tuned on smaller labeled datasets for downstream tasks, often yielding superior performance, in what is called the **fine-tuning** stage. In NLP, this approach has led to the development of powerful language models with tremendous success. The importance of self-supervised learning is also recognized by Yann LeCun, a leading figure in AI research, who aptly states that “If intelligence is a cake, the bulk of the cake is self-supervised learning” underscoring the critical role of self-supervised learning.

The field of SSL is broadly categorized into two main methodologies: **discriminative** and **generative**. Discriminative methods, with instance discrimination as their primary task, focus on distinguishing between different instances of data. They encompass four main families: **contrastive** methods, which learn by contrasting similar and dissimilar examples; **distillation** methods, which transfer knowledge from a larger teacher model to a smaller student model; **clustering** methods, which group similar data points together; and **information-maximization** methods, which aim to maximize the mutual information between different views of the data. On the other hand, generative methods strive to learn the underlying distribution of data in pixel space, often through computationally expensive procedures such as autoregressive models or variational autoencoders. Among these, masked image modeling, which involves predicting masked-out portions of an image, has gained significant traction, recently

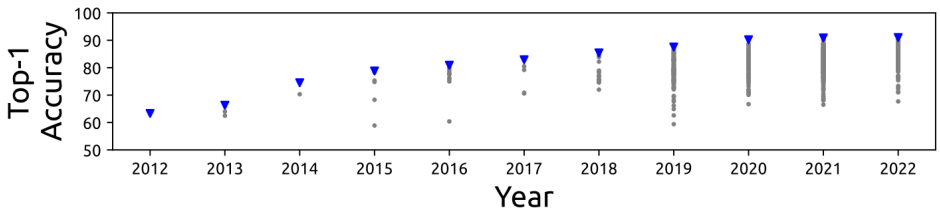


Figure 4.1: ImageNet accuracy over the years. Taken from [28]

.

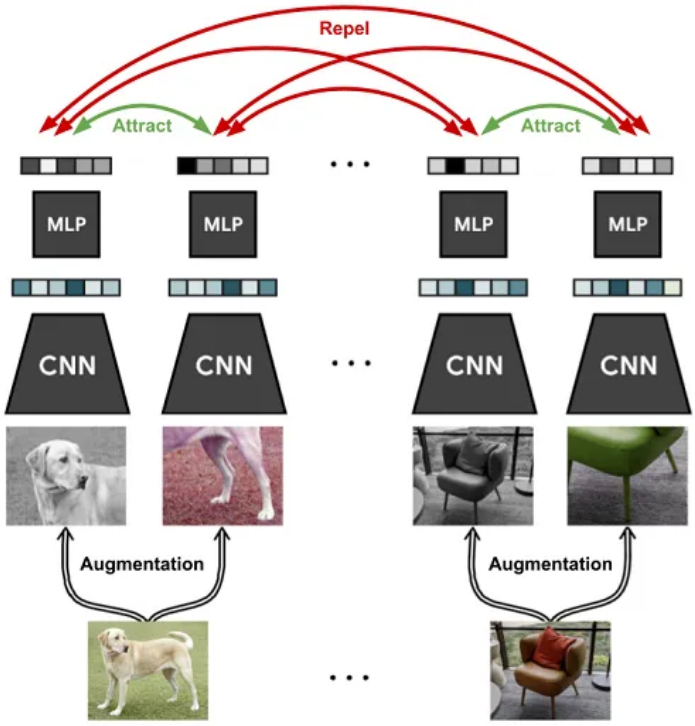


Figure 4.2: Simplified SimCLR architecture [1]

surpassing discriminative methods in performance.

In the remainder of this chapter, we will delve into these families of models, providing a concise overview of the diverse landscape of self-supervised learning.

4.1. Contrastive approaches

Contrastive methods constitute the most popular family of discriminative SSL approaches. Pioneering works in this area include SimCLR [6, 7], MoCo [21, 10, 9], alongside numerous other variations and improvements.

The fundamental principle of contrastive learning is to train a model to distinguish between pairs of examples labeled as positive (similar) or negative (dissimilar). Typically, positive pairs are generated by applying various augmentations to the same data point, while negative pairs consist of augmentations from different data points. These augmentations, which include transformations like cropping, resizing, color jittering, and rotation, play a vital role in preventing the model from learning trivial solutions and encourage it to capture meaningful representations.

The typical contrastive learning framework involves several key components. First, data augmentation techniques are applied to create multiple views of each data point, forming the basis for positive and negative pairs. Next, an encoder, often a convolutional neural network (CNN) for images, is employed to extract feature vectors (representations) from these augmented views. Optionally, a projector can further transform these representations into a lower-dimensional space, where the contrastive loss is applied. This loss function guides the model to produce similar representations for positive pairs and dissimilar representations for negative pairs. The InfoNCE (Noise Contrastive Estimation) [27] loss is a popular choice for this purpose.

Contrastive methods have proven to be a powerful tool in SSL, demonstrating impressive results on various downstream tasks, across different modalities (e.g. [29]). However, they are not without challenges. One significant drawback is their dependence on large batch sizes to provide a sufficient number of negative pairs for effective training, which can be computationally expensive. Additionally, the

[1] Taken from Medium

quality of negative samples can greatly influence performance, leading to the development of strategies like memory banks and momentum encoders to mitigate this issue.

4.2. Distillation approaches

Distillation methods within the realm of self-supervised learning (SSL) offer an alternative approach to contrastive learning, forgoing the need for negative samples and the associated contrastive loss. Pioneered by BYOL [18], these methods introduce an asymmetric framework where a student model is trained to predict the representations of a teacher model. Similar approaches, such as SimSiam [8] and DINO [4], have followed suit, achieving impressive results in various SSL tasks.

A key aspect of these methods is the use of stop-gradient operations and projection/prediction MLPs (multi-layer perceptrons). The stop-gradient operation prevents the gradients from flowing back to the teacher model, ensuring that the student learns to predict the teacher's representations without directly influencing them. The projection/prediction MLPs further transform the representations, adding complexity and potentially enhancing the learning process.

However, distillation methods are not without challenges. One significant issue is the potential for representational collapse, where both the student and teacher networks converge to predict a single, trivial representation. The susceptibility to representational collapse and the lack of a clear solution pose challenges to the extendability of distillation methods.

4.3. Clustering approaches

Clustering approaches in SSL offer a distinct perspective, focusing on grouping similar data points together without relying on explicit labels. One of the earliest methods in this domain is DeepCluster [3], which alternates between clustering image features and using the cluster assignments as pseudo-labels to train the network. This iterative process aims to learn representations that naturally align with the underlying structure of the data.

Building upon the concept of clustering, recent advancements have incorporated optimal transport theory, particularly the Sinkhorn-Knopp algorithm, into SSL. Optimal transport provides a mathematically principled way to measure the distance between probability distributions, enabling more robust and efficient clustering. SwAV [5], for instance, leverages optimal transport to assign codes to different augmented views of an image and then trains the network to predict these codes for other views. This online clustering approach has shown promising results in learning visual representations without the need for explicit labels or negative samples.

4.4. Information-Maximization methods

Information-maximization methods, as the name suggests, aim to maximize the information content of representations. Unlike contrastive methods, they don't rely on negative samples, and unlike distillation methods, they don't require an asymmetric architecture. Instead, they leverage novel loss functions that encourage the model to learn informative and diverse representations.

The early stages of information maximization methods saw the introduction of the W-MSE loss [15], which sought to enforce a spherical distribution of representations. This was a step towards promoting information preservation, but it had limitations in capturing complex relationships within the data. Subsequently, the Barlow Twins [32] method emerged, aiming to achieve an identity correlation matrix between different views of the data. This approach further encouraged the model to learn decorrelated features, leading to improved representation quality. VICReg [1], seen in Figure 4.3, extends this with a more comprehensive loss framework. It combines three key components: variance regularization to prevent representational collapse, invariance regularization to ensure consistent representations across different views, and covariance regularization to encourage decorrelation among features.

4.5. Masked Image Modelling approaches

Similar to the Masked Language Modeling (MLM) approach that has fueled advancements in NLP and underpinned the development of powerful language models, **Masked Image Modeling** (MIM) has

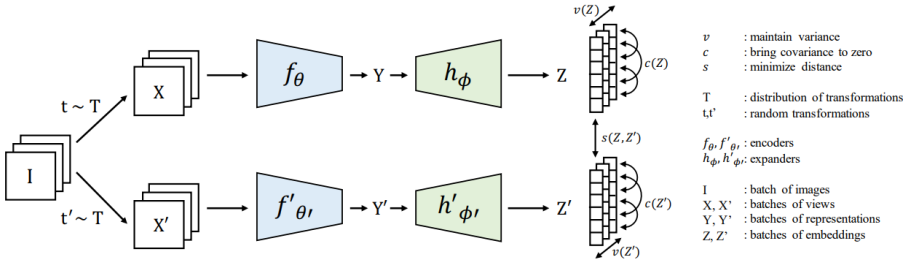


Figure 4.3: VICReg architecture. Taken from [1].

emerged as a dominant technique in SSL for computer vision. MIM has become a cornerstone in state-of-the-art models, often combined with other discriminative SSL techniques to achieve superior performance in various tasks.

The most notable example of MIM is the Masked Autoencoder (MAE) [20], seen in Figure 4.4, which has been widely adopted and integrated into numerous cutting-edge models. MAE's success can be attributed to its ability to learn robust and generalizable representations by reconstructing masked-out portions of images. While MIM typically leverages ViTs due to their effectiveness in handling masked image patches, recent work like SparK has demonstrated the potential of modern convolutional architectures in conjunction with MIM, opening up new avenues for exploration.

As MIM continues to evolve and mature, we can anticipate further innovations and refinements in this technique. The exploration of novel architectures, masking strategies, and combinations with other SSL methods holds the promise of unlocking even greater potential in self-supervised representation learning for computer vision.

In particular, we will focus on alleviating some of these limitations observed in MAEs by incorporating a simple yet effective manifold-aware regularization term. This regularization technique exploits the inherent manifold structure of representations within a batch, offering several advantages. Notably, it requires no changes to the existing architecture, incurs no additional computational cost, and does not rely on data augmentations. Empirically, we also demonstrate that this regularization approach improves the overall quality of learned representations.

As MIM continues to evolve and mature, we can anticipate further innovations and refinements in this technique. The exploration of novel architectures, masking strategies, and combinations with other SSL methods holds the promise of unlocking even greater potential in self-supervised representation learning for computer vision.

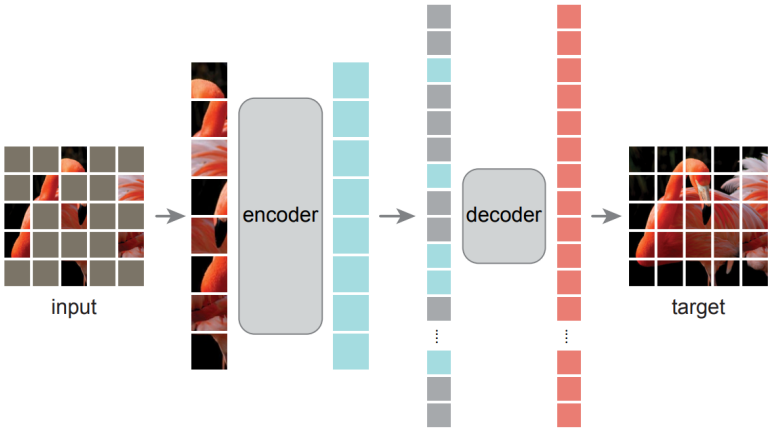


Figure 4.4: MAE architecture. Taken from [20].

5

Manifold regularization

In the realm of machine learning, the concept of a manifold provides a powerful framework for understanding the structure of complex, high-dimensional data. At its core, a manifold is a geometric object that locally resembles Euclidean space, meaning that small regions of the manifold can be approximated by flat spaces. This characteristic is crucial in machine learning, where data often resides in high-dimensional spaces, making it difficult to visualize and analyze.

The manifold hypothesis suggests that high-dimensional data often lies on or near a lower-dimensional manifold embedded within the high-dimensional space. Take, for example, images of dogs. Each image might have tens of thousands of pixels, but the different dog breeds, colors, and sizes can be boiled down to a much smaller set of variables. Manifold learning techniques help us find and use this simpler structure, making it easier to analyze and represent the data effectively.

The amount of unlabeled data in SSL allows us to more accurately estimate the underlying manifold structure of the data distribution. By leveraging a large set of unlabeled examples, models can learn to capture the intrinsic geometry of the data manifold, which in turn facilitates the discovery of more meaningful and informative representations, which would directly translate into better performance on downstream tasks. Let's take for example Figure 5.1. By only looking at the two labeled examples, a good decision boundary seems to be the straight line that separates the blue and red shapes. However, when we take into consideration all unlabeled examples as well (right), we notice new underlying factors in the data distribution that also influence our decision. The more sensible decision boundary now becomes a circle.

Introduced by Belkin et al. [2], manifold regularization emerges as a powerful technique to further exploit the manifold structure of data in self-supervised learning. By explicitly incorporating knowledge of the data manifold into the learning process, we can encourage models to learn representations that are not only semantically meaningful but also geometrically faithful to the underlying structure. The way this is done is by incorporating a loss term with a simple underlying intuition: two similar points should

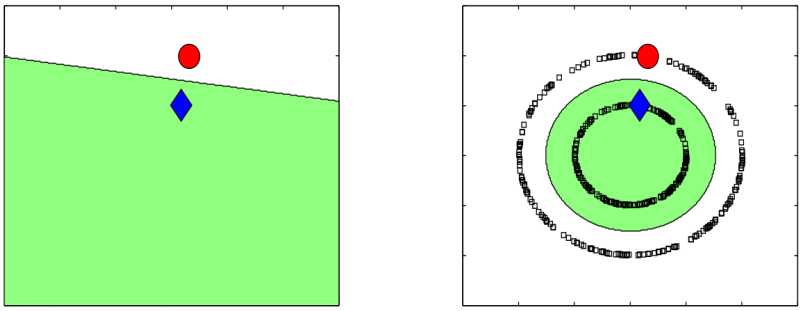


Figure 5.1: How unlabeled data (right) can influence the decision boundary between two labeled samples. Taken from [2].

map to two similar outputs. This gives rise to the following term:

$$L_{reg} = \sum_{i,j=1}^{N} (f(x_i) - f(x_j))^2 W_{ij} \tag{5.1}$$

where f is a neural network, x_i are the inputs, and W_{ij} represents a similarity metric between the inputs x_i and x_j. Based on this, we can see that L_{reg} will be large whenever both the similarity between inputs is large, as well as the distance between the outputs. This penalizes drastic "non-smooth" small surfaces across the underlying manifold.

It is important to note that the regularizer can be trivially minimized by mapping all inputs to a single point, effectively collapsing the representation space. This highlights the need for a main loss function that actively encourages the model to learn diverse and meaningful representations. In this regard, SSL methods offer a natural fit, as their objective functions typically incorporate mechanisms to prevent collapse. By combining SSL losses with manifold regularization, we can leverage the strengths of both approaches: the SSL loss guides the model to learn informative representations, while the regularization term ensures that these representations remain faithful to the underlying manifold structure. This synergy is particularly beneficial for masked image modeling MIM, contrastive, and infomax learning techniques, which have proven effective in preventing collapse. However, applying manifold regularization to distillation methods presents a greater challenge due to their inherent susceptibility to representational collapse, often requiring specific architectural modifications to mitigate this issue.

As a regularization technique applied exclusively during the training phase, manifold regularization does not impact the runtime performance or efficiency of the deployed model. This makes it an attractive addition to existing SSL frameworks, as it can be seamlessly integrated into the training process without introducing any computational overhead. Moreover, its simple formulation and flexibility make it adaptable to various model architectures and learning objectives, further highlighting its potential as a versatile tool for enhancing self-supervised representation learning.

6

Conclusion and Future Work

In this thesis, we presented a novel regularization technique, MAGMA (Manifold-Aware Graph-based Masked Autoencoder), for enhancing self-supervised representation learning within masked autoencoders. The core of our approach lies in applying a manifold-based loss term that encourages consistency and smoothness between representations across different layers of the network. Our approach does not rely on any additional augmentations, or positive/negative pairs and avoids additional forward passes, maintaining efficiency. Our extensive evaluations showcase the efficacy of MAGMA.

For **future work**, given the promising results of MAGMA in the image domain, a natural extension of our work lies in exploring its applicability to other modalities. Our approach is easy to include in self-supervised pretraining tasks for text, audio, and perhaps even video data. It would be interesting to see how the model can perform in a multimodal setting as well, enforcing consistency between representations coming from different modalities. Another extension would be to apply this to the latest state-of-the-art SSL frameworks that are based on MIM in similar settings. While these avenues offer exciting possibilities, they require significant computational resources. Another path for future work is a deeper theoretical analysis of the relationship between manifold regularization and representation learning in masked autoencoders. While our empirical results strongly suggest the effectiveness of this approach, a more rigorous theoretical foundation could help uncover the underlying mechanisms and inform further optimizations. This could involve investigating the relationship between the graph construction, the choice of similarity metric, and the specific properties of the learned representations.

References

[1] Adrien Bardes, Jean Ponce, and Yann LeCun. “Vicreg: Variance-invariance-covariance regularization for self-supervised learning”. In: *arXiv preprint arXiv:2105.04906* (2021).

[2] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. “Manifold regularization: A geometric framework for learning from labeled and unlabeled examples.” In: *Journal of machine learning research* 7.11 (2006).

[3] Mathilde Caron et al. “Deep clustering for unsupervised learning of visual features”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 132–149.

[4] Mathilde Caron et al. “Emerging properties in self-supervised vision transformers”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 9650–9660.

[5] Mathilde Caron et al. “Unsupervised learning of visual features by contrasting cluster assignments”. In: *Advances in neural information processing systems* 33 (2020), pp. 9912–9924.

[6] Ting Chen et al. “A simple framework for contrastive learning of visual representations”. In: *International conference on machine learning*. PMLR. 2020, pp. 1597–1607.

[7] Ting Chen et al. “Big self-supervised models are strong semi-supervised learners”. In: *Advances in neural information processing systems* 33 (2020), pp. 22243–22255.

[8] Xinlei Chen and Kaiming He. “Exploring simple siamese representation learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 15750–15758.

[9] Xinlei Chen, Saining Xie, and Kaiming He. “An empirical study of training self-supervised vision transformers”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 9640–9649.

[10] Xinlei Chen et al. “Improved baselines with momentum contrastive learning”. In: *arXiv preprint arXiv:2003.04297* (2020).

[11] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[12] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[13] Carl Doersch, Abhinav Gupta, and Alexei A Efros. “Unsupervised visual representation learning by context prediction”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1422–1430.

[14] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).

[15] Aleksandr Ermolov et al. “Whitening for self-supervised representation learning”. In: *International conference on machine learning*. PMLR. 2021, pp. 3015–3024.

[16] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. “Unsupervised representation learning by predicting image rotations”. In: *arXiv preprint arXiv:1803.07728* (2018).

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[18] Jean-Bastien Grill et al. “Bootstrap your own latent-a new approach to self-supervised learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 21271–21284.

[19] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[20] Kaiming He et al. “Masked autoencoders are scalable vision learners”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16000–16009.

[21] Kaiming He et al. “Momentum contrast for unsupervised visual representation learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9729–9738.

[22] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5 (1989), pp. 359–366.

[23] Rui Jin and Qiang Niu. “Automatic fabric defect detection based on an improved YOLOv5”. In: *Mathematical Problems in Engineering* 2021 (2021), pp. 1–13.

[24] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[25] Warren S McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *The bulletin of mathematical biophysics* 5 (1943), pp. 115–133.

[26] Mehdi Noroozi and Paolo Favaro. “Unsupervised learning of visual representations by solving jigsaw puzzles”. In: *European conference on computer vision*. Springer. 2016, pp. 69–84.

[27] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. “Representation learning with contrastive predictive coding”. In: *arXiv preprint arXiv:1807.03748* (2018).

[28] Utku Ozbulak et al. “Know your self-supervised learning: A survey on image-based generative and discriminative training”. In: *arXiv preprint arXiv:2305.13689* (2023).

[29] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[30] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536.

[31] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[32] Jure Zbontar et al. “Barlow twins: Self-supervised learning via redundancy reduction”. In: *International conference on machine learning*. PMLR. 2021, pp. 12310–12320.