UNDER REVIEW

RESCIENCE C

Replication / ML Reproducibility Challenge 2022

[Re] Semantic Masked Autoencoders

Anonymous[1, ID]

[1]Anonymous Institution

Edited by
(Editor)

Reviewed by
(Reviewer 1)
(Reviewer 2)

Received
03 February 2023

Published
–

DOI
–

Reproducibility Summary

Scope of Reproducibility – The Masked Autoencoder (MAE) was recently proposed as a framework for efficient self-supervised pre-training in Computer Vision [1]. In this paper, we attempt a replication of the MAE under significant computational constraints. Specifically, we target the claim that masking out a large part of the input image yields a nontrivial and meaningful self-supervisory task, which allows training models that generalize well. We also present the Semantic Masked Autoencoder (SMAE), a novel yet simple extension of MAE which uses perceptual loss to improve encoder embeddings.

Methodology – The datasets and backbones we rely on are significantly smaller than those used by [1]. Our main experiments are performed on Tiny ImageNet (TIN) [2] and transfer learning is performed on a low-resolution version of CUB-200-2011 [3]. We use a ViT-Lite [4] as the backbone. We also compare the MAE to DINO, an alternative framework for self-supervised learning [5]. We implemented the ViT, MAE as well as perceptual loss from scratch. For DINO, we used the official published code [5]. Similarly, for our perceptual loss, we used the SqueezeNet model from Torchvision [6]. The computational budget for our reproduction and extension was approximately 150 GPU hours.

Results – This paper successfully reproduces the claim that the MAE poses a nontrivial and meaningful self-supervisory task. We show that models trained with this framework generalize well to new datasets and conclude that the MAE is reproducible with exception for some hyperparameter choices. Our results also suggest that MAE performs well with smaller backbones and datasets. Finally, we demonstrate that the SMAE extension improves the downstream classification accuracy of the MAE when transferred to CUB (+5.0 pp) when coupled with an appropriate masking strategy.

What was easy – The work of re-implementing the paper was relatively straightforward, with sufficient details given in the paper. In addition, implementing the Perceptual Loss was another aspect we found uncomplicated.

What was difficult – We temporarily struggled with the details of implementing random shuffling and unshuffling of masked/non-masked patches correctly and efficiently. A significant amount of time was spent tuning hyperparameters, as the choices of [1] turned out to be not completely applicable to a smaller dataset and backbone.

Communication with original authors – We have not had contact with the original authors.

Copyright © 2023 Anonymous, released under a Creative Commons Attribution 4.0 International license.
Correspondence should be addressed to ()
The authors have declared that no competing interests exists.
Code is available at https://anonymous.4open.science/r/SMAE-0CF8.

1 Introduction

As computational capabilities increase, deep learning models for computer vision (CV) grow to the point where access to labeled data becomes the performance bottleneck. The introduction of BERT in 2018 enabled effective and scalable *self-supervised* pre-training in Natural Language Processing (NLP) through *masked autoencoding* with transformers [7]. Adapting the masked autoencoding scheme to the image domain posed two main problems: 1) architectural differences between convolutional neural networks and transformers, 2) much lower information density in images than in written language. Dosovitskiy et al. addressed the former in 2020 with the introduction of the Vision Transformer (ViT) [8]. The difference in information density would remain a challenge until He et al. proposed the Masked Autoencoder (MAE) in 2022 [1]. By masking out a large part of the input and only encoding the visible parts, the MAE managed to perform efficient and effective self-supervised pre-training with images while keeping its design exceptionally simple [1].

The MAE uses pixel-wise mean squared error (MSE) as the loss function during pre-training [1]. However, it has been shown that loss functions that promote accurate reconstructions do not necessarily lead to useful representations when transferring to downstream tasks, suggesting that pixel-wise reconstruction error might be a flawed metric for measuring the quality of latent representations [9, 10]. Perceptual loss has been proposed as an alternative to pixel reconstruction loss for autoencoders and has shown improved performance in downstream tasks such as image classification and object localization [9].

In this paper, we attempt a reproduction of [1] under significant computational constraints. We also present the Semantic Masked Autoencoder (SMAE), a novel and simple extension of the MAE which uses perceptual loss to improve the autoencoder embeddings. The datasets and backbone we rely on are significantly smaller than those used by [1]. Our main experiments are performed on Tiny ImageNet (TIN) [2] and transfer learning is performed on a low-resolution version of CUB-200-2011 [3]. As backbone, we use a ViT-Lite [4] with 3.72M parameters, thus being two orders of magnitude smaller than ViT-Large (307M parameters) which was used as baseline in [1]. We also compare the MAE and SMAE to DINO, an alternative framework for self-supervised learning [5].

The contributions of this paper can be summarized as follows:

- We reproduce the results of [1] at a much lower scale. Through ablation studies, we settle on the same masking ratio and masking strategy as in [1]. We demonstrate favorable performance with MAE compared to supervised learning and a comparable SSL method (DINO). Models pre-trained with MAE are also shown to generalize well when transferred to another dataset.
- We demonstrate that the proposed SMAE extension improves the transfer performance of the MAE when paired with an appropriate masking strategy.

1.1 Scope of reproducibility

This paper aims to reproduce a subset of the main claims made by He et al. in the paper "Masked Autoencoders Are Scalable Vision Learners" [1]. The main claims made by He et al. are:

1. Masking out a large part of the input image uniformly at random yields a nontrivial and meaningful self-supervisory task.
2. Masked autoencoding allows for learning models that generalize well.

3. The MAE is a scalable and efficient method in the sense that pre-training larger models is tractable and improves performance without overfitting to training data.

With respect to our restricted computational budget, **we set out to reproduce claims 1 and 2**. We aim to reproduce claim 1 by 1) ablating the masking ratio, 2) ablating the masking method, 3) pre-training with MAE and fine-tuning on TIN. We aim to reproduce claim 2 by pre-training with MAE on TIN and transferring the encoder to CUB. By further developing the idea of masked autoencoding through the use of perceptual loss in our proposed SMAE, we aim to provide additional evidence in support of claim 2.

2 Background

This section presents the relevant background for the SMAE, our extension of the MAE which uses perceptual similarity to improve the autoencoder embeddings.

2.1 Perceptual Similarity

Perceptual Similarity is a way of measuring the distance between two images. Originally proposed by Zhang et al. [11], this method relies on convolutional neural networks' ability to extract semantic representations of images. This leads to a metric that is more consistent with the human visual system, as it promotes closeness of high-level structures. When used as a loss function to train autoencoding models, *perceptual loss* enables learning robust representations of the input images, as opposed to pixel-space loss functions that encourage color reproduction [9]. Employing perceptual loss has also been observed to significantly improve the usefulness of embeddings when training autoencoders in an adversarial setting [10].

2.2 Semantic Masked Image Modelling

Masked Image Modelling (MIM) is a method for self-supervised learning in CV, which involves masking parts of the input images and training models on this partial information. BEiT [12] is a recent method that reconstructs discretized tokens from masked images; taking inspiration from BERT pre-training in NLP [7]. Semantic Masked Image Modelling (SMIM) brings perceptual loss to MIM. SMIM is an approach to improving the usefulness of embeddings when performing MIM, which has become a trend as of the last year. PeCo, a concurrent work to ours, is one such technique; extending the BEiT method by using perceptual similarity as loss function when learning the visual words used as MIM targets. PeCo demonstrates state-of-the-art performance on ImageNet-1K image classification [13]. BootMAE is another SMIM method. It is an extension of the MAE which uses a temporal ensemble of an MAE model as a perceptual critic during pre-training [14]. Compared to BootMAE, our SMAE has the advantage of being exceedingly simple in its design.

3 Methodology

We re-implemented the MAE from the description provided in the paper without consulting the author's published code. We implemented the ViT model and perceptual loss from scratch. For DINO, we used the official published code [5] and our perceptual critic implements the SqueezeNet model from Torchvision [6]. The following sections outline our methodology; detailing the models, datasets, and hyperparameters as well as the experimental setup and computational resources used.

3.1 Masked Autoencoder

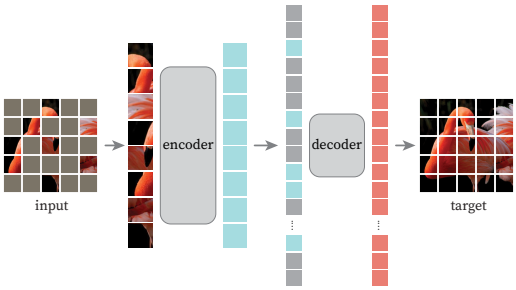

Figure 1. The MAE architecture. A large portion of the input image is masked and only non-masked patches are encoded. Mask tokens are injected into the encoded sequence before decoding, resulting in a computationally efficient MIM scheme with a non-trivial reconstruction task. Figure from [1].

The MAE is an autoencoder-type framework for MIM. A large portion of the image is masked before being fed to an encoder, consisting of a ViT, which creates a latent representation of the unmasked input. The decoder, which is a lightweight ViT, reconstructs the entire image from the latent representation created by the encoder (see Figure 1). After pre-training on unlabeled data, the decoder is discarded, as the main interest lies in transferring the encoder to downstream tasks. Regarding the reconstruction target, [1] found that downstream performance was improved by: 1) only computing the loss for masked patches, and 2) patch-wise normalizing the target pixel values.

3.2 Masking

The MAE stands out in that it masks a large portion of the input image. He et al. argue that this rules out solving the task by simply extrapolating from nearby unmasked patches and motivates the encoder to learn meaningful representations of the input [1]. We therefore ablate the ratio of masked patches in our experiments.

The findings of [1] suggest that uniform random masking of patches is the most effective masking strategy. However, [14] found that masking large random blocks was more effective when using perceptual loss; implying that reconstructing large blocks is a difficult task for pixel regression while being helpful for a perceptual model in reasoning about the semantic structure. We thus decided to ablate the masking strategy for all of our models.

3.3 Semantic Masked Autoencoder

We propose the SMAE, which extends the MAE by incorporating perceptual loss [11]. Instead of only reconstructing pixels, the SMAE objective is to minimize a combination of pixel-wise loss $\mathcal{L}_{pixel}$ and perceptual loss $\mathcal{L}_{percep}$:

$$\mathcal{L}_{SMAE} = (1-\alpha)\,\mathcal{L}_{pixel} + \alpha\mathcal{L}_{percep} \tag{1}$$

where $\alpha = 0$ reduces to the MAE objective and $\alpha = 1$ implies using only perceptual loss. Our implementation of perceptual loss largely follows that of [9]. To avoid using a critic pre-trained on larger datasets, we train a SqueezeNet v1.1 [15, 6] from scratch on TIN and use it as a critic in the perceptual loss. The original image and the reconstruction are fed to the critic, from which the learned representations are extracted. In order to preserve spatial information, the representations are extracted at an early stage in the critic network, as argued for in [9]. Finally, the perceptual loss is computed as the MSE

between the representation of the original image and the reconstruction. Formally, let f denote the MAE and g denote the SqueezeNet v1.1 up until and including the second *Fire* module. The perceptual loss between a sample x and it's reconstruction $f(x)$ can then be formulated as:

$$\mathcal{L}_{percep} = \frac{c}{d} \sum_{i}^{d} (g(x) - g(f(x)))^2 \tag{2}$$

where d is the dimensionality of the extracted representations and c is a scaling factor used to ensure $\mathcal{L}_{percep}$ and $\mathcal{L}_{pixel}$ are of similar scale. In practice, we set c before optimization, to the ratio between the initial pixel-wise and perceptual loss values. We tried using an uncertainty-based weighting of the losses [16], but found that using a mixing coefficient and scaling by a constant factor performed best.

3.4 Datasets

Tiny ImageNet — In [1] the pre-training was done on IN1K [17]. In order to use the same data distribution while honoring our computational constraints, we pre-train on TIN [2]; a smaller dataset containing 100 000 labeled training images from IN1K scaled down to a size of 64×64. There are 200 distinct classes, each containing 500 examples. Additionally, there is a validation set containing 10 000 examples. We use a crowd-sourced version of Tiny ImageNet from Hugging Face available at https://huggingface.co/datasets/Maysee/tiny-imagenet. Literature where training is performed on TIN is quite sparse. The current SOTA classification performance on TIN among methods trained only on TIN data is 72.39%, achieved with a ResNeXt backbone trained with decoupled scenario-agnostic mixup loss [18].

CUB-200-2011 — We performed transfer experiments on a scaled-down (64×64) version of CUB-200-2011 (CUB) containing 11 788 images of birds belonging to 200 classes [3]. Out of the 11 788 images, 5 994 are for training and 5 794 are used for testing. The dataset is available at https://data.caltech.edu/records/65de6-vp158. We chose CUB because it is a challenging dataset for supervised learning methods due to having very few examples per class and an uneven class distribution. A common role for CUB is as a benchmark for few-shot learning techniques.

3.5 Backbone Architecture

Due to our computational constraints, we chose to replicate the method of [1] using a smaller backbone. The encoder backbone that we used was ViT-Lite proposed by Hassani et al. [4]. ViT-Lite has 3.72M parameters, thus being two orders of magnitude smaller than ViT-Large (307M) which was the baseline used in [1]. ViT-Lite has 7 transformer blocks with a dimensionality of 256 and an MLP dimensionality of 512. The multi-head self-attention uses 4 heads.

We aimed to maintain the ratio between the size of patches and the image size from [1]. Therefore, we chose to reduce the patch size from 16 to 4, accounting for the smaller images of TIN. This would have made it even more computationally expensive to choose a larger backbone than we did, since its complexity grows quadratically with sequence length.

3.6 Hyperparameters

As part of our reproduction efforts, we strived to stick as close as possible to the settings used for training in [1]. Even so, we had to adjust some parameters due to our

smaller datasets and backbone. We employed a grid search strategy over learning rate (lr), weight decay (wd) and number of layers in the decoder (dd). This resulted in a pre-training setup of $lr = 5\text{e}{-4}$, $wd = 0.15$ and $dd = 2$ (see Appendix D). Differently from [1], we found that a shallower decoder was beneficial. Our decoder had a dimensionality of 128 and an MLP dimensionality of 256.

We did not use any data augmentation during pre-training. We tried using random cropping as used by [1], but observed that pre-training without augmentations performed better. As for the reconstruction target, we used raw pixel values; having found this to perform better than patch-wise normalized pixel values as used by [1]. The hyperparameters for DINO and SqueezeNet are deferred to Appendix B and C. We remark that due to our limited computational resources, the hyperparameter search for DINO and SqueezeNet was not as thorough as for MAE. Finally, for the SMAE mixing coefficient, we found $\alpha = 0.5$ to be a good choice. Results for different values of α are presented in Appendix E.

3.7 Experimental setup and code

We used a batch size of 128 in all our experiments. During the hyperparameter search, we pre-trained for 200 epochs and performed linear probing for 50 epochs. Our final models were pre-trained for 400 epochs and fine-tuned for 100 epochs. Supervised training from scratch was done for 400 epochs. While linear-probing, we froze the backbone and trained an added linear classification head. During fine-tuning, we jointly trained the backbone and an added linear classification head. All experiments were evaluated using the Top-1 validation set accuracy. The MAE was only trained to reconstruct masked patches, whereas the SMAE reconstructed all patches; this was done in order to avoid conflicts of interest between the pixel-wise loss and the perceptual loss. If the pixel-wise loss is minimized for masked patches only, the discontinuity between masked and unmasked patches might increase, consequently increasing the perceptual loss which operates on the entire reconstruction. When using block masking, we masked 50% of the input image, following that of [1]. Due to our restricted computational budget, all experiments were performed once. As such, our results should be seen as indicative rather than conclusive. Our code is written in PyTorch and is publicly available at https://gits-15.sys.kth.se/simonevh/PMAE.

3.8 Computational requirements

The experiments were performed locally on an NVIDIA GTX 1080 Ti, an RTX 3060 and an RTX 2070, as well as on Google Compute Engine using V100s. Pre-training with MAE for 400 epochs on TIN took roughly 9 hours on a V100, while fine-tuning ViT-Lite for 100 epochs on TIN took around 3 hours on the same hardware. The total computational budget for our reproduction and extension was approximately 150 GPU hours. Details on runtimes are provided in appendix A.

4 Results

All results presented in this section support the main claims of [1]. Our results on masking ratio and masking method support claim 1. Our pre-trained models performed favorably to supervised learning as well as DINO on both TIN and CUB, supporting both claim 1 and 2. The SMAE extension significantly improved the performance of the MAE on transfer learning, providing further evidence in support of claim 2.

Figure 2. Reconstructions during MAE pre-training. The masking ratio is 75% for random masking (left) and 50% for block masking (middle).

4.1 Results reproducing original paper

Masking — We ablated the effect of masking ratio on the usefulness of the autoencoder embeddings when using uniform random masking.

Masking ratio	50%	60%	65%	70%	75%	80%	85%	90%
Linear probing valid. acc.	31.32	32.03	**32.32**	30.58	**32.04**	31.66	29.61	28.14

The results somewhat reflect those of [1]; suggesting that masking 65% of the input creates the most useful embeddings. Masking 75% of the input resulted in similar linear probing accuracy, but was 17% faster to train. As such, we used 75% masking for the rest of our experiments with uniform random masking, the same value that [1] used. Overall, the experiment successfully reproduces that of [1] and the results support claim 1. We also ablated the masking strategy of [1], reproducing its findings that uniform random masking creates more useful embeddings than block masking; as seen by the fine-tuning validation accuracy on both TIN and CUB:

		TIN	CUB
Masking	Random	**55.00**	**42.54**
	Block	48.77	33.50

In Figure 2 we present example reconstructions from our pre-trained MAE under different masking strategies.

Fine-Tuning on TIN — We successfully replicated the fine-tuning experiment from [1]. Our results show that fine-tuning a model which was pre-trained using MAE outperformed both the same backbone pre-trained with DINO as well as the same ViT-Lite trained from scratch using supervised training (see Table 1). The results from our experiments on fine-tuning on TIN support claim 1, that the MAE method yields a nontrivial and meaningful self-supervisory task.

Transferring to CUB — We transferred the pre-trained MAE by fine-tuning on unbalanced image classification on CUB. The results are compared to both a ViT-Lite pre-trained with DINO and a ViT-Lite trained from scratch (see Table 1). The pre-trained MAE outperforms both methods, reproducing the findings of [1]. Thus, our results on transfer learning suggest that representations learnt with MAE transfer well to new datasets, providing support for claim 2.

Method	TIN	CUB
Supervised	48.58	19.70
DINO	41.72	24.09
MAE	**55.00**	42.54
SMAE ($a = 0.5$)	54.40	**47.50**

Table 1. Top-1 validation accuracy on image classification datasets.

4.2 Results beyond original paper

This section includes the results for SMAE, our extension of the MAE.

SMAE — The results from our proposed SMAE are presented in Table 1. The SMAE performed similarly to the MAE when fine-tuned on TIN. When transferred to CUB, our extension improved the performance of the MAE by a large margin (+5.0 pp), providing further evidence in support of claim 2. Regarding the masking method for SMAE, the choice between random masking and block masking did not have a significant impact on performance when fine-tuning on TIN. When transferred to CUB, SMAE displayed significantly greater performance with block masking than random masking (+6.0 pp) in terms of validation set accuracy:

		TIN	CUB
Masking	Random	**54.40**	41.50
	Block	53.90	**47.50**

5 Discussion

Our experimental results support the main claims of [1] that we set out to reproduce, i.e. claim 1 and 2. Due to our significant computational constraints, we have not attempted to reproduce claim 3. We conclude that the MAE framework is reproducible in general, and that it also performs well with smaller datasets and backbones. A weakness of our reproducibility approach is the comparison between MAE and DINO. Due to our limited resources and the high computational demand of DINO, we could not perform a proper grid search for it. Provided this, our comparison between MAE and DINO is not entirely fair.

We have also presented the SMAE, an extension to the MAE that incorporates recent ideas about using perceptual loss to improve usefulness of embeddings. Our results suggest that the SMAE learns semantically meaningful representations that are more useful than those of the MAE when transferring to another dataset.

The authors of BootMAE [14] observed that training with perceptual loss benefits from masking out larger connected blocks. Our findings for transferring to CUB align with this, as we see a big performance increase when going from random masking to block masking with the SMAE. In contrast, fine-tuning on TIN did not express any meaningful performance difference with respect to the masking method, suggesting that the choice of masking method might be more important when transferring to a different data distribution. Our results with SMAE motivate carrying out more in-depth studies investigating whether the SMAE would scale to the same datasets and backbones as used in [1]; something that was out-of-scope for this study due to our limited computational budget.

5.1 What was easy

Given prior experience with a deep learning framework, re-implementing the paper was relatively straight-forward, with exception for the random shuffling and un-shuffling (see Section 5.2). The paper provided sufficient details on the MAE method, including training configurations and implementational details.

5.2 What was difficult

It was not trivial to apply the approach in the paper on a different choice of dataset and backbone, as the paper's choices of hyperparameters turned out to require significant

recalibration. Therefore, we had to spend quite some time tuning hyperparameters. We temporarily struggled with the details of implementing random shuffling and un-shuffling of masked/non-masked patches correctly *and* efficiently. Our final implementation for the shuffling and un-shuffling operations used the scatter and gather functions in PyTorch, which are fairly involved operations.

5.3 Communication with original authors

We have not had contact with the original authors.

References

1. K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. "Masked autoencoders are scalable vision learners." In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. 2022, pp. 16000–16009.
2. Y. Le and X. Yang. "Tiny imagenet visual recognition challenge." In: **CS 231N** 7.7 (2015), p. 3.
3. C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. "The caltech-ucsd birds-200-2011 dataset." In: (2011).
4. A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi. "Escaping the big data paradigm with compact transformers." In: **arXiv preprint arXiv:2104.05704** (2021).
5. M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. "Emerging properties in self-supervised vision transformers." In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. 2021, pp. 9650–9660.
6. S. Marcel and Y. Rodriguez. "Torchvision the machine-vision package of torch." In: **Proceedings of the 18th ACM international conference on Multimedia**. 2010, pp. 1485–1488.
7. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**. Minneapolis, Minnesota, June 2019, pp. 4171–4186.
8. A. Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In: **International Conference on Learning Representations**. 2021.
9. G. G. Pihlgren, F. Sandin, and M. Liwicki. "Improving image autoencoder embeddings with perceptual loss." In: **2020 International Joint Conference on Neural Networks (IJCNN)**. IEEE. 2020, pp. 1–7.
10. A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. "Autoencoding beyond pixels using a learned similarity metric." In: **International conference on machine learning**. PMLR. 2016, pp. 1558–1566.
11. R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. "The unreasonable effectiveness of deep features as a perceptual metric." In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2018, pp. 586–595.
12. H. Bao, L. Dong, S. Piao, and F. Wei. "BEiT: BERT Pre-Training of Image Transformers." In: **International Conference on Learning Representations**. 2022.
13. X. Dong, J. Bao, T. Zhang, D. Chen, W. Zhang, L. Yuan, D. Chen, F. Wen, and N. Yu. "Peco: Perceptual codebook for bert pre-training of vision transformers." In: **arXiv preprint arXiv:2111.12710** (2021).
14. X. Dong, J. Bao, T. Zhang, D. Chen, W. Zhang, L. Yuan, D. Chen, F. Wen, and N. Yu. "Bootstrapped Masked Autoencoders for Vision BERT Pretraining." In: **European Conference on Computer Vision**. Springer. 2022, pp. 247–264.
15. F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size." In: **arXiv preprint arXiv:1602.07360** (2016).
16. A. Kendall, Y. Gal, and R. Cipolla. "Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics." In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. 2018, pp. 7482–7491.
17. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "Imagenet: A large-scale hierarchical image database." In: **2009 IEEE conference on computer vision and pattern recognition**. Ieee. 2009, pp. 248–255.
18. Z. Liu, S. Li, G. Wang, C. Tan, L. Wu, and S. Z. Li. "Decoupled Mixup for Data-efficient Learning." In: **arXiv preprint arXiv:2203.10761** (2022).
19. P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. "Accurate, large minibatch sgd: Training imagenet in 1 hour." In: **arXiv preprint arXiv:1706.02677** (2017).

20. I. Loshchilov and F. Hutter. "Decoupled Weight Decay Regularization." In: **International Conference on Learning Representations**. 2019.
21. M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever. "Generative pretraining from pixels." In: **International conference on machine learning**. PMLR. 2020, pp. 1691–1703.
22. I. Loshchilov and F. Hutter. "Stochastic gradient descent with warm restarts." In: **Proceedings of the 5th Int. Conf. Learning Representations**, pp. 1–16.
23. E. D. Cubuk, B. Zoph, J. Shlens, and R. Le QV. "Practical automated data augmentation with a reduced search space." In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops**, pp. 702–703.
24. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. "Attention is all you need." In: **Advances in neural information processing systems** 30 (2017).
25. D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization." In: **ICLR (Poster)**. Ed. by Y. Bengio and Y. LeCun. 2015.

A Runtimes

Table 2 reports the runtimes for our models on the TIN validation set without gradient computations. The SMAE does not increases the runtime significantly over the MAE.

Model	Task	Time (s)
ViT-Lite	Classification	11
MAE	Reconstruction	5
SMAE	Reconstruction	5

Table 2. The runtime of various models for the TIN validation set of 10 000 images. The runtimes are reported for a batch size of 128 on a GTX 1080 Ti with no gradient computations.

B Hyperparameters for DINO

A non-exhaustive manual search was performed for learning rate ($5e^{-4}$, $1e^{-4}$) and last layer normalization. We also manually searched the momentum (0.9995, 0.9998, 0.9960), warm-up epochs (0, 30) and temperature (0.02, 0.04) for the teacher. Due to computational constraints, we had to disable multi-crop. The final hyperparameters for DINO are presented in Table 3.

config	value
base learning rate	$1e{-}4$
min learning rate	$1e{-}6$
weight decay	0.04
max weight decay	0.4
teacher momentum	0.9995
batch size	128
warmup epochs [19]	10
training epochs	100
out dim	1024
hidden dim	512
bottleneck dim	128
local crops number	0
teacher warmup temperature	0.02
teacher temperature	0.04
teacher warmup episodes	30
normalize last layer	✓
gradient clipping	3
global crops scale	(0.6, 1)

Table 3. Hyperparameters for DINO.

C Hyperparameters for SqueezeNet

We performed a manual search for learning rate ($5e^{-4}$, $1e^{-3}$, $4e^{-3}$) and weight decay (0.05, 0.15, 0.1). The final parameters are presented in Table 4.

config	value
optimizer	AdamW [20]
base learning rate	$\mathbf{1e^{-3}}$
weight decay	0.15
optimizer momentum (β_1, β_2)	$\mathbf{0.9, 0.999}$ [21]
batch size	128
learning rate schedule	cosine decay [22]
warmup epochs [19]	10
training epochs	200
augmentation	RandAug (2, 9) [23]
dropout rate	0.5

Table 4. Hyperparameters for SqueezeNet.

D Hyperparameter Grid Search for MAE

In order to find appropriate values for MAE hyperparameters we employed a grid search approach. As explained in subsection 3.6, the MAE encoder architecture follows that of ViT-Lite-7/4 [4]. We employ sinusoidal position encoding [24]. The MAE decoder architecture follows the encoder but is lighter, according to [1]. We fix the decoder width to 128 and the number of nodes in its attention layers to 256 throughout all our experiments.

During the grid search, we pre-trained the ViT-Lite backbone for 200 epochs and then linearly-probed it for 50, both on TIN. The pre-training AdamW optimizer weight decay (wd) and learning rate (lr) were searched in the set $\{0.05, 0.15\}$ and $\{1e{-}3, 1.5e{-}4, 5e{-}4\}$ respectively. In addition, we repeated this search for two different decoder depths (i.e. number of layers in the decoder), $\{2, 3\}$. In Table 5 we present the validation set accuracy for different hyperparameter settings.

	wd=0.05	wd=0.15
lr=$1e-3$	28.07	-
lr=$1.5e-4$	13.75	27.15
lr=$5e-4$	11.89	32.04

$dd = 2$

	wd=0.05	wd=0.15
lr=$1.5e-4$	13.75	27.15
lr=$5e-4$	11.89	32.04

$dd = 3$

Table 5. Grid search results. Each sub-table contains the values for a single decoder depth; 2 (left) and 3 (right). Each cell contains the top-1 validation accuracy for linear probing of a model pre-trained under the corresponding optimizer settings.

Next, we enclose the training hyperparameters we use for each training task on the TIN dataset. When fine-tuning on CUB we use the same hyperparameters as on TIN, but halve the learning to account for the difference in dataset sizes.

E Perceptual loss (SMAE with $\alpha = 1$)

We observe that when applying uniform random masking using only perceptual loss (SMAE with $\alpha = 1$), the downstream classification performance notably decreases. When instead using block masking, we observe a notable performance improvement, especially when we transfer to CUB. The results of the corresponding runs are given in Table 7 below.

config	value		
	Pre-Training	Fine-Tuning	Linear-Probing
optimizer	AdamW [20]	AdamW	Adam [25]
base learning rate	5e-4	1e-3	1e-3
weight decay	0.15	0.05	-
optimizer momentum (β_1, β_2)	$0.9, 0.95$ [21]	$0.9, 0.999$	$0.9, 0.95$
batch size	128	128	128
learning rate schedule	cosine decay [22]	cosine decay	cosine decay
warmup epochs [19]	20	5	2
training epochs	400	100	50
augmentation	-	RandAug (2, 9) [23]	-

Table 6. Training settings for the different training tasks on TIN.

	TIN	CUB
random	**52.70**	39.67
block	51.05	**42.90**

Table 7. Top-1 validation accuracy on the 2 image-classification tasks. The models were pre-training on TIN by using perceptual loss only, i.e. SMAE loss with a=1.