Are Large-scale Datasets Necessary for Self-Supervised Pre-training?

Alaaeldin El-Nouby*[1,2] Gautier Izacard*[1,2] Hugo Touvron[1,3] Ivan Laptev[2]
Hervé Jégou[1] Edouard Grave[1]

[1]Facebook AI Research [2]Inria [3]Sorbonne University

Abstract

Pre-training models on large scale datasets, like ImageNet, is a standard practice in computer vision. This paradigm is especially effective for tasks with small training sets, for which high-capacity models tend to overfit. In this work, we consider a self-supervised pre-training scenario that only leverages the target task data. We consider datasets, like Stanford Cars, Sketch or COCO, which are order(s) of magnitude smaller than Imagenet.

Our study shows that denoising autoencoders, such as BEiT or a variant that we introduce in this paper, are more robust to the type and size of the pre-training data than popular self-supervised methods trained by comparing image embeddings. We obtain competitive performance compared to ImageNet pre-training on a variety of classification datasets, from different domains. On COCO, ***when pre-training solely using COCO images****, the detection and instance segmentation performance surpasses the supervised ImageNet pre-training in a comparable setting.*

1. Introduction

Modern computer vision neural networks are heavily parametrized: they routinely have tens or hundreds of millions of parameters [1, 2, 3, 4]. This has been the key to their success for leveraging large-scale image collections such as ImageNet. However, these high capacity models tend to overfit on small, or even medium sized datasets consisting of hundreds of thousands of images. In order to circumvent this issue, a standard approach in computer vision is to pre-train models on large datasets, and transfer these models to the target tasks [5]. This is usually performed by finetuning the weights of the models using a shorter optimization procedure than the one employed when training from scratch (or more precisely from randomly generated weights).

Hence, the most common pre-training method for computer vision applications is to learn a supervised model using the ImageNet dataset [6]. This simple approach has led to impressive results, which are state-of-the-art in many tasks such as detection [7, 8], segmentation [9] and action recognition [10]. Despite this success, we point out that it is difficult to disentangle the benefits offered by such a large-scale curated label dataset from the limitations of this pre-training paradigm. Putting aside the discussion on the collection effort (cost, requiring in-domain expertise, etc), we point out that pre-training a model on a dataset and fine-tuning it on another can introduce two sort of discrepancies.

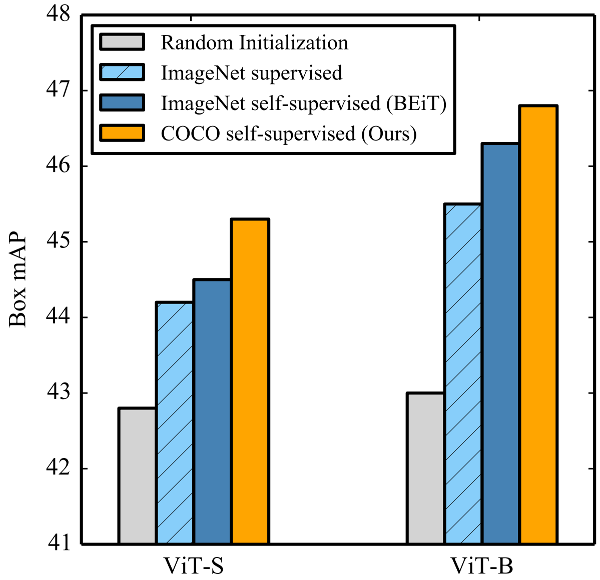

Figure 1. We demonstrate that self-supervised pre-training using denoising autoencoders like BEiT and our variant SplitMask are more robust to the type and/or size of pre-training data used. For example, the object detection performance of such models, when pre-trained only using COCO images and a Mask R-CNN pipeline, outperforms both supervised and BEiT self-supervised baselines pre-trained on ImageNet, as well as a randomly initialized baseline trained for a long schedule.

First, this setting introduces a domain shift between the images used to pre-train the model and those targeted

*equal contribution

by the fine-tuning stage. Imagenet images may be sufficiently representative of natural images (despite the collecting bias). They may not suffice when transferring a pre-trained model to out-of-domain distributions such as sketches, painting, clipart or data captured by specialized devices like those employed in medical imaging or astronomy. To date, most researchers consider that the benefit of having a large amount of images vastly compensates the domain discrepancy on benchmarks involving natural images, such as the fine-grained iNaturalist datasets [11, 12].

The second question, discussed by Doersch *et al.* [13], is the so-called *supervision collapse*. This phenomenon is inherent to pre-training with a fixed set of labels: the network learns to focus on the mapping between images and the labels of the pre-training stage, but can discard information that is relevant to other downstream tasks. In other terms, pre-training on large-scale classification datasets does not necessarily align with the goal of learning general-purpose features, as it uses only a subset of the available information controlled by the given dataset categorization bias [14].

These limitations have motivated the development of self-supervised pre-training methods which learn directly from data, without relying on annotations. Most notably, the contrastive and joint embedding approaches [15, 16, 17, 18, 19] can serve as effective pre-training strategies. While obtained a strong performance on many tasks, such methods have a strong bias towards ImageNet data since the transformations have been hand-designed to perform well on the ImageNet benchmark. Some of the most effective transformations, like cropping, rely on the images being object centric [20]. When applied on uncurated data, these methods degrade significantly and require larger datasets to obtain similar performance [21].

This is in contrast with natural language processing, where nowadays, most applications use large models which were pre-trained on uncurated data. In particular, the (masked) language modeling loss has been applied to transformer networks, leading to the BERT model [22], which is now the foundation of most NLP models. Inspired by this success, Bao et al. [23] have shown the potential of the Masked Image Modeling (MIM) task to pre-train vision transformers. Such model can be thought of as a denoising autoencoder [24] where the noise corresponds to the patch masking operation. This technique has been successfully applied to ImageNet, but research questions remain: (1) How much does this pre-training technique rely on the number of pre-training samples, and in particular, does it require millions of images to be useful? (2) Is this technique robust to different distributions of training images? In particular, can it be an effective learning paradigm when training on non object-centric or uncurated images? If the answer to both questions is positive, it will enable pre-training using a larger variety of datasets, including the training sets of many tasks that are smaller or belong to a different domain than ImageNet.

In this work, we make the following contributions:

- First, we demonstrate that denoising autoencoders are more sample efficient than joint embedding techniques, enabling pre-training without relying on large-scale datasets (e.g. ImageNet);
- Second, as a consequence of the better sample efficiency, we show on multiple datasets that it is possible to pre-train directly on the target task data and obtain a competitive performance, even with datasets that are orders of magnitude smaller than ImageNet.
- Third, we demonstrate that denoising autoencoders can be successfully applied to non object-centric images such as COCO, achieving performance similar to the one obtained when pre-training with ImageNet, unlike joint embedding techniques which seem to suffer a drop in performance;

2. Related Work

In this section, we briefly review some previous work on self-supervised learning, including autoencoders and instance discrimination methods.

Pre-training with autoencoders has a long history in deep learning, where it was initially used as a greedy layer-wise method to improve optimization [24, 25, 26, 27, 28]. In the context of unsupervised feature learning for image classification, different tasks related to denoising autoencoders have been considered, such as in-painting [29], colorization [30] or de-shuffling of image patches [31]. In natural language processing, denoising autoencoders have been applied by masking or randomly replacing some tokens of the input, and reconstructing the original sequence, leading to the BERT model [22]. Similar methods have been proposed to pre-train sequence-to-sequence models, by considering additional kind of noises such as word shuffling or deleting [32, 33]. There has been efforts to adopt such successful ideas in NLP to computer vision, but with limited success. Chen et al. [34] proposed iGPT, a transformer-based autoregressive model that operates over image pixels, while Atito et al. [35] trained a ViT model on denoising of images where the noise is applied at pixel level. More recently, Bao et al. [23] introduced the Masked Image Modeling loss in computer vision, where image patches are masked, and the goal is to predict the discretized label of the missing patches corresponding to their visual words according to a pre-trained discrete VAE [36].

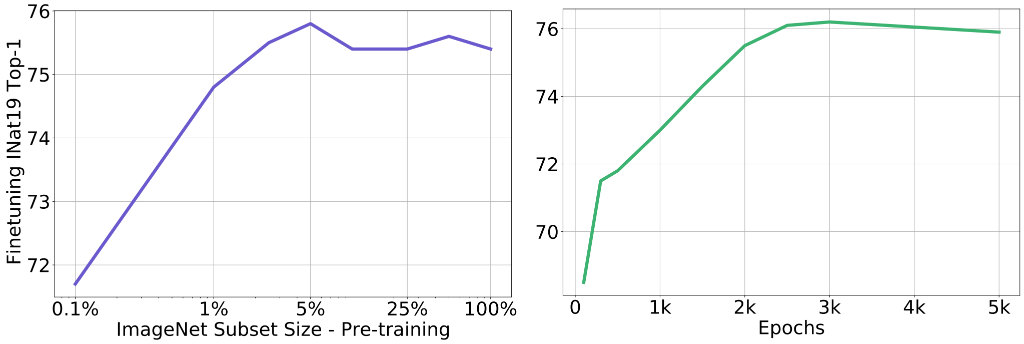

Figure 2. (Left) Pre-training using different subsets of ImageNet. Performance is stable even when using a subset as small as 10% when trained for the same number of iterations of 60k. (Right) Varying the number of pre-training epochs for the 10% subset. We observe a consistent increase in performance with longer training.

Instance discrimination is a set of self-supervised techniques which consider that each image corresponds to its own class [37, 38]. A set of data augmentations (or transformations) is then applied to each image to generate multiple examples for each class. The global image representations are trained in a contrastive framework, typically using the InfoNCE loss [39], to have high similarity for instances transformed from the same source image and low similarity with all other images. As the performance of these methods depends on the number of negatives, it either requires large batches or memory banks to work well [15, 18, 39]. It was later shown that when using a momentum encoder [15], simpler loss functions that did not directly discriminate against other images could be used [17, 19, 40, 41, 42]. Finally, a related line of work is to use clustering techniques to pre-train deep neural networks [43, 44, 45, 46, 47].

Transformer networks were originally introduced in the context of machine translation, replacing recurrent neural networks by an attention-based mechanism [48]. Transformers were later applied to image recognition, by splitting images into patches, embedding these independently, and then processing the obtained representations as a sequence [2]. Initially, only vision transformers pre-trained on very large collections obtained good performance, but smaller models trained on ImageNet with heavy augmentation can also yield competitive tradeoffs [49].

Pre-training data is an important ingredient of self-supervised learning, and multiple works have studied its impact on the transfer performance of models. While it is possible to learn high quality features from non-curated (eg. YFCC or IG) data using instance discrimination, this usually requires order of magnitude more data than ImageNet [21, 50]. Similarly, one can perform supervised pre-training using weakly supervised data, such as using hashtags as labels, but this strategy also requires large amount of data to work well [2, 51, 52]. On the other hand, it was shown that for many natural language processing tasks, increasing the size of the pre-training dataset did not lead to strong improvement when using denoising autoencoders [32]. Finally, some work studied how much could be learned from a single pre-training image [53] or from synthetic data [54, 55].

Table 1. Analysis of different self-supervision methods transfer performance using iNaturalist-2019 dataset when varying the size of the ImageNet subset used in the pre-training stage, in addition to using non object-centric dataset like COCO for pre-training. We observe that denoising autoencoders have a more robust behaviour w.r.t. pre-training data size or nature compared to joint embedding methods like DINO as well as supervised pre-training.

Method	IMNet 1% *epochs: 30k*	IMNet 10% *epochs: 3k*	IMNet Full *epochs: 300*	COCO *epochs: 3k*
Supervised	71.6	75.0	75.8	-
DINO [17]	70.1	73.1	78.4	71.9
BEiT [23]	74.1	74.5	75.2	74.4
SplitMask	74.8	75.4	75.4	76.3

3. Analysis

In this section, we study the impact of the pre-training data on the performance of denoising autoencoder, and how they compare to those of joint embedding methods. More precisely, we investigate how the number of images, and their nature, influence the quality of self-supervised models. In this premilinary analysis, we consider the recent method BEiT as representative of a denoising autoencoder, and DINO of a joint embedding method, respectively.

3.1. Sample Efficiency

First, we start by studying the impact of the pre-training dataset size, by varying the number of ImageNet examples we use to train models. We consider subsets of ImageNet containing 10% and 1% of the total number of examples, and use the balanced (in terms of classes) subsets from [56]. To decouple the effect of using smaller datasets and the effect of doing less training updates, we adapt the number of epochs to keep the number of iterations constant. This means that we perform 3k and 30k epochs on ImageNet 10% and 1% respectively. We report results in Table 1. Observe how pre-training with an autoencoder loss such as masked image modeling is robust to the reduction of the

Table 2. Ablation study on the effect of the method used to tokenize images. We compare the DALL-E tokenizer originally used in BEiT with patch level techniques: random projection, random patches and k-means clustering. We observe that the DALL-E tokenizer can be efficiently replaced by simpler methods that do not require training on a large dataset.

	DALL-E	Rand. Proj.	Rand. Patches	K-Means
iNat19	75.2	75.2	75.3	75.0

dataset size. In contrast, like for supervised pre-training, the performance of models pre-trained with DINO self-supervision degrades when training with smaller datasets.

3.2. Learning on non object-centric images

We now study the impact of changing the nature of the pre-training data. In particular we use images that are not object-centric, like in Imagenet. To this end, instead of pre-training on ImagetNet, we pre-train with images from the COCO dataset only. As COCO contains roughly 118k images, this dataset is approximately equivalent in terms of size to ImageNet 10%. Again, to disentangle the effect of training with a different number of iterations, we adapt the number of epochs and thus use 3k epochs on COCO.

We report the results of this experiments in Table 1. When pre-trained on COCO, DINO drops significantly compared to full ImageNet pre-training (-8.3). Interestingly, the drop is even higher than using 10% ImageNet even though the numbers of samples is roughly the same. We hypothesis this is because COCO images are not biased to be object-centric, while this joint embedding method was designed with ImageNet. In contrast, BEiT's performance only decreases slightly while SplitMask attains +0.7 improvement over full ImageNet pre-training. This is an interesting property which makes such models prime candidates for learning effectively from uncurated images in the wild.

3.3. Tokenizers

The BEiT method, as proposed by Bao et al. [23], relies on the discrete VAE tokenizer from DALL-E, which has been pretrained on a large weakly supervised dataset. Since we want to study whether it is possible to pre-train models solely on small datasets, or non object-centric ones, we replace the DALL-E tokenizer by a simple alternative. To this end, we consider different simple alternatives to discretize images at the patch level without any pre-training. Each of these techniques is applied on each patch independently, making them relatively lightweight and more efficient than the original tokenizer considered in BEiT.

Given a vocabulary of size V, each element of the vocabulary is represented by a vector $\mathbf{e}_i \in \mathbb{R}^d$, where $i \in \{1, ..., V\}$ and d is the dimension of patches (in the case of 16x16 patches, $d = 768$). Then, to tokenize an image, we associate each patch to the element of the vocabulary which has the highest cosine similarity with the patch in the pixel space. Hence, for a normalized patch $\mathbf{x}$, its corresponding token t is obtained as

$$t = \text{argmax}_{i \in \{1,...,V\}} \mathbf{x}^\top \mathbf{e}_i. \qquad (1)$$

We now discuss three simple ways to obtain the elements of the vocabulary $\mathbf{e}_i$. First, we can sample random vectors with uniform element-wise distribution, and call the corresponding tokenizer *random projection*. Second, we can sample V random patches uniformly in the set of all patches of images from the training set, and refer to the tokenizer as *random patches*. Finally, we can perform k-means clustering on the patches of images from the training set, and use the centroids as elements of the vocabulary. We refer to this last tokenizer, which was once widely employed in computer vision for bag-of-words representations, as *k-means*.

We train a ViT-base model on the ImageNet dataset, using these three tokenizers, as well as the DALL-E tokenizer originally considered by BEiT. We report results in Table 2. We observe that it is possible to replace the DALL-E tokenizer by simpler choices without suffering any significant degradation of accuracy. This also improves runtime by 26% with base models relatively to its variants using the DALL-E tokenizer on 16 gpus with a batch size of 1024. More implementation details and pseudo-code for our *random projection* are provided in the Appendix.

4. Methodology

In this section, we introduce a variant of denoising autoencoders based on vision transformers. An overview of our method is illustrated in Figure 3.

4.1. SplitMask

Our approach is based on three steps, which we refer to as *split*, *inpaint*[1] and *match*. As in standard vision transformers, an image is first split into patches of 16×16 pixels. Then, we *split* the patches into two disjoint subsets $\mathcal{A}$ and $\mathcal{B}$, which are processed independently by our deep ViT encoder. Next, using the patch representations of the subset $\mathcal{A}$ and a shallow decoder (e.g. 2 layers), we *inpaint* the patches of the subset $\mathcal{B}$, by solving a MIM task, and vice versa. Finally, we obtain a global image descriptor by average pooling of the patch representations from the decoder output corresponding to each subset.

The feature aggregation is over both observed and hallucinated patches. We try to *match* the global descriptors of the image obtained from subset $\mathcal{A}$ to that obtained from subset $\mathcal{B}$. In other words, we use the masking operation of the mask image modeling loss as a data augmentation for a contrastive learning loss similar to NPID or SimCLR. Note,

[1] Inpainting in this context is implemented by solving a Masked Image Modeling task rather than the typical inpainting by reconstruction of pixels.

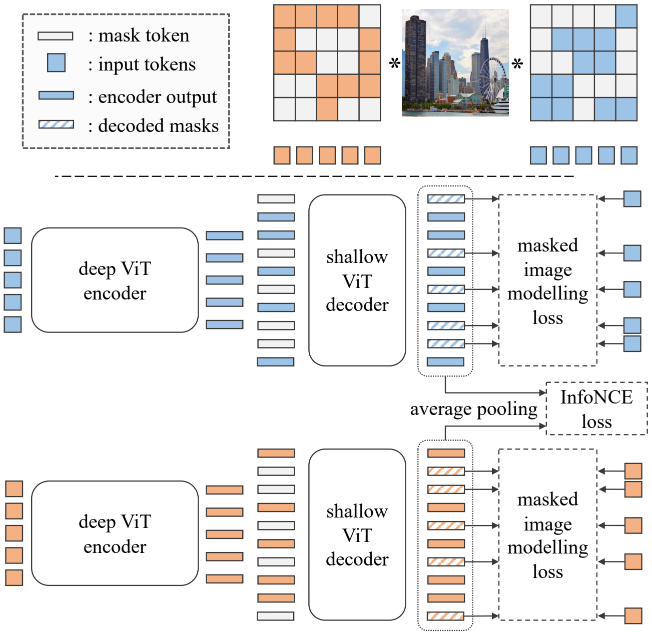

Figure 3. SplitMask consists of three steps. First, the input image patches are split into two disjoint subsets. Second, a shared deep ViT encoder processes each subset separately. The encoder outputs on each branch are augmented with a set of special mask tokens, representing the positions of the missing patches, and fed to a shallow ViT decoder. The decoder output corresponding to the mask tokens is used to solve a MIM task similar to BEiT. Finally, a global image descriptor is extracted from the decoder outputs of each branch by means of average pooling. The descriptors are trained to have high similarity using a contrastive loss (InfoNCE).

SplitMask does not add any significant computational cost over MIM methods like BEiT to produce this global contrastive training signal.

4.2. Encoder-Decoder Architecture

We now discuss in more details the architecture of the model that we use to implement the SplitMask paradigm described in the previous subsection. Our method relies on an encoder-decoder architecture. The encoder of our model is a standard vision transformer, with absolute positional embeddings. In contrast to BEiT method, our encoder does not process representations of the *masked* tokens, but only of the observed ones[2]. Hence, an image is divided into patches, which are linearly embedded, and positional embeddings are added to these representations. Then, these representations are split into two subsets $\mathcal{A}$ and $\mathcal{B}$, which are processed independently by standard transformer layers. Before feeding the output representations to the decoder, we insert mask embeddings that includes the position information of the missing patches in the sequences $\mathcal{A}$ and $\mathcal{B}$. Finally, using the decoded representations of the masked patches, we predict the corresponding tokens using a cross entropy loss function.

Thus, if an image contains n patches, the encoder processes two sequences of size $n/2$, while the decoder processes two sequences of size n. Since in practice we use decoder which is much more lightweight than standard vision transformers, the computational complexity of our models is similar to a standard ViT. One advantage of our approach compared to BEiT is that at each iteration, the encoder processes all the patches of the image. The loss function is also computed over all the patches of the image, instead of only on a subset.

4.3. Global Contrastive Loss

In addition to the MIM loss, which is computed at the patch level, our approach also uses a contrastive loss at the image level. To this end, we apply an average pooling operation over all the output representations of the decoder (including representations of the masked patches). For each image, we obtain two representations $\mathbf{x}_a$ and $\mathbf{x}_b$, corresponding to the subsets $\mathcal{A}$ and $\mathcal{B}$ of observed patches. We then apply the InfoNCE loss [58] over these representations:

$$\ell(\mathbf{x}_a) = \frac{\exp(\mathbf{x}_a^\top \mathbf{x}_b/\tau)}{\sum_{\mathbf{y}\in\{\mathbf{x}_b\}\cup\mathcal{N}} \exp(\mathbf{x}_a^\top \mathbf{y}/\tau)}, \tag{2}$$

where τ is a temperature hyper-parameter and $\mathcal{N}$ is a set of negatives, corresponding to the representations of the other images in the batch. Following previous work [18], we symmetrize the contrastive loss, and apply it similarly on the representation $\mathbf{x}_b$ from the subset $\mathcal{B}$. The motivation for adding this contrastive loss is to encourage the model to produce globally coherent features that is consistent across different observed subsets without relying on any hand-designed transformations. Using our design of SplitMask, we attain such signal with almost no overhead.

5. Experiments

In this section, we perform empirical evaluations of denoising autoencoders, and the impact of the pre-training data on downstream task performance. In particular, we study how well pre-training performs when only the target task data is used instead of relying on a large-scale dataset such as ImageNet. We perform experiments on different tasks, such as classification, detection and instance segmentation. We consider datasets of varying size, including some significantly smaller than ImageNet. We also compare our variant SplitMask method to BEiT, either pre-trained on target task data or ImageNet, in addition to the supervised pre-training baselines. Finally, we perform an ablation study on

[2] Concurrent to our work, He et al. [57] propose MAE. This is an encoder-decoder architecture where the encoder processing the observed patches only, similar to what we do in our SplitMask variant.

Table 3. Data size, number of classes and number of pre-training epochs details for all datasets used for pre-training.

Dataset	#Train	#Test	#Classes	Epochs
ImageNet [6]	1,281,167	50,000	1000	300
iNaturalist 2018 [11]	437,513	24,426	8,142	800
iNaturalist 2019 [12]	265,240	3,003	1,010	1,400
Food 101 [60]	75,750	25,250	101	5,000
Stanford Cars [59]	8,144	8,041	196	5,000
Clipart [61]	34,019	14,818	345	5,000
Painting [61]	52,867	22,892	345	5,000
Sketch [61]	49,115	21,271	345	5,000
ADE20K [63]	20,210	2,000	150	21,000
COCO [62]	118,287	5,000	80	3,000

our method to investigate the impact of its different components on finetuning and linear evaluation.

5.1. Datasets

We study the pre-training and finetuning of computer vision models on a variety of datasets, see Table 3 for details. For image classification, we consider the iNaturalist 2018 and 2019 [11], Stanford Cars [59] and Food101 [60] datasets, which all contain fine-grained categories. We also consider three subsets from the DomainNet dataset [61], *clipart*, *painting* and *sketch*, which are not natural images and hence from different domains than ImageNet. For object detection and instance segmentation, we use the COCO dataset [62]. Finally, we also use the ADE20k dataset [63] for segmentation. The training set sizes of these different datasets vary from 8k to 437k images, thus all being significantly smaller than ImageNet, some more than two order of magnitude smaller. This allows to investigate under different data regimes how feasible it is to pre-train directly on the target task data, alleviating the need for a large scale curated dataset as ImageNet. As previously mentioned, we want to perform a constant number of updates during pre-training, and we thus adapt the number of epochs when training on target task data to match the number of updates corresponding to 300 epochs on ImageNet. For smaller datasets, we limit the number of pre-training epochs to 5000 since we observed pre-training for longer generally does not result in further improvement in terms of downstream performance. For very small datasets, like Stanford Cars, we observed an overfitting behaviour with training for very long schedules (e.g. 30k epochs). Note that the adjusted number of pre-training epochs are also provided in Table 3.

5.2. Dense Prediction

5.2.1 Object detection and Instance Segmentation

First, we evaluate our approach on the COCO object detection and instance segmentation dataset using the Mask R-CNN pipeline [7] and report our results in Table 4. We compare models pre-trained on the COCO dataset alone with their equivalent counterparts that were pre-trained on ImageNet, either in a self-supervised fashion, or using supervision. First, we observe that BEiT models which were pre-trained on the COCO dataset alone obtain better downstream task performance than the same models pre-trained on ImageNet. For example, when using a ViT-base backbone, pre-training on COCO instead of ImageNet leads to a +0.4 box AP boost. Since COCO contains one order of magnitude less images than ImageNet, we believe that this is clear indication that large scale datasets are not necessary for pre-training. Additionally, we observe that a similar pre-training of DINO using COCO images provides a relatively weak performance, only outperforming random initialization. This indicates that strong pre-training on COCO is a unique property of denoising autoencoders and it does not extend to other self-supervised learning methods. Finally, we observe that SplitMask leads to a consistent improvement compared to the BEiT baseline, such as +0.6 box AP when using a ViT-small and +0.3 mask AP for ViT-base backbones. All put together, in a comparable setting, we obtain a +1.1 box AP improvement while not using ImageNet.

5.2.2 Semantic Segmentation

Similarly, in Table 5 we compare models pre-trained on ADE20K only to their counterparts pre-trained on ImageNet. For the finetuning on ADE20K we use the code in [23] and use an UperNet pipeline [65]. In order to make the pre-training on ADE20K on par with the ImageNet pre-training, we adapt the cropping strategy by reducing the maximal size of the crop from 100% of the size of the original image to 25%, see Appendix **??** for more details. It appears that it is possible to match performance of models pre-trained on ImageNet using the 20k images of ADE.

5.3. Image Classification

Next, we perform empirical evaluation on the classification datasets and report our results in Table 6. First, we compare ImageNet pre-training to the target data pre-training with BEiT and observe that for many cases, pre-training on the target data alone leads to better results compared to both supervised and self-supervised ImageNet pre-training. With a ViT-small backbone, this is true for all the datasets including Stanford cars (+1.1% acc), which consists of only 8k images. When using a ViT-base backbone, pre-training on the target task data outperforms BEiT self-supervised ImageNet pre-training for datasets as small as Food101 (+0.7 acc), which is more than 10x smaller than ImageNet. Second, with the exception of the Stanford Cars dataset, we observe that SplitMask leads to improved performances: for example, on the iNaturalist 2018 dataset, we see +2.2 in accuracy with a ViT-base model. Finally,

Table 4. COCO detection and instance segmentation performance, using a Mask R-CNN pipeline, for models with different pre-training recipes. We see that BEiT and SplitMask pre-training using COCO images outperform supervised ImageNet pre-training of DeiT as well as self-supervised ImageNet pre-training using BEiT. Additionally, joint embedding methods like DINO cannot attain the same performance as their denoising autoencoder counterparts. †: Method uses a longer 6x schedule instead of the default 3x following He et al. [64].

Method	Backbone	Pre-training			AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
		Supervised	IMNet	COCO						
Random Initialization	ViT-S	✗	✗	✗	38.3	60.1	41.4	35.6	57.1	37.7
Random Initialization†		✗	✗	✗	42.8	64.5	45.6	39.1	61.5	41.7
DeiT [49]		✓	✓	✗	44.2	66.6	47.9	40.1	63.2	42.7
BEiT [23]		✗	✓	✗	44.5	66.2	48.8	40.3	63.2	43.1
DINO [17]		✗	✗	✓	43.7	65.5	47.7	39.6	62.3	42.3
BEiT		✗	✗	✓	44.7	66.3	48.8	40.2	63.1	43.2
SplitMask		✗	✗	✓	**45.3**	**66.9**	**49.4**	**40.6**	**63.6**	**43.5**
Random Initialization	ViT-B	✗	✗	✗	40.7	62.7	44.2	37.1	59.1	39.4
Random Initialization†		✗	✗	✗	43.0	64.2	46.9	38.8	61.3	41.6
DeiT [49]		✓	✓	✗	45.5	**67.9**	49.2	41.0	64.6	43.8
BEiT [23]		✗	✓	✗	46.3	67.6	50.6	41.6	64.5	44.9
DINO [17]		✗	✗	✓	43.1	64.4	46.9	38.9	61.4	41.4
BEiT		✗	✗	✓	46.7	67.7	51.2	41.8	65.0	44.6
SplitMask		✗	✗	✓	**46.8**	**67.9**	**51.5**	**42.1**	**65.3**	**45.1**

Table 5. Segmentation performance for different pre-trained models on ADE20K using an UperNet pipeline [65]. All models reported use a ViT-B architecture. In spite of the small size of the ADE20k dataset, performance of our models provides a performance competitive to those pre-trained using ImageNet. The pre-training time of the DINO model is significantly longer than other models in the table. †: Method uses a three times longer schedule.

Method	Pre-training			mIoU
	Supervised	IMNet	ADE20k	
Random Init.	✗	✗	✗	25.4
DeiT [49]	✓	✗	✗	46.1
DINO [17]	✗	✓	✗	47.4
BEiT [23]	✗	✓	✗	45.6
BEiT	✗	✗	✓	45.6
SplitMask	✗	✗	✓	

as it was already observed in previous work [17, 18, 66], we also see in many cases that self-supervised training outperforms supervised pre-training on ImageNet. For example, on the iNaturalist datasets, training with the target task data alone (including a pre-training step) gives better results than pre-training on ImageNet with labels: with a ViT-base model and the SplitMask method, we see an improvement of +2.7% in top-1 accuracy on the iNaturalist 2019 dataset.

As for the *clipart*, *painting* and *sketch* datasets, we see that SplitMask provides a competitive performance, outperforming an ImageNet pre-trained BEiT across all datasets for ViT-S. However, for the aforementioned datasets, supervised pre-training achieves the best performance for both ViT-S and ViT-B. We note that when pre-training using the *clipart* and *sketch* datasets with the BEiT method, we experienced numerical instability that prevented the model from converging with long schedules (e.g. 5000 epochs). More investigation might be needed to fully understand how to optimize pre-training of such models.

5.4. Pre-training using ImageNet

In addition to our main study concerning the robustness of denoising autoencoders w.r.t the size and type of pre-training data, we study SplitMask in the more commonly used setting of pre-training and finetuning using ImageNet.

In Table 7 we show the performance of our SplitMask method using the ViT-S and ViT-B backbones and 300 epochs pre-training compared to other recent transformer-based self-supervised learning methods. It can be observed that SplitMask provides a strong performance outperforming both BEiT and MocoV3 for both backbones. Additionally, SplitMask achieves a performance on par with DINO while being significantly cheaper and simpler to train. While SplitMask and BEiT attain a strong finetuning performance, denoising autoencoding methods typically fall behind in terms of linear probing compared to instance discrimination methods like DINO.

5.5. Implementation Details

Tokenizers. Similarly to the tokenizer used in [23], all tokenizers presented in Table 2 have a vocabulary of size 8192. For the random tokenizer, we sample 8192 vectors with uniform component-wise distribution. For the random patches tokenizer we sample 8192 patches from different images. For the K-means tokenizer, the 8192 elements of the vocabulary are obtained by applying the K-means algorithm to 3 millions patches sampled from the dataset.

Table 6. Comparison between finetuning performance on the target datasets of different sizes and domains when pre-trained using the target datasets themselves, ImageNet pre-training (both supervised and self-supervised), and training from scratch. Both denoising autoencoders (BEiT and SplitMask) obtain competitive performance when solely using the target data. For each dataset. †: Liu et al. [67] uses different pre-training setup and backbones.

Method	Backbone	Supervised pre-training	Data Used		iNat-18	iNat-19	Food 101	Cars	Clipart	Painting	Sketch
			IMNet	Target	437k	265k	75k	8k	34k	52k	49k
Liu et al. [67]†	CVT-13	✗	✗	✓	-	-	-	-	60.6	55.2	57.6
	ResNet-50	✗	✗	✓	-	-	-	-	63.9	53.5	59.6
DeiT [49]	ViT-S	✓	✓	✓	69.9	75.8	**91.5**	92.2	**79.6**	**74.2**	**72.5**
BEiT [23]		✗	✓	✓	68.1	75.2	90.5	92.4	75.3	68.7	68.5
Random Init.		✗	✗	✓	59.6	67.5	84.7	35.3	41.0	38.4	37.2
BEiT		✗	✗	✓	68.8	76.1	90.7	92.7	-	69.0	-
SplitMask		✗	✗	✓	**70.1**	**76.3**	**91.5**	**92.8**	78.3	69.2	70.7
DeiT [49]	ViT-B	✓	✓	✓	73.2	77.7	**91.9**	92.1	**80.0**	**73.8**	**72.6**
BEiT [23]		✗	✓	✓	71.6	78.6	91.0	**93.9**	78.0	71.5	71.4
Random Init.		✗	✗	✓	59.6	68.1	83.3	36.9	41.9	37.6	34.9
BEiT		✗	✗	✓	72.4	79.3	91.7	92.7	-	70.7	-
SplitMask		✗	✗	✓	**74.6**	**80.4**	91.2	93.1	79.3	72.0	72.1

Table 7. Finetuning performance on ImageNet. Here, epochs refer to the number of pre-training epochs on ImageNet.

Method	Backbone	Epochs	Top-1
MocoV3 [66]	ViT-S	300	81.4
DINO [17]		300	**81.5**
BEiT [23]		300	81.3
SplitMask		300	**81.5**
MocoV3 [66]	ViT-B	300	83.2
DINO [17]		400	**83.6**
BEiT [23]		300	82.8
BEiT [23]		800	83.2
SplitMask		300	**83.6**

Pre-training. We use the original ViT formulation as proposed by Dosovitskiy et al. [2] and we follow the pre-training hyperparameters of Bao et al. [23]. All baselines reported use the same backbone implementation and trained in similar settings. For SplitMask, we use random block masking [23] of 50% masking ratio to obtain a mask and its complement to extract the two subsets. The maximum and minimum number of patches per block is 75 and 16 respectively. We use the standard random cropping and horizontal flipping as data augmentations. We use 2 transformer layers for the decoder with embedding dimension matching that of the encoder. The BEiT baselines pre-trained on ImageNet and reported in Table 4 and 6 use the DALL-E tokenizer. Other BEiT and SplitMask models have been pre-trained using our random projection tokenizer. For the InfoNCE loss we use $\tau = 0.2$ following Chen et al. [66].

Object detection and Instance segmentation. We use the Mask R-CNN detection method[7] with ViT backbone as our detection method. In order to obtain features compatible with the Feature Pyramid Network (FPN) design [68], we use max pooling and transposed convolution operations similar to El-Nouby et al. [69]. To accommodate for the variable resolution we replace the absolute positional encoding for our models and the baselines with sinusoidal positional encoding [48]. All models are trained using the 3x schedule (36 epochs) unless mentioned otherwise. We use the training hyper-parameters used by Liu et al. [3].

Image classification finetuning. Hyperparameters used for finetuning each of the specific image classification datasets reported in Table 6 is provided in the Appendix.

6. Conclusion

In this paper, we have raised the question of how to pre-train models with self-supervised learning, wondering in particular on whether large scales datasets such as Imagenet are necessary for pre-training. Our study on ImageNet shows that taking a smaller pre-training dataset does not lead to big performance drop for denoising autoencoders, as opposed to instance discrimination self-supervised techniques or supervised pre-training. Similarly, training on non object-centric images does not impact the downstream task performance significantly.

Building upon these observations, we have pre-trained models directly on the target task data, instead of ImageNet, and performed evaluations on datasets of various sizes. We have shown that it is possible to pre-train on datasets 10x smaller than ImageNet, for example obtaining +0.5 box AP gains by solely using COCO images. *We believe that this is strong evidence that large scale datasets, such as ImageNet, are not necessary for self-supervised pre-training when using denoising autoencoders.*

Acknowledgement. We thank Armand Joulin, Jakob Verbeek, Natalia Neverova and Gabriel Synnaeve for the fruitful discussion around this project.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition*, 2016. 1

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021. 1, 3, 8

[3] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *arXiv preprint arXiv:2103.14030*, 2021. 1, 8

[4] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár, "Designing network design spaces," in *Computer Vision and Pattern Recognition*, 2020. 1

[5] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson, "How transferable are features in deep neural networks?" *arXiv preprint arXiv:1411.1792*, 2014. 1

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition*, 2009. 1, 6

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask r-cnn," in *International Conference on Computer Vision*, 2017. 1, 6, 8

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, 2020. 1

[9] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014. 1

[10] Joao Carreira and Andrew Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308. 1

[11] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie, "The inaturalist species classification and detection dataset," *Computer Vision and Pattern Recognition*, 2018. 2, 6

[12] Grant Van Horn, Oisin Mac Aodha, Yang Song, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie, "The iNaturalist species classification and detection dataset," *arXiv preprint arXiv:1707.06642*, 2017. 2, 6

[13] Carl Doersch, Ankush Gupta, and Andrew Zisserman, "Crosstransformers: spatially-aware few-shot transfer," *arXiv preprint arXiv:2007.11498*, 2020. 2

[14] Eleanor H. Rosch, "Natural categories," *Cognitive Psychology*, 1973. 2

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, "Momentum contrast for unsupervised visual representation learning," in *Computer Vision and Pattern Recognition*, 2020. 2, 3

[16] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *arXiv preprint arXiv:2006.09882*, 2020. 2

[17] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin, "Emerging properties in self-supervised vision transformers," *arXiv preprint arXiv:2104.14294*, 2021. 2, 3, 7, 8

[18] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020. 2, 3, 5, 7

[19] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020. 2, 3

[20] Senthil Purushwalkam and Abhinav Gupta, "Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases," *arXiv preprint arXiv:2007.13916*, 2020. 2

[21] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin *et al.*, "Self-supervised pretraining of visual features in the wild," *arXiv preprint arXiv:2103.01988*, 2021. 2, 3

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018. 2

[23] Hangbo Bao, Li Dong, and Furu Wei, “Beit: Bert pre-training of image transformers,” *arXiv preprint arXiv:2106.08254*, 2021. 2, 3, 4, 6, 7, 8, I

[24] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103. 2

[25] Geoffrey E Hinton and Ruslan R Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *science*, vol. 313, no. 5786, pp. 504–507, 2006. 2

[26] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle, “Greedy layer-wise training of deep networks,” in *Advances in neural information processing systems*, 2007, pp. 153–160. 2

[27] Marc Ranzato, Christopher Poultney, Sumit Chopra, Yann LeCun *et al.*, “Efficient learning of sparse representations with an energy-based model,” *Advances in neural information processing systems*, vol. 19, p. 1137, 2007. 2

[28] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.” *Journal of machine learning research*, vol. 11, no. 12, 2010. 2

[29] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros, “Context encoders: Feature learning by inpainting,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544. 2

[30] Richard Zhang, Phillip Isola, and Alexei A Efros, “Colorful image colorization,” in *European conference on computer vision*. Springer, 2016, pp. 649–666. 2

[31] 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. 2

[32] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *arXiv preprint arXiv:1910.10683*, 2019. 2, 3

[33] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer, “Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” *arXiv preprint arXiv:1910.13461*, 2019. 2

[34] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever, “Generative pretraining from pixels,” in *International Conference on Machine Learning*, 2020. 2

[35] Sara Atito, Muhammad Awais, and Josef Kittler, “Sit: Self-supervised vision transformer,” *arXiv preprint arXiv:2104.03602*, 2021. 2

[36] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever, “Zero-shot text-to-image generation,” *arXiv preprint arXiv:2102.12092*, 2021. 2

[37] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox, “Discriminative unsupervised feature learning with convolutional neural networks,” *Advances in neural information processing systems*, vol. 27, pp. 766–774, 2014. 3

[38] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox, “Discriminative unsupervised feature learning with exemplar convolutional neural networks,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 9, pp. 1734–1747, 2015. 3

[39] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin, “Unsupervised feature learning via non-parametric instance discrimination,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3733–3742. 3

[40] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny, “Barlow twins: Self-supervised learning via redundancy reduction,” *arXiv preprint arXiv:2103.03230*, 2021. 3

[41] Adrien Bardes, Jean Ponce, and Yann LeCun, “Vicreg: Variance-invariance-covariance regularization for self-supervised learning,” *arXiv preprint arXiv:2105.04906*, 2021. 3

[42] 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. 15 750–15 758. 3

[43] Junyuan Xie, Ross Girshick, and Ali Farhadi, “Unsupervised deep embedding for clustering analysis,” in *International conference on machine learning*. PMLR, 2016, pp. 478–487. 3

[44] Jianwei Yang, Devi Parikh, and Dhruv Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016. 3

[45] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 132–149. 3

[46] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi, "Self-labelling via simultaneous clustering and representation learning," *arXiv preprint arXiv:1911.05371*, 2019. 3

[47] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *arXiv preprint arXiv:2006.09882*, 2020. 3

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017. 3, 8

[49] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou, "Training data-efficient image transformers and distillation through attention," *arXiv preprint arXiv:2012.12877*, 2020. 3, 7, 8

[50] Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin, "Unsupervised pre-training of image features on non-curated data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2959–2968. 3

[51] Armand Joulin, Laurens Van Der Maaten, Allan Jabri, and Nicolas Vasilache, "Learning visual features from large weakly supervised data," in *European Conference on Computer Vision*. Springer, 2016, pp. 67–84. 3

[52] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten, "Exploring the limits of weakly supervised pretraining," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 181–196. 3

[53] Yuki M Asano, Christian Rupprecht, and Andrea Vedaldi, "A critical analysis of self-supervision, or what we can learn from a single image," *arXiv preprint arXiv:1904.13132*, 2019. 3

[54] Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh, "Pre-training without natural images," in *Proceedings of the Asian Conference on Computer Vision*, 2020. 3

[55] Kundan Krishna, Jeffrey Bigham, and Zachary C Lipton, "Does pretraining for summarization require knowledge transfer?" *arXiv preprint arXiv:2109.04953*, 2021. 3

[56] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Armand Joulin, Nicolas Ballas, and Michael Rabbat, "Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples," *arXiv preprint arXiv:2104.13963*, 2021. 3

[57] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick, "Masked autoencoders are scalable vision learners," *arXiv preprint arXiv:2111.06377*, 2021. 5

[58] Aaron van den Oord, Yazhe Li, and Oriol Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018. 5

[59] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei, "3d object representations for fine-grained categorization," in *IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, 2013. 6

[60] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool, "Food-101 – mining discriminative components with random forests," in *European Conference on Computer Vision*, 2014. 6

[61] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang, "Moment matching for multi-source domain adaptation," in *International Conference on Computer Vision*, 2019. 6

[62] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*, 2014. 6

[63] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba, "Scene parsing through ade20k dataset," in *Computer Vision and Pattern Recognition*, 2017. 6

[64] Kaiming He, Ross Girshick, and Piotr Dollár, "Rethinking im-agenet pre-training," *arXiv preprint arXiv: 1811.08883*, 2018. 7

[65] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun, "Unified perceptual parsing for scene understanding," in *European Conference on Computer Vision*, 2018. 6, 7

[66] Xinlei Chen, Saining Xie, and Kaiming He, "An em-

pirical study of training self-supervised vision transformers," *arXiv preprint arXiv:2104.02057*, 2021. 7, 8

[67] Yahui Liu, Enver Sangineto, Wei Bi, Nicu Sebe, Bruno Lepri, and Marco Nadai, "Efficient training of visual transformers with small datasets," in *Advances in Neural Information Processing Systems*, 2021. 8

[68] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie, "Feature pyramid networks for object detection," in *Computer Vision and Pattern Recognition*, 2017. 8

[69] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek *et al.*, "Xcit: Cross-covariance image transformers," *arXiv preprint arXiv:2106.09681*, 2021. 8

Appendix

A. Random Projection Tokenizer Pseudo code

Algorithm 1 Pseudocode of Random Projection Tokenizer

```
# input_size: 112x112

class RandomTokenizer(torch.nn.Module):

    def __init__(self,
                 vocab_size=8192,
                 kernel_size=8,
                 stride=8):

        self.proj = torch.nn.Conv2d(
            3, vocab_size, kernel_size,
            stride=stride, bias=False
        )

        self.proj.weight.data = F.normalize(
            self.proj.weight.data.view(vocab_size, -1),
            dim=-1
        ).view(vocab_size, 3, kernel_size, kernel_size)

    def get_codebook_indices(self, images):
        return torch.argmax(self.proj(images), dim=1)
```

A.1. SplitMask vs BEiT

We ablate our proposed components in SplitMask compared to a BEiT baseline in Table 8. All models use a ViT-B backbone and pre-trained for 300 epochs. First, we observe that the ImageNet finetuning performance improves with a margin (+0.5) by simply adopting the encoder-decoder architecture and processing two disjoint subsets per iteration. Second, the global contrastive loss on its own, without the MIM objective, provides a very weak performance. This is expected since there is no training signal for the local patch representations, and a global matching objective with 50% masking of patches may be too hard, providing a noisy training signal and hindering the model's ability to learn informative features.

Our full SplitMask model that uses both the MIM and contrastive objectives obtains the best performance and outperforms BEiT by a large margin of +0.8. The Linear probing performance of SplitMask is stronger than BEiT. However, both models provide a relatively weak performance on this benchmark compared to instance discrimination methods, whose final layers are more aligned to the classification task. Note, SplitMask adds a negligible computing overhead compared to the BEiT baseline: its wall-clock training time is marginally higher as detailed in Table 8. All models are trained using 16 GPUs and batch size of 2048.

Table 8. Ablations of different components in our SplitMask model in comparison with a BEiT baseline. All models including the baseline have been trained for 300 epochs using a ViT-B backbone. We report the finetuning and linear probing performance, as well as the wall-clock training time of each model.

Method	Split	Inpaint	Match	Finetune	Lin.	Hours
BEiT [23]	✗	✓	✗	82.8	41.0	32.5
SplitMask	✓	✓	✗	83.3	46.4	**31.0**
	✓	✗	✓	79.3	4.0	32.5
	✓	✓	✓	**83.6**	**46.5**	34.0

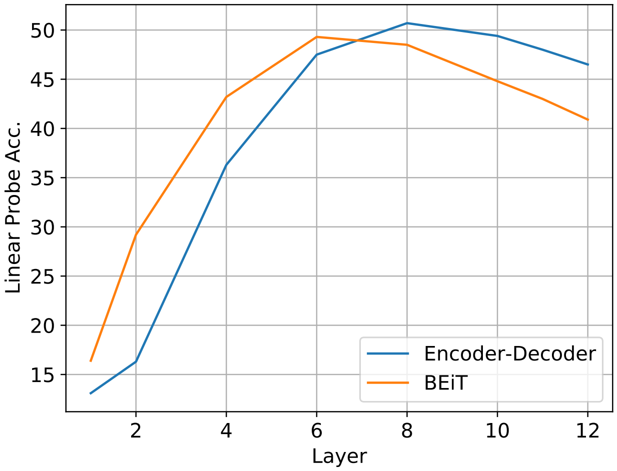

Figure 4. Linear probing accuracy on ImageNet for SplitMask and BEiT using features extracted from different layers. We can observe that, while both models peak performance does not correspond to their last layers, SplitMask has a better performance at the later layers compared to BEiT which has its peak performance at the sixth layer.

B. Encoder-Decoder vs BEiT

An advantage of the encoder-decoder design we propose in 4.2 is that it encourages decoupling of general-purpose encoding of image features, which is required for the downstream tasks, and features specific to solving the pretext task of MIM. In particular, compared to BEiT the encoder is not capable of solving the pretext task on its own since it does not have access to the mask token. Therefore, it can only help solve the task by providing informative representation to the decoder which is the component responsible of solving the pretext task. We can see in Figure 4 that this property improves the transferability of later layers representation to downstream tasks compared to BEiT which has a stronger drop in linear probing performance in later layers.

C. Overfitting during pre-training

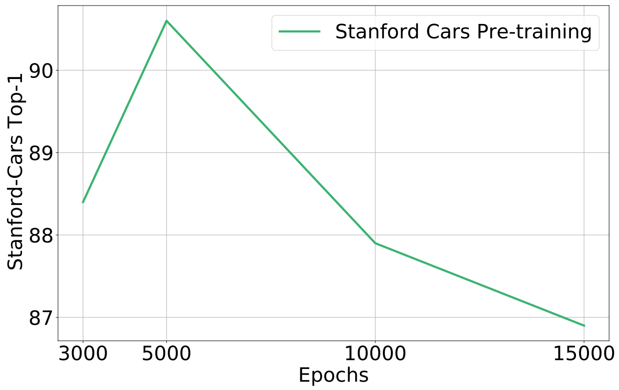

Figure 5

D. Image Classifcation Finetuning

Hyperparameters used for finetuning on the different classification datasets are reported in Table 9.

Table 9. Hyperparameters used for finetuning on the different classification datasets.

Dataset	iNat18	iNat19	Food 101	Cars	Clipart	Painting	Sketch
Train Res	224	224	224	224	224	224	224
Test Res	224	224	224	224	224	224	224
Epochs	300	300	300	1000	300	300	300
Batch size	1024	1024	1024	1024	1024	1024	1024
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Learning rate (LR)	1.4e-4	1.4e-4	1.4e-4	4e-3	4e-3	4e-3	4e-3
LR schedule	cosine	cosine	cosine	cosine	cosine	cosine	cosine
LR layer decay small models	✗	✗	✗	0.75	0.75	0.75	0.75
LR layer decay base models	✗	✗	✗	0.65	0.65	0.65	0.65
Weight decay	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Warmup epochs	5	5	5	200	60	60	60
Label smoothing	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Dropout	✗	✗	✗	✗	✗	✗	✗
Stoch. Depth	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Repeated Aug	✓	✓	✓	✗	✗	✗	✗
Gradient Clip.	✗	✗	✗	✗	✗	✗	✗
H. flip	✓	✓	✓	✓	✓	✓	✓
Random Resize Crop	✓	✓	✓	✓	✓	✓	✓
Rand Augment (magnitude/std)	7/0.5	7/0.5	7/0.5	9/0.5	9/0.5	9/0.5 9/0.5	9/0.5
Auto Augment	✓	✓	✓	✓	✓	✓	✓
Mixup alpha	0.8	0.8	0.8	0.8	0.8	0.8	0.8
Cutmix alpha	1.0	1.0	1.0	1.0	1.0	1.0	1.0
ColorJitter	0.4	0.4	0.4	0.4	0.4	0.4	0.4
Test crop ratio	0.875	0.875	0.875	0.875	0.875	0.875	0.875