
Position Prediction as an Effective Pretraining Strategy

Shuangfei Zhai¹ Navdeep Jaitly¹ Jason Ramapuram¹ Dan Busbridge¹ Tatiana Likhomanenko¹
Joseph Yitan Cheng¹ Walter Talbott¹ Chen Huang¹ Hanlin Goh¹ Joshua Susskind¹

Abstract

Transformers (Vaswani et al., 2017) have gained increasing popularity in a wide range of applications, including Natural Language Processing (NLP), Computer Vision and Speech Recognition, because of their powerful representational capacity. However, harnessing this representational capacity effectively requires a large amount of data, strong regularization, or both, to mitigate overfitting. Recently, the power of the Transformer has been unlocked by self-supervised pretraining strategies based on masked autoencoders which rely on reconstructing masked inputs, directly, or contrastively from unmasked content. This pretraining strategy which has been used in BERT models in NLP (Devlin et al., 2019), Wav2Vec models in Speech (Baevski et al., 2020) and, recently, in MAE models in Vision (Bao et al., 2021; He et al., 2021), forces the model to learn about relationships between the content in different parts of the input using autoencoding related objectives. In this paper, we propose a novel, but surprisingly simple alternative to content reconstruction – that of predicting locations from content, without providing positional information for it. Doing so requires the Transformer to understand the positional relationships between different parts of the input, from their content alone. This amounts to an efficient implementation where the pretext task is a classification problem among all possible positions for each input token. We experiment on both Vision and Speech benchmarks, where our approach brings improvements over strong supervised training baselines and is comparable to modern unsupervised/self-supervised pretraining methods. Our method also enables Transformers trained without position embeddings to outperform ones trained with full position information.

¹Apple Inc. Correspondence to: Shuangfei Zhai <szhai@apple.com>.

1. Introduction

Transformers (Vaswani et al., 2017) have become a unified architecture in NLP, Computer Vision and Speech. Their high capacity and lack of domain specific inductive biases means Transformers require large amounts of training data to achieve good generalization. One effective remedy, first developed in the NLP community, is unsupervised pretraining. For example, BERT (Devlin et al., 2019) trains a Transformer with unlabeled text by solving masked token prediction. This greatly benefits downstream applications, and has become the standard approach for various NLP tasks.

Recently, there have been a few attempts to apply the BERT pretraining idea to Computer Vision, with Vision Transformers (ViTs) (Dosovitskiy et al., 2021) being the backbone architecture. In particular, BEiT (Bao et al., 2021) converts image patches to discrete tokens with a separately trained VQVAE (van den Oord et al., 2017). This makes it possible to use the same cross entropy loss for masked image patch prediction as for token prediction in BERT. MAE (He et al., 2021) further simplifies the recipe of BEiT by directly predicting the masked patches with a regression loss in the pixel space.

In this paper, we propose a simple and effective approach for Transformer pretraining that removes the need for reconstructing dense patch values. Our idea is inspired by the observation that Transformers are relatively insensitive to the order of input tokens. In (Naseer et al., 2021), it is shown that pretrained ViTs demonstrate strong robustness to image patch shuffling perturbation at test time. (Sinha et al., 2021) shows that training the BERT model with randomly shuffled word order gives surprisingly competitive performance. (Chen et al., 2021) also suggests that a ViT without positional embeddings shows only a small degradation in the linear probing task for self-supervised learning. This evidence suggests that much of the power of Transformers results from the ability to reason about the co-occurrence of the set of unordered input tokens. We thus ask the question: *How much can unsupervised pretraining learn using only contents for prediction?* This motivates us to formulate a novel pretraining strategy, explained as follows.

In the pretraining phase, the model (e.g., a ViT) receives a set of tokens (e.g., image patches) but not their positions,

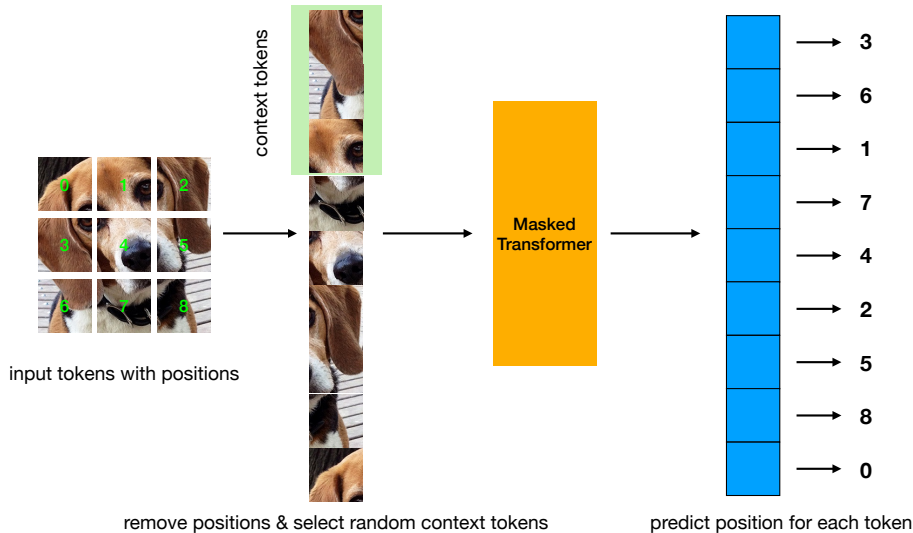


Figure 1. Illustration of our method MP3 on images. MP3 removes the position information for all tokens (image patches); it then randomly select a subset of tokens as context tokens. A **Masked Transformer** is used, where in each attention layer only context tokens contribute to the keys and values, and all tokens contribute to the queries. Each token predicts its position with a linear classifier head.

and the pretext task is to recover the position of each input token, cast as a classification problem among all positions. By doing so, we formulate a special case of masked autoencoder, where the positions, rather than tokens, are removed from the input and predicted by the model. This training objective can also be interpreted as training a Set Transformer (Lee et al., 2019) to solve a Jigsaw puzzle (Noroozi & Favaro, 2016). In order to solve the task, the Transformer needs to reason about the high order interaction of the input tokens, which amounts to understanding the underlying semantics (e.g., part-whole relationship of the given object) represented by the inputs. Empirically, we have found that large Transformers can often achieve near perfect accuracy on the position prediction task¹. We then propose to increase the task’s difficulty by selecting a random subset of tokens as context, and modify the attention layers such that only context tokens are used as keys and values. In this way, the Transformer not only needs to order the context tokens, but also infer the positions for masked out tokens by querying into the context. We hence dub our method **MP3**, denoting Masked Patch Position Prediction.

During finetuning, we disable token masking and add absolute positional embeddings in the same way as standard Transformers. We then remove the linear position prediction head and replace it with a linear head for the downstream task (e.g., classification). All the parameters are updated for a desired number of finetuning steps with the downstream

¹Except for Speech, where a patch (audio frame) is a small part of the full sequence, and it is much more challenging without providing some reference points with known positions.

task’s training objective.

MP3 amounts to a simple implementation. In the pretraining phase, no additional modules are needed other than a linear head with $d \times n$ parameters, where d is the model’s feature dimension and n is the number of positions. The training objective is simply the cross entropy loss. Also, thanks to the context masking, full self-attention is reduced to sparse attention, which effectively makes the pretraining cost lower than that of finetuning.

We conduct experiments on both Vision and Speech tasks. MP3 consistently improves the performance of Transformer models compared to strong supervised training baselines, and matches other more sophisticated unsupervised/self-supervised pretraining methods, despite its simplicity. Remarkably, MP3 enables strong finetuning performance even without using position embeddings, sometimes outperforming the supervised training baselines by a large margin.

2. Related Work

Denosing Autoencoders (DAEs). DAEs (Vincent et al., 2010) are well studied models in the context of unsupervised pretraining. The idea is to reconstruct the inputs given noisy versions of themselves. Masked autoencoder (MAE) is a special case of DAE, where part of the inputs are masked out (with a multiplicative Bernoulli noise). When combined with Transformers, MAEs have shown great success as an unsupervised pretraining technique, with BERT (Devlin et al., 2019), BEiT (Bao et al., 2021) and MAE (He et al.,

2021) as notable examples. MP3 can also be viewed as a special case of MAEs, but it masks out the positional information (and optionally input tokens), rather than input tokens. The reconstruction objective is then turned into a sorting task, which has very different implications than reconstructing missing tokens given positions.

Self-supervised learning with order prediction. Unsupervised feature learning with order prediction of image patches is first proposed in (Noroozi & Favaro, 2016), and then extended in followup works such as (Lee et al., 2017; Ahsan et al., 2019; Xu et al., 2019; Santa Cruz et al., 2018; El-Nouby et al., 2019). What these works share is that they often adopt a CNN based encoder for an image patch or a video clip, and an MLP based prediction network to output the correct order of a set of inputs (except (El-Nouby et al., 2019) which uses order prediction to approximate future prediction in videos). The output of these methods is then a *local* representation for image patches or video clips, as the order prediction network is discarded. This is in stark contrast with our work, MP3 focuses on learning the *global* representation via attention. This is only made possible by the powerful Transformer architecture, which focuses on learning the interactions between input elements, and the same global knowledge is transferred to downstream tasks in the finetuning step.

Importance of positional embedding in Transformers. Positional embeddings (PEs) are of unique importance to Transformers, and improving PEs is an active research area, see (Dufter et al., 2021) for an overview. However, it is empirically observed that the performance of Transformers is surprisingly robust to the order of the inputs. For ViTs, (Naseer et al., 2021) shows that pretrained ViTs suffer much less from patch shuffling perturbations than CNNs. (Sinha et al., 2021) shows that masked language models perform well even when trained with shuffled sentences. (Chen et al., 2021) also shows that a Transformer without PEs shows only a small degradation when evaluated with linear probing in a contrastive learning setup. MP3 confirms the hypothesis that much of the Transformer’s power lies in its ability to model the co-occurrence of input tokens. In particular, our pretraining method does not use or train PEs at all (instead of randomly shuffling input tokens while using PE), and still performs competitively compared to other baselines.²

Contrastive Learning. This is a family of methods for self-supervised learning, where the learning objective is to assign high similarity to augmented views from the same example (van den Oord et al., 2018; Chen et al., 2020; 2021; Caron et al., 2021). MP3 differs as it does not rely on data augmentation as the source of training signal, which gives it much more flexibility. Besides, MP3 does not enforce clustering

of the representation for different positions within an input, which makes it not suitable for linear probing tasks. These differences also suggest a possibility of combining MP3 and contrastive learning to achieve the best of both worlds. There has also been attempts combining contrastive learning with predictive tasks (Dangovski et al., 2021), which suggests possible ways of combining MP3 with contrastive learning in a similar fashion.

Position prediction in NLP. In concurrent works, the idea of position prediction has also been explored in the NLP domain (Cui et al., 2022; Br uel-Gabrielsson & Scarvelis, 2022). These works, combined with MP3, suggest that position prediction is a promising technique across a wide range of problems.

3. Method

3.1. Architecture

For Vision, our architecture is based on ViTs (Dosovitskiy et al., 2021). In a nutshell, ViTs divide an image into non-overlapping patches of a given size (e.g., 16×16). A linear projection with shared weights to all image patches to obtain a sequence of image tokens is then applied. Token vectors are additively combined with their respective positional embeddings to form the input sequence. Standard self-attention layers are then applied to process the input sequence.

For Speech, our architecture is based on the vanilla Transformer. The input to the model is a sequence of frames of 40 mel filterbank cepstral coefficients (MFCCs), computed from 30ms of raw waveforms, strided by 10ms between frames, following (Choi et al., 2019). Each frame is transformed by the same linear projection into the dimension of the transformer model (thus, each frame is treated as a 1D patch). A fixed sinusoidal positional embedding (Vaswani et al., 2017) is added to these projected representation and the result is fed into an 8 layer Transformer. As with ViTs, we add a learnable “cls” token frame at the beginning of the model input sequence. Compared to Vision, in Speech we have a patch that is 1D rather than 2D as we ignore the structure in the frequency domain. Later in the text, we refer to a frame as a patch in the context of Speech tasks.

3.2. Masked Position Prediction Pretraining

In the pretraining phase, we apply the same patch projection as standard ViTs but remove the positional embeddings from all the patch representations. This results in a set of patch representations. We next randomly select $1 - \eta$ fraction of patches as “context patches”, where η denotes the masking ratio. We then modify the self-attention layers accordingly, where only the context patches take part in the computation of keys and values; queries are computed for all patches. In other words, we perform cross attention from

²Note that for Speech some positions are needed to be added – otherwise, the pretraining task is too hard for the model to solve.

Table 1. Overview of datasets and baseline models. The ViT-S and ViT-B architectures are defined according to (Touvron et al., 2021).

Dataset	Input size	#Examples	#Classes	Model config	Patch size	Patch stride	#Positions
CIFAR-100	32×32	50K	100	ViT-S	4×4	4	64
Tiny ImageNet	64×64	100K	200	ViT-B	8×8	8	64
ImageNet-1K	224×224	1.3M	1K	ViT-B	16×16	16	196
Google Speech Commands	1s	22246	12	8 layer Transformer	30ms	10ms	100

all input patches to the context patches. With $\eta > 0$, the Transformer needs to formulate a good representation of the input given only a subset of the input patches, while ordering all the input patches. This forces the model to reason about the relationship of the context patches and infer masked patches at the same time. As a byproduct, a high masking ratio effectively reduces the computational cost of the Transformer in the pretraining phase.

We attach a linear prediction head after the last attention layer, with input and output dimensions being the feature dimension d and number of patches n , respectively. The outputs of the linear head are passed to Softmax to form a distribution over patch positions. The position prediction loss is obtained with the cross entropy between the position index and the prediction head’s outputs. See Figure 1 for an illustration, and Appendix A for a sketch implementation.

3.3. Supervised Finetuning

After the unsupervised pretraining step, we finetune the network with labels. Specifically, we remove the position prediction head, and attach a linear classifier head after the “cls” token, as in standard ViTs. We also apply randomly initialized (learned) positional embeddings (or fixed sinusoidal) to the patch embeddings, also following standard ViTs. Random masking is disabled during this stage and full self-attention is used. The remaining setting of the finetuning step largely resembles that of the supervised training.

4. Evaluations

4.1. Experimental Setting

While there has been a lot of interest in scaling Transformers on large datasets in the literature, their performance on small datasets remains under explored. As Transformers tend to overfit easily with pure supervised learning, we believe that it is of great importance to investigate the power of unsupervised pretraining in scarce data settings. In the domain of vision, we experiment with small to medium sized datasets: CIFAR-100 (Krizhevsky et al., 2009), Tiny ImageNet³ and ImageNet-1K (Deng et al., 2009). In the Speech domain we did not attempt a full blown application of MP3 to Automatic Speech Recognition because the

notion of locations is vague, with the streaming nature of the data. Instead we opted here to show proof of concept by applying MP3 to the keyword spotting task, which is a classification problem on a fixed length snippet of audio. We use the Google Speech Commands dataset v1 (Warden, 2018) and implemented our models using the publicly available implementation of TC-ResNet (Choi et al., 2019)⁴, keeping their audio preprocessing routines, data splits and other details intact. For each dataset above, we choose a baseline Transformer model configuration, the details of which are summarized in Table 1.

4.2. Pretraining and Finetuning on Vision Data

Implementation details. For CIFAR-100, Tiny ImageNet and ImageNet-1k, both our pretraining and finetuning settings largely follow DeiT (Touvron et al., 2021), which uses AdamW (Loshchilov & Hutter, 2017) optimizer, weight decay of 0.05, drop path (Ghiasi et al., 2018) rate of 0.1, RandAugment (Cubuk et al., 2020), CutMix (Yun et al., 2019), MixUp (Zhang et al., 2017), Random Erasing (Zhong et al., 2020), Repeated Augmentation (Hoffer et al., 2020) and label smoothing. In the pretraining phase, we do not use CutMix, MixUP, Random Erasing, Repeated Augmentation and label smoothing. The finetuning phase follows exactly the same protocol as the supervised training recipes suggested in (Touvron et al., 2021). We search for optimal η for each dataset in the pretraining phase, which is 0.5, 0.8, 0.75 for CIFAR-100, Tiny ImageNet and ImageNet-1K, respectively. The batch size is 256, 512 and 2048, respectively.

Baselines. On each dataset, the supervised baseline is trained with strong regularizations. We fix the total training epoch to 400 epochs for CIFAR-100 and Tiny ImageNet, and 300 for ImageNet-1K. We also consider two additional supervised training baselines, one without positional embeddings and another with 2D relative position biases (Shaw et al., 2018). We also consider two Transformer based self-supervised pretraining methods, MOCO V3 (Chen et al., 2021) and MAE (He et al., 2021). In both cases, we use the official code bases and search for the optimal hyper parameter for each case (data augmentation, learning rate for MOCO V3; masking ratio and learning rate for MAE).

⁴<https://github.com/hyperconnect/TC-ResNet>

³<http://cs231n.stanford.edu/tiny-imagenet-200.zip>

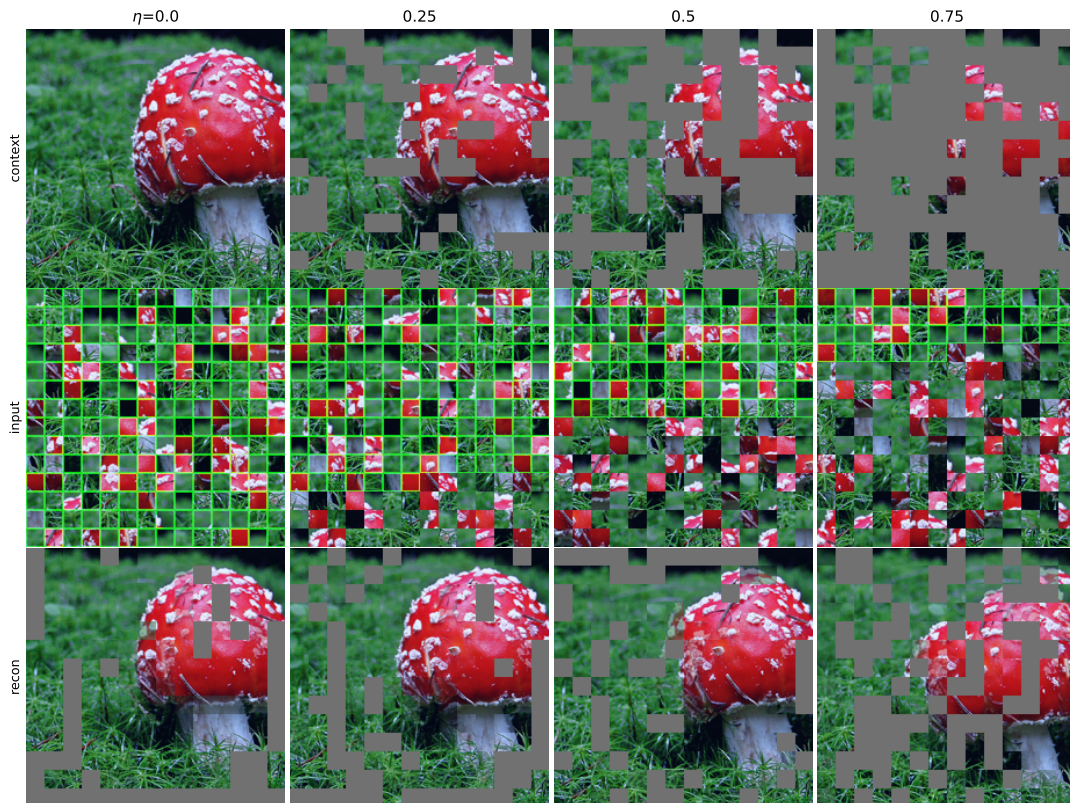


Figure 2. An image from the ImageNet validation set (top left corner) and its reconstructed images for a model trained with $\eta = 0.75$. **Column 1 - 4:** different η used at *test time*, ranging in $\{0, 0.25, 0.5, 0.75\}$. **Row 1:** the random context patches, placed in their original locations. **Row 2:** the unordered inputs to the model, with the context patch tokens outlined in green. **Row 3:** each patch is placed in the predicted position, and patches falling in the same position are averaged. The content in the reconstructed images are still apparent despite distortions. See Appendix F for additional examples.

4.2.1. PRETRAINING EFFICIENCY

We first measure the training efficiency of MP3, compared to MAE as well as the supervised training baseline ViT-B. In Table 2 we report the training time (seconds per iteration) and the memory consumption (in gigabytes) on ImageNet-1K with a single A100 GPU. Compared to ViT-B, MP3 has significantly lower time and memory cost across different values of the masking ratio η . Compared to MAE, MP3 has favorable efficiency for most of the η values, especially when η is small.

4.2.2. POSITION PREDICTION

Next we examine a Transformer’s ability to solve the position prediction task. We show the results for ImageNet-1K where vary the masking ratio in $\{0, 0.75\}$ and train for 100 epochs. We measure the position prediction accuracy on the validation sets with different test η . The results are shown in Figure 3. Interestingly, when trained with $\eta = 0$, the Transformers are able to solve the task almost perfectly. Large masking ratio $\eta = 0.75$ leads to decreasing accuracy

as expected, but the accuracy remains decent up to a high masking ratio. This suggests that there is enough information in the input patches alone to recover their corresponding position information.

In order to understand the behavior of MP3 with large η , we show one example in Figure 2. Specifically, we obtain a model trained with $\eta = 0.75$, and vary η at test time. For each test η , we generate a random set of context patches, and show the reconstructed images with the predicted positions. We see that the model makes sensible reconstructions, even when the overall accuracy is not high (e.g., with test $\eta = 0.75$). This suggests the model can learn to reason effectively about the underlying objects given only a small, positionless subset of input patches. More examples can be seen in Appendix F.

4.2.3. QUANTITATIVE RESULTS

We report the finetuning accuracy in Table 3 for CIFAR-100 and Table 4 for ImageNet-1K. In all our experiments, MP3 significantly improves upon the supervised training

Table 2. Training time and memory efficiency for MP3, MAE and the ViT-B baseline, while varying the masking ratio η . MP3 has favorable speed and memory efficiency than both MAE and ViT-B in most settings.

η	Time (Seconds / Iter)				Memory (GB / Batch)			
	0.3	0.5	0.75	0.9	0.3	0.5	0.75	0.9
MAE	OOM	0.57	0.47	0.41	OOM	35.1	28.0	24.4
MP3	0.52	0.51	0.46	0.44	30.0	27.4	24.2	22.3
ViT-B		0.67				33.5		

Table 3. Classification results on CIFAR-100 and Tiny ImageNet. We include a strong ResNeXT baseline (Xie et al., 2017; Li et al., 2021) as a reference in both cases. For baseline ViT-S, we train three versions with absolute, relative and no positional embeddings. We also compare to MOCO V3 (Chen et al., 2021) and MAE (He et al., 2021). MP3 achieves much better results than the supervised learning baseline ViT-S, and is comparable to MOCO V3 and MAE with the same number of pretraining epochs. MP3 without PE achieves surprisingly competitive results in both cases.

Method	PT epochs	PE	CIFAR-100 Acc	Tiny ImageNet Acc
ResNeXT	0	conv	82.7	72.2
ViT-S Baseline	0	absolute	73.4	57.6
ViT-S Baseline	0	2D relative	75.0	59.4
ViT-S Baseline	0	none	64.6	60.0
MOCO V3	2K	2D absolute	83.3	73.4
MAE	2K	2D absolute	84.5	73.7
MP3	2K	absolute	84.0	72.8
MP3	2K	2D relative	84.2	73.2
MP3	2K	none	82.6	68.2

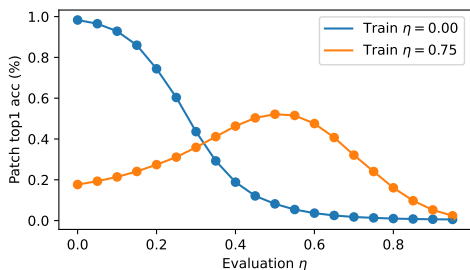


Figure 3. Validation accuracy for the position prediction task on ImageNet-1K for train masking ratios $\eta \in \{0.00, 0.75\}$. The number of total positions is 196. For train $\eta = 0$, the position prediction task can be solved near perfectly at evaluation masking ratio $\eta = 0$ (which is a standard Jigsaw puzzle), and a large η consistently leads to decreasing accuracy. Interesting the converse is true for train $\eta = 0.75$, with a patch performance maximum occurring around evaluation $\eta = 0.55$.

baseline’s accuracy, sometimes by a large margin. Note that we do not change the finetuning hyper parameters, compared to the supervised training baseline, and the gain comes completely from effective pretraining.

Compared to other self-supervised pretraining methods, MP3 achieves comparable results. This is also surprising to some extent, as MP3 does not use or train positional embeddings information in the pretraining phase. We further

performed studies of adding zero initialized relative position biases, similar to BEiT (Bao et al., 2021), and not using PE during finetuning. Relative position bias consistently improves upon the absolute PE version, though with a small margin. Interestingly, the version of not using PE shows strong performance, outperforming all the supervised training baselines (including ones with relative position biases).

Finally, on our largest dataset ImageNet-1K, MP3 requires only 100 pretraining epochs to outperform the supervised training baseline, where the total number of epochs are equated. Due to the large masking ratio ($\eta = 0.75$) and the use of masked attention, this results in an effective reduction of total training costs (see Table 2 for efficiency measures). We have also experimented with a larger backbone ViT-L. With 150 epochs of pretraining, we are able to outperform the supervise training baseline by 1 point, as well as MAE pretrained with 200 epochs (number taken from the paper).

Note that although MP3 does not outperform the state of the art MAE’s performance, we believe that MP3 learns complementary representations. To show this, we performed a simple ensembling test by averaging the outputs of an MP3 and MAE finetuned model from Tab 4 (the ones with 83.0% and 82.7% top 1 accuracy, respectively). This results in a strong classifier with 84.0% accuracy, outperforming MAE pretrained with 1600 epochs. This suggests great potential of potentially combining MP3 and MAE and achieve even greater benefits from pretraining.

Table 4. Classification results on ImageNet-1K. MP3 outperforms the supervised training ViT-B baseline with the same number of total training epochs, which is less overall training cost due to the efficiency of the pretraining phase. Remarkably, MP3 finetuned without any positional information outperforms the full ViT model. MP3’s finetuning performance is on par with competitive methods, while with much fewer pretraining epochs.

Method	PT Epochs	FT Epochs	PE	Acc
ViT-B (Touvron et al., 2021)	0	300	absolute	81.8
ViT-B (Touvron et al., 2021)	0	300	none	79.1
ViT-B DINO (Caron et al., 2021)	300	300	absolute	82.8
ViT-B MOCO V3 (Chen et al., 2021)	300	150	2D absolute	83.2
ViT-B BEiT (Bao et al., 2021)	800	100	2D relative	83.2
ViT-B MAE (He et al., 2021)	1600	100	2D absolute	83.6
ViT-B MAE (He et al., 2021)	150	150	2D absolute	82.7
ViT-B MP3	100	150	absolute	83.0
ViT-B MP3	100	300	none	81.9
ViT-L (He et al., 2021)	0	200	absolute	82.6
ViT-L MAE (He et al., 2021)	200	50	2D absolute	83.3
ViT-L MAE (He et al., 2021)	1600	50	2D absolute	85.1
ViT-L MP3	150	150	absolute	83.6

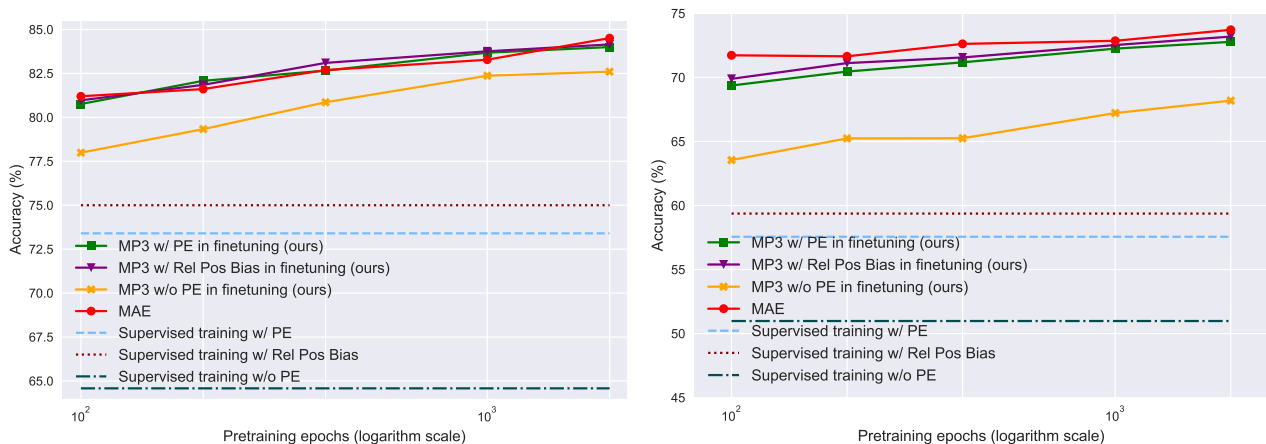


Figure 4. Finetuning accuracy of MP3 on CIFAR-100 (left) and Tiny ImageNet (right) as the pretraining epochs are varied.

4.2.4. ABLATIONS

Pretraining epochs. We vary the total number of pretraining epochs with everything else fixed, and show the resultant accuracy in Figure 4. We see that MP3 works well with a small number of epochs (e.g., 100) but consistently benefits from more pretraining.

Finetuning epochs. For MP3, the position embeddings are not learned or used during pretraining, which suggests that it can potentially benefit from longer finetuning epochs. To see this, we take a ViT-B based MP3 model pretrained at 100 epochs (see Table 4) and vary the finetuning epochs. In Figure 5, we see that this is indeed the case. Moreover, MP3 is able to outperform the supervised training baseline with as few as 60 finetuning epochs (which amounts to 160 total training epochs). This corresponds to an approximate 50% reduction on the training time.

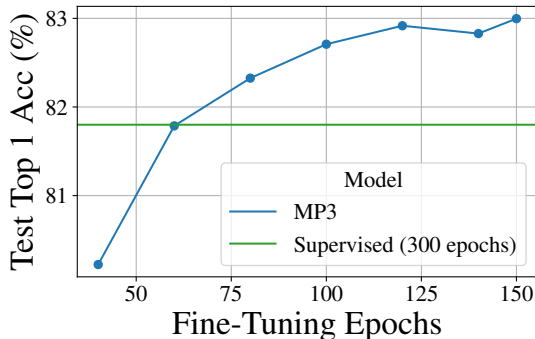


Figure 5. Test top 1 accuracy on ImageNet-1K as the finetuning epochs is varied, with pretraining epochs fixed at 100. MP3 matches the 300 epoch supervised training baseline with as few as 160 total training epochs.

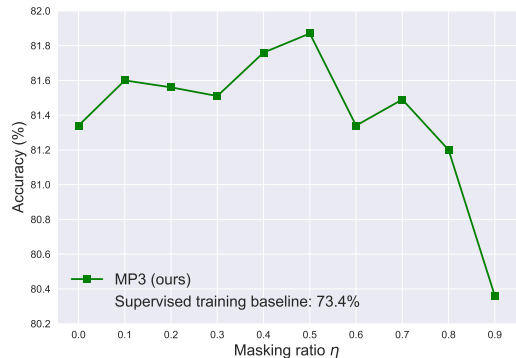


Figure 6. The CIFAR-100 **finetuning** validation accuracy as the pretraining η is varied, with 200 epochs of pretraining. All the settings provide significant improvement to the supervised training baseline, with the performance peaking at 0.5. Extremely large η degrades the performance as less contextual information is learned.

Masking ratio. We evaluate models pretrained with the same number of epochs (200) under different masking ratios. Figure 6 shows that there exists an optimal value that induces the highest finetuning accuracy. Extreme large η leads to notable degradation, which suggests that it is important to train with a reasonably large context token set.

Patch size. For ViTs, the patch size affects the model’s performance. We experimented with two additional patch configurations on CIFAR-100 with the default ViT-S architecture. Figure 7 shows the accuracy for the supervised training baselines and the finetuning results. We see consistent improvements across small and large patch sizes.

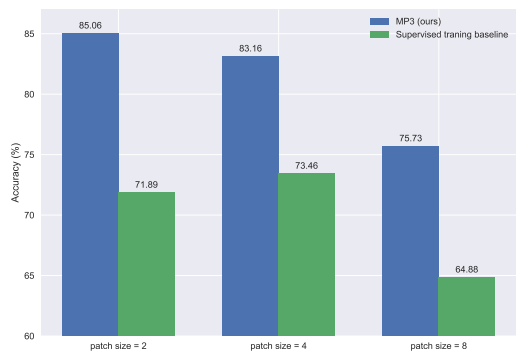


Figure 7. The CIFAR-100 **finetuning** validation accuracy for MP3 (pretrained for 1000 epochs) and the supervised training baselines, as the patch size is varied. MP3 provides consistent improvements under different patch resolutions.

4.2.5. VISUALIZING AND UNDERSTANDING ATTENTION

The improvements demonstrated by MP3 in Section 4 raise two important questions: what are the qualitative properties of the attention mechanism that MP3 learns, and which

Table 5. Comparison with other baselines on Speech Commands (test accuracy %).

Model	Accuracy
TC-ResNet8 (Choi et al., 2019)	96.1
Transformer	91.9
Transformer + MP3	94.2

Table 6. Validation accuracy (%) after finetuning ($\eta = 0$) with different amount of pretraining (PT) updates on Speech Commands with 0.05 fraction of the patches being provided positional information.

1K PT	5K PT	10K PT	25K PT	50K PT
91.7	93.3	94.2	93.1	93.8

aspects are preserved under finetuning?

We observe that, at all layers, MP3 yields heads that are more local, as well as heads that are more global than those found in supervised ViTs. Upon finetuning, head locality becomes more similar to that of a supervised ViT, with early layer locality being much less modified than the locality of later layers. The results for highly local heads at masking ratio $\eta = 0$ are illustrated in Figure 8. For a full unbiased selection and more details, see Appendix E.

4.3. Pretraining and Finetuning on Speech Data

For Google Speech Commands we use a Transformer model with 8 self-attention layers, a dropout of 0.1, feature dimension of 32 and fully connected feedforward layer dimension of 64. The model has around 70K parameters in total to be comparable with the smallest convolutional models from (Choi et al., 2019). All pretraining and finetuning models are trained with exactly the same experimental setting as follows. Optimization is done with Adam (Kingma & Ba, 2015) with a batch size of 256 and early stopping is done based on validation accuracy. Warmup of learning is done for 500 updates with a constant learning rate of 10^{-4} . Subsequently the learning rate is increased to 10^{-3} and dropped by a factor of 2 every 10k updates. For supervised baselines and finetuning phase we also use label smoothing ($=0.1$) for regularization and we train the models for 30K updates.

Compared to Vision, the position prediction task (pretraining step) is very hard – even with $\eta = 0$ the top-1 accuracy is only 4%. Nevertheless, a higher value of top-5 accuracy of 11% demonstrates that the model is able to learn to roughly position the patches but cannot resolve it further. This result shows the difference between image and audio data: different granularity and locality properties. To sim-

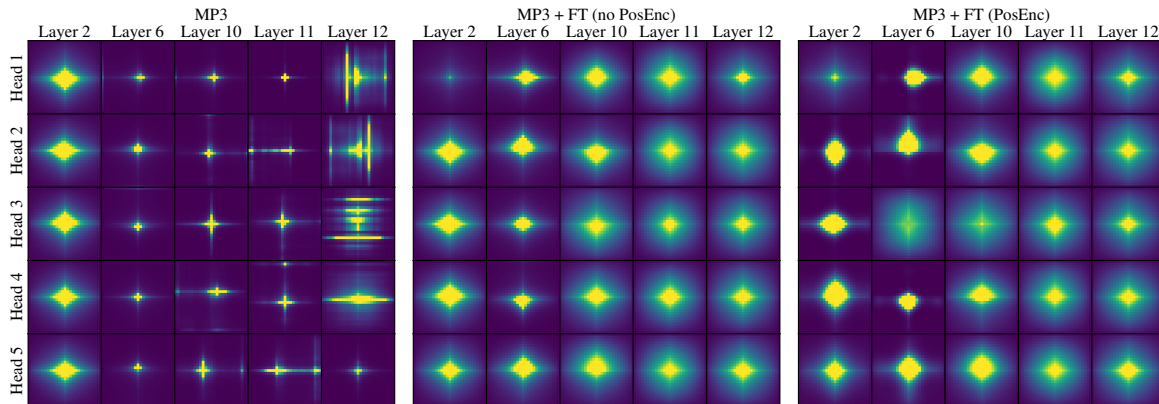


Figure 8. Average relative 2D attention maps for (left) MP3, (center) MP3 + finetuning without positional encoding, and (right) MP3 + finetuning with positional encoding. Both fine tuned models are tuned from the same MP3 model (left), which was trained with masking ratio $\eta = 0$. The heads of the MP3 model are those with the lowest attention entropy $H = -\mathbb{E}_{p_x} \sum_{i=1}^N \sum_{j=1}^M \alpha_{i,j}^{(x)} \log \alpha_{i,j}^{(x)} / N$, and heads of the fine-tuned models are selected to match those of the MP3 model. MP3 learns strong localizations in layers 6, 10 and 11, despite not having access to any explicit positional encoding. Although localization does occur in early layers of supervised models, we do not see early locality in MP3. We expect this is because of the lack of positional encoding, and a context sufficient for localization has not yet been formed. The attention patterns in layer 12 are unlike those of a standard supervised ViT, and we assume they display behaviour specific to the MP3 task. Under both finetuning scenario, the later layers are dramatically altered, whereas the earlier layers are less changed. The primary difference between when using position encoding is that some localization appears in early layers, whereas in the absence of positional encoding, there is not. Each attention map is of size 27×27 , with the class token excluded.

plify the position prediction task, in contrast to vision, we use $\eta = 0$ and, moreover, provide positional information for a randomly chosen 5% of the patches for every sample. Table 6 shows the results achieved with different amounts of pretraining steps of MP3. It can be seen that 5K steps of pretraining is sufficient to improve model accuracy. The test set result for the base validation model above is 94.2% which is 2.3% better than supervised baseline with the same architecture (=91.9%).

5. Conclusions

We have presented MP3, a simple yet effective method for Transformer pretraining. MP3 differs from most other Transformer and token prediction based pretraining method, which bypasses the complexity of designing sophisticated decoder for dense inputs, such as images and speech. MP3 provides competitive performance on small to medium sized data and model sizes, for both Vision and Speech. In particular, MP3 finetuned without position embedding outperform strong supervised training baselines. We also demonstrate the intriguing properties of the position prediction task, which is of independent interest from the pretraining setting. We believe that the strong performing permutation invariant Transformers will be of great interest to the robust ML community.

There are obvious limitations of this work. First of all, MP3 is not designed to produce linearly separable features which many self-supervised methods excel at (e.g., contrastive

learning). Also, despite the high level intuition on sorting input tokens and its relation to semantic understanding, it is not entirely clear how the finetuning benefits from such a pretraining objective. Finally, it is also interesting to test MP3 on NLP applications, and we leave it as future work.

References

- Ahsan, U., Madhok, R., and Essa, I. Video jigsaw: Un-supervised learning of spatiotemporal context for video action recognition. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 179–189. IEEE, 2019.
- Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33, 2020.
- Bao, H., Dong, L., and Wei, F. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.
- Brüel-Gabrielsson, R. and Scarvelis, C. Relative position prediction as pre-training for text encoders. *arXiv preprint arXiv:2202.01145*, 2022.
- Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

- Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.
- Chen, X., Xie, S., and He, K. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021.
- Choi, S., Seo, S., Shin, B., Byun, H., Kersner, M., Kim, B., Kim, D., and Ha, S. Temporal convolution for real-time keyword spotting on mobile devices. *Proc. Interspeech*, 2019.
- Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: 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, 2020.
- Cui, Y., Yang, Z., and Liu, T. Pert: Pre-training bert with permuted language model. *arXiv preprint arXiv:2203.06906*, 2022.
- Dangovski, R., Jing, L., Loh, C., Han, S., Srivastava, A., Cheung, B., Agrawal, P., and Soljačić, M. Equivariant contrastive learning. *arXiv preprint arXiv:2111.00899*, 2021.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pp. 4171–4186, 2019. URL <https://aclweb.org/anthology/papers/N/N19/N19-1423/>.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houshby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=YicbFdNTTy>.
- Dufter, P., Schmitt, M., and Schütze, H. Position information in transformers: An overview. *arXiv preprint arXiv:2102.11090*, 2021.
- El-Nouby, A., Zhai, S., Taylor, G. W., and Susskind, J. M. Skip-Clip: Self-Supervised Spatiotemporal Representation Learning by Future Clip Order Ranking. *arXiv preprint arXiv:1910.12770*, 2019.
- Ghiasi, G., Lin, T.-Y., and Le, Q. V. Dropblock: A regularization method for convolutional networks. *arXiv preprint arXiv:1810.12890*, 2018.
- He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.
- Hoffer, E., Ben-Nun, T., Hubara, I., Giladi, N., Hoefler, T., and Soudry, D. Augment your batch: Improving generalization through instance repetition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8129–8138, 2020.
- Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL <http://arxiv.org/abs/1412.6980>.
- Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.
- Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.
- Lee, H.-Y., Huang, J.-B., Singh, M., and Yang, M.-H. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 667–676, 2017.
- Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pp. 3744–3753. PMLR, 2019.
- Li, S., Liu, Z., Wu, D., Liu, Z., and Li, S. Z. Boosting discriminative visual representation learning with scenario-agnostic mixup. *arXiv preprint arXiv:2111.15454*, 2021.
- Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- Naseer, M., Ranasinghe, K., Khan, S., Hayat, M., Khan, F. S., and Yang, M.-H. Intriguing properties of vision transformers. *arXiv preprint arXiv:2105.10497*, 2021.
- Noroozi, M. and Favaro, P. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pp. 69–84. Springer, 2016.
- Santa Cruz, R., Fernando, B., Cherian, A., and Gould, S. Visual permutation learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):3100–3114, 2018.

- Shaw, P., Uszkoreit, J., and Vaswani, A. Self-attention with relative position representations. *NAACL*, 2018.
- Sinha, K., Jia, R., Hupkes, D., Pineau, J., Williams, A., and Kiela, D. Masked Language Modeling and the Distributional Hypothesis: Order Word Matters Pre-training for Little. *arXiv preprint arXiv:2104.06644*, 2021.
- Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.
- van den Oord, A., Vinyals, O., and Kavukcuoglu, K. Neural discrete representation learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL <https://proceedings.neurips.cc/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf>.
- van den Oord, A., Li, Y., and Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv preprint arXiv:1807.03748*, 2018.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.
- Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., and Bottou, L. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(12), 2010.
- Warden, P. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018. URL <http://arxiv.org/abs/1804.03209>.
- Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated Residual Transformations for Deep Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1492–1500, 2017.
- Xu, D., Xiao, J., Zhao, Z., Shao, J., Xie, D., and Zhuang, Y. Self-supervised spatiotemporal learning via video clip order prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10334–10343, 2019.
- Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6023–6032, 2019.
- Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond Empirical Risk Minimization. *arXiv preprint arXiv:1710.09412*, 2017.
- Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. Random Erasing Data Augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 13001–13008, 2020.

A. Implementation

Algorithm 1 illustrates the MP3 implementation in the pretraining mode. We see that only two simple modifications to the forward pass of a standard Transformer model is needed, which results in a more efficient masked Transformer. The loss is also very easy to compute with the help of a linear prediction head.

Algorithm 1 Pseudo code of MP3 in a PyTorch-like style, where we ignore the ‘cls’ token for simplicity. In the pretraining phase, we first call *mask_sample* to randomly sample the context tokens; the context token indices are then passed to *masked_attention* in each attention block.

```
def mask_sample(x, eta):
    # x: input tokens, shape (batch_size, num_tokens, input_dim)
    # eta: masking ratio in range [0, 1)
    # return kv_ind, indices of context tokens, shape (batch_size, num_context_tokens)
    B, N, D = x.size()
    M = int(N * eta) # number of context tokens
    rand_ind = torch.randn(B, N).argsort(dim=1) # generate a random permutation of positions per input
    kv_ind = rand_ind[torch.arange(B).unsqueeze(1), rand_ind[:, :M]] # get the first M positions per example
    return kv_ind

def masked_attention(x, kv_ind):
    # x: input tokens, shape (batch_size, num_tokens, input_dim)
    # kv_ind: indices of context tokens returned by mask_sample, shape (batch_size, num_context_tokens)
    # return y, output of masked attention
    B, N, D = x.size()
    q = query_proj(x) # apply query projection to all tokens
    k = key_proj(x[torch.arange(B).unsqueeze(1), kv_ind])
    v = value_proj(x[torch.arange(B).unsqueeze(1), kv_ind]) # apply key and value projection to context tokens
    y = multi_head_attention(q, k, v) # perform standard multi-head attention
    return y

def mp3_loss(x):
    # x: output of the Transformer backbone, shape (batch_size, num_tokens, input_dim)
    # return a scalar loss
    B, N, D = x.size()
    targets = torch.arange(N).repeat(B) # the targets is each patch's original position
    # apply a linear projection to get the predictions
    pred = linear_head(x) # shape (batch_size, num_tokens, num_tokens)
    loss = cross_entropy(pred, targets) # classification across all positions
    return loss

def forward(x, eta):
    # x: input tokens (e.g., image patches), eta: masking ratio.
    # x = x + pos_embed -- we do not use position embeddings
    kv_ind = mask_sample(x, eta)
    x = masked_transformer(x, kv_ind) # with masked_attention
    loss = mp3_loss(x)
    return loss
```

B. Transfer Learning Results

We further test MP3’s ability in Transfer Learning. We obtain a ViT-B model pretrained with 150 epochs with $\eta = 0.75$, and finetune it on CIFAR-10 and Stanford Cars (Krause et al., 2013) dataset, which have 50K training examples in 10 classes and 8144 training examples in 196 classes, respectively. We compare with ViT-B, DeiT-B and DINO, all trained with the same architecture. Table 7 shows that MP3 gives competitive performance with supervised and self-supervised models.

Table 7. Transfer learning result on CIFAR-10 and Stanford Cars datasets.

Dataset	ViT (Dosovitskiy et al., 2021)	DeiT-B (Touvron et al., 2021)	DINO (Caron et al., 2021)	MP3
CIFAR-10	98.1	99.1	99.1	98.0
Stanford Cars	-	92.1	93.0	91.8

C. Layerwise KNN probe

For self-supervised learning, k-nearest neighbor classification (KNN) is a popular way of testing the linear separability of the pretrained features (which is similar to linear probing). We perform a study on ImageNet-1K, where we vary the masking ratio η in pretraining, and examine each layer’s average pooled representation with an KNN classifier. The results are shown in Figure 9. We see that all trained layers show significantly higher validation accuracy than a random model.

There is also a positive correlation between η and the peak performance on KNN classification. The optimal layer also appears in the middle, rather than at the very top.

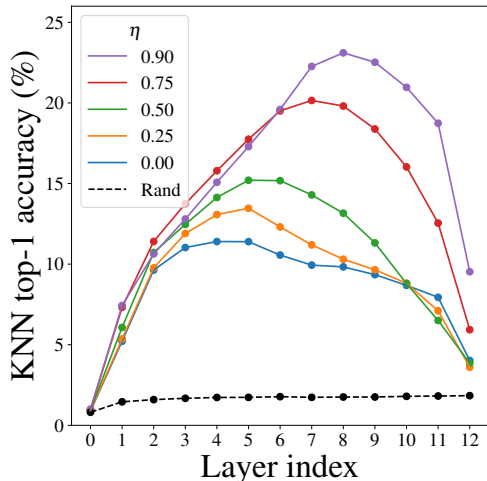


Figure 9. KNN classification accuracy on ImageNet-1K, as the pretraining η and the target layer are varied.

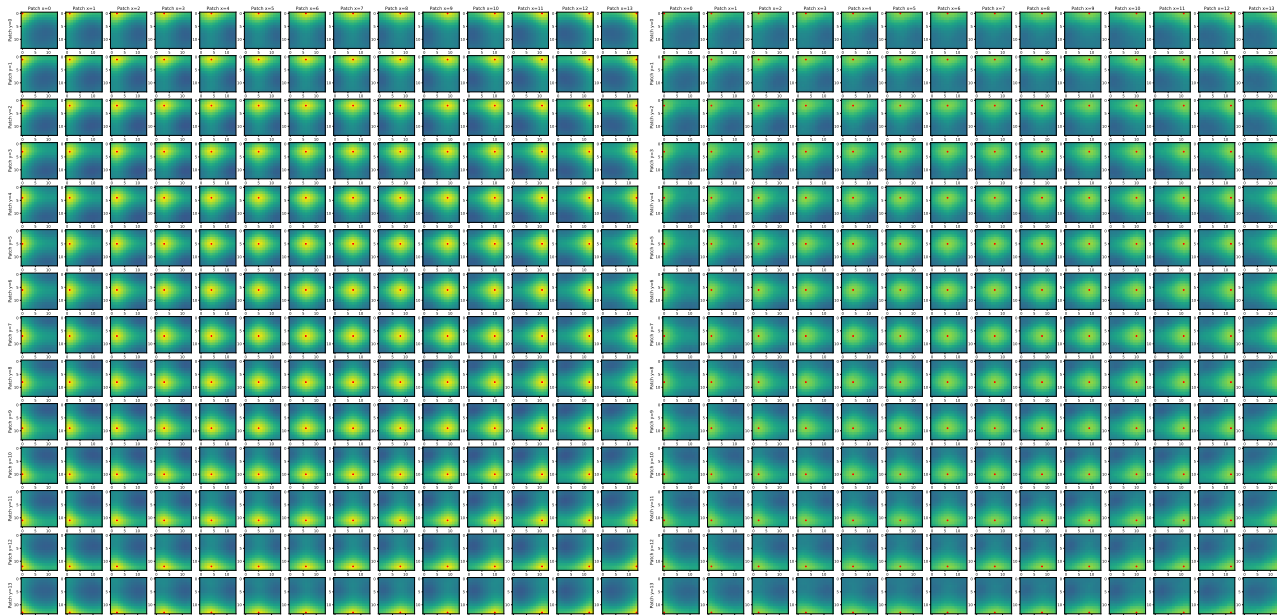


Figure 10. **Left**: average position correlation within the same image; **Right**: average position correlation across random image pairs.

D. Feature Visualization

We also visualize the correlation pattern of the representations within an image, and across images. To do so, we conduct two experiments. In the first one, we compute the Pearson Correlation of the last layer’s representation between each position pair within the same image, averaged across the ImageNet-1K validation set. In the second, we compute the correlation between each position pair of two *random* images. Each experiment results in a correlation matrix of 196×196 , which is reshaped to a $14 \times 14 \times (14 \times 14)$ grid. The results are shown in Figure 10. We see that the representations are biased to their positions. However, there is stronger correlations within the same images than across different ones, which demonstrates that there is an implicit clustering effect of representations within the same image.

E. Relative attention maps

Here we present the full (all layers and heads) relative attention maps for MP3 ($\eta = 0.75$), finetuning with/without positional encoding, and supervised baseline with positional encoding. The results are shown in Figure 11. We see that finetuning drastically changes the locality patterns of the last three layers, while lower layers remain similar.

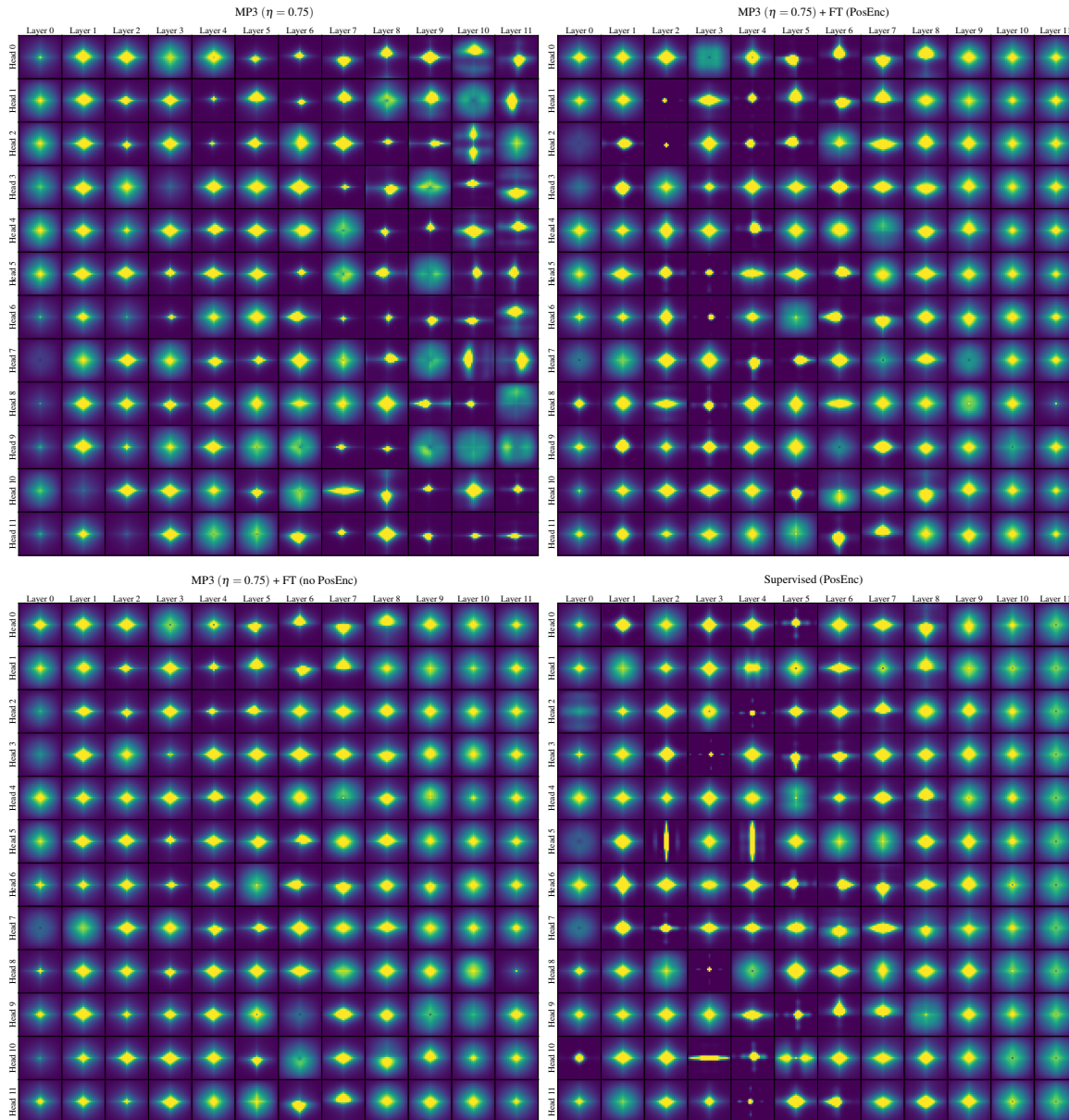


Figure 11. Average relative attention visualization for MP3 pre-trained and finetuned models, compared with the supervised training baseline. **Top left:** MP3 pre-trained; **top right:** MP3 finetuned with PE; **bottom left:** MP3 finetuned without PE; **bottom right:** supervised baseline with PE.

F. ImageNet Reconstruction Visualization

As performed in Figure 2, additional images from the ImageNet validation set were used to test the position prediction of a model trained with $\eta = 0.5$ (Figure 12) and $\eta = 0.75$ (Figure 13). Reconstructions were generated by placing each patch in the predicted position, and patches falling in the same position were averaged. Different η was used at test time, ranging in

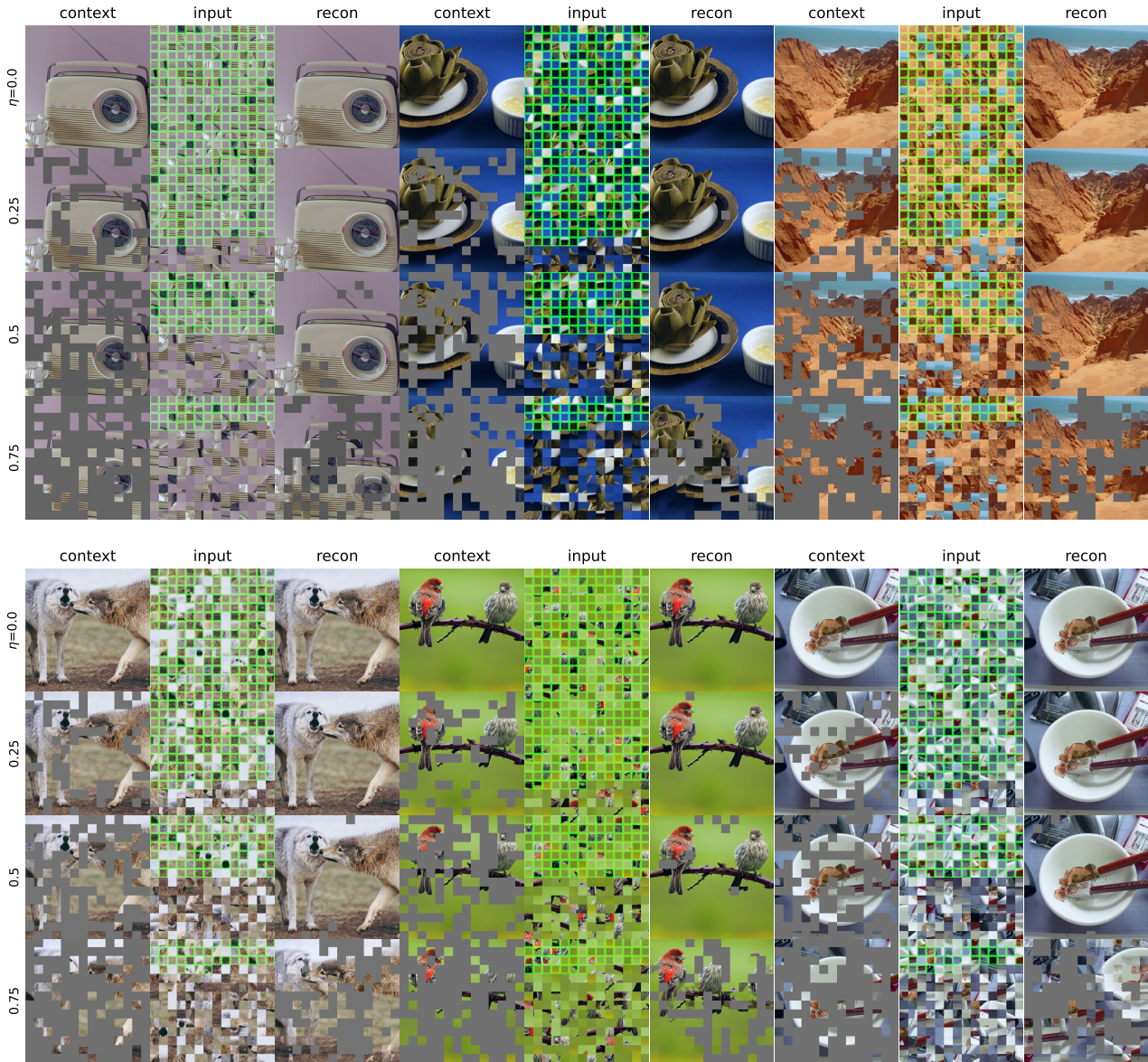


Figure 12. Example reconstructed images from the ImageNet validation set for a model trained with $\eta = 0.5$.

$\{0, 0.25, 0.5, 0.75\}$. In Figure 12 with $\eta = 0.5$, the model can accurately predict majority of the patches for $\eta < 0.75$. In Figure 13 with $\eta = 0.75$, the patches were not placed in the absolute true location, but they were placed in positions that still made sense semantically.

To visualize the role of the context patches with the query patches, patches from two different images in the ImageNet validation set were shuffled together and separated into two distinct sets. A random subset of the patches were used as context patches to predict positions for both context and query patches. The final reconstructions are visualized in Figures 14 and 15. In Figure 14, the two original images of dogs look visually similar in content, composition, and color distribution. The resulting images created a dog-like animal in the center of the image. In Figure 15, two different contents were mixed together: boat in one image and a hot air balloon in another. Similar patches were grouped together creating coherent boat-like structure in one part of the image and a balloon-like structure in another part.

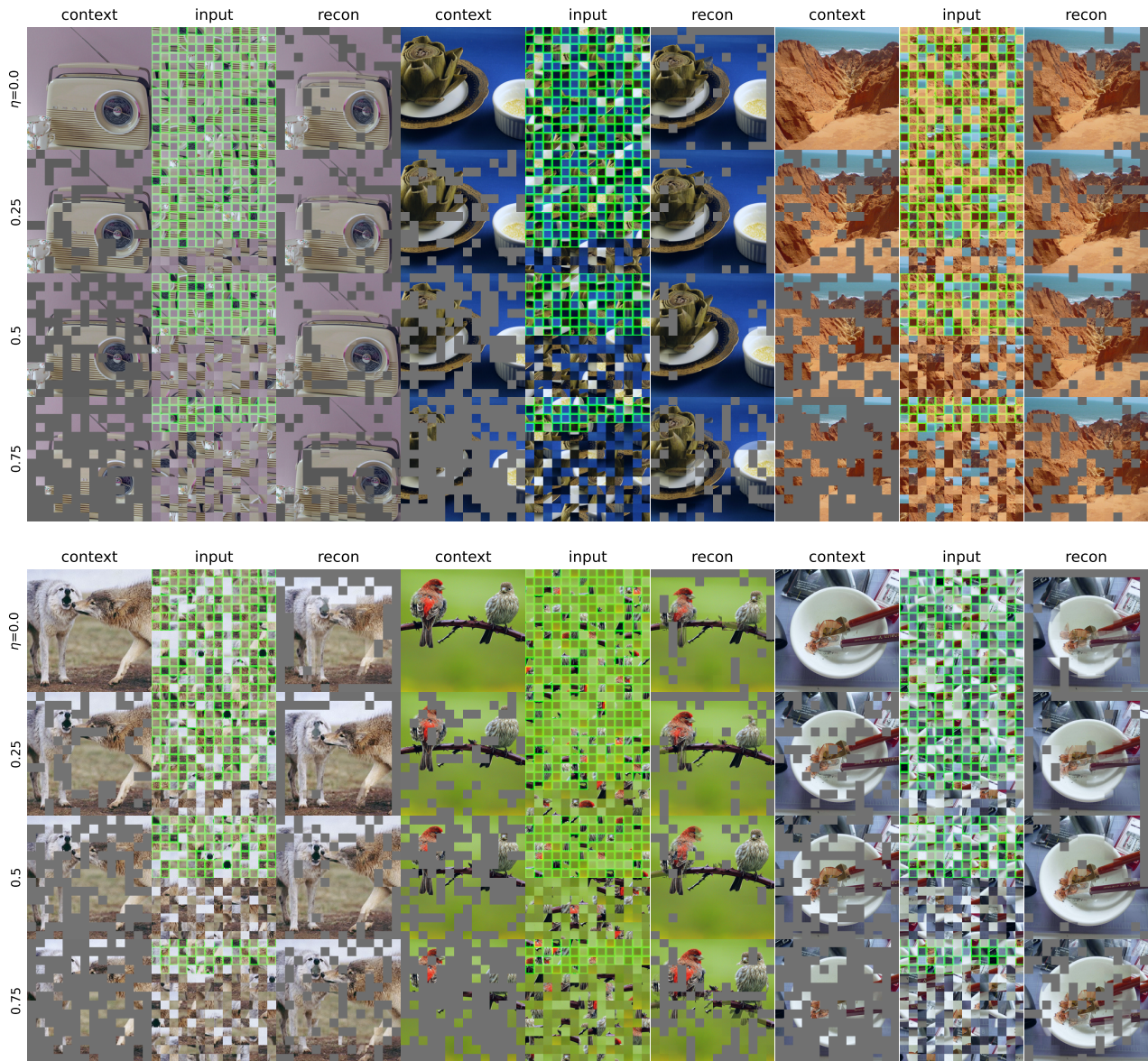


Figure 13. Example reconstructed images from the ImageNet validation set for a model trained with $\eta = 0.75$.

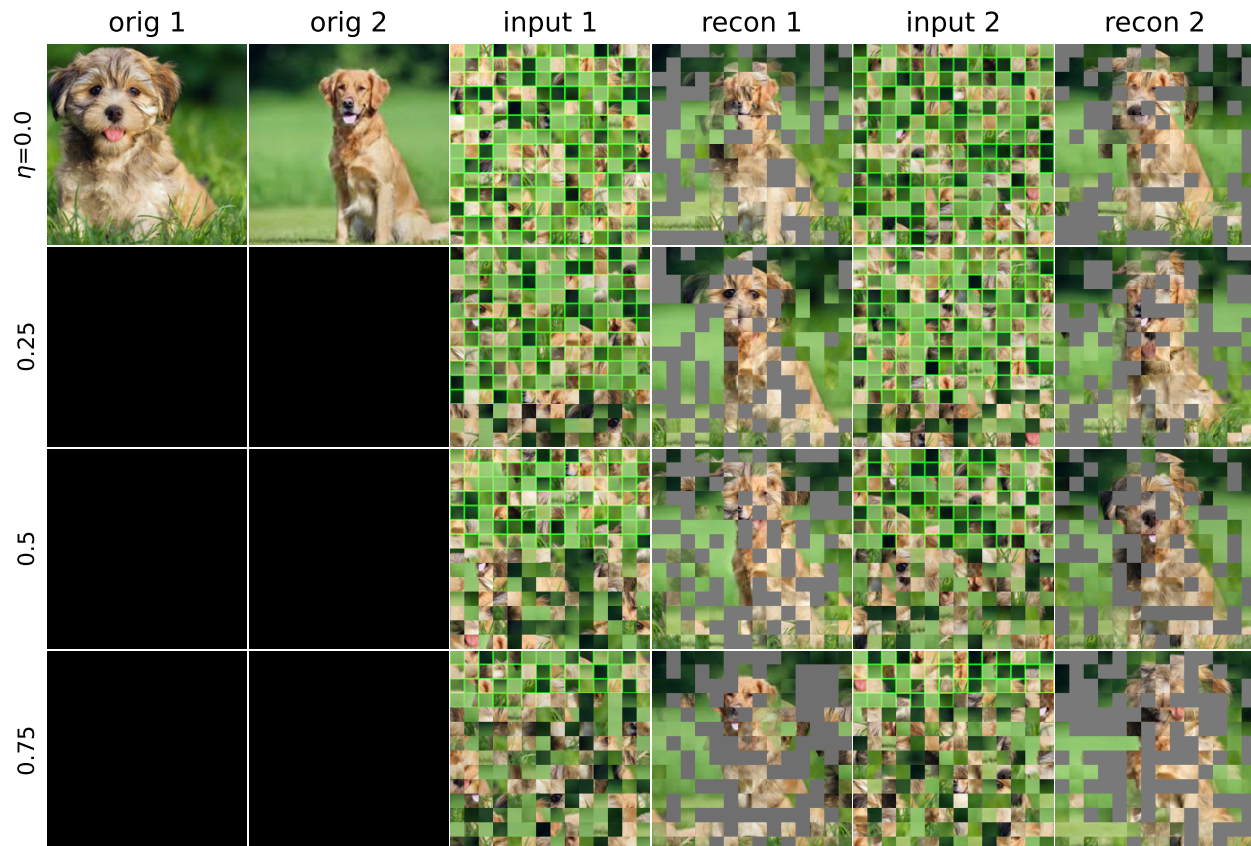


Figure 14. Patches from two images from the ImageNet validation set (top left images) were shuffled together and separated into two distinct sets. **Rows 1–4**: positions were predicted using the shuffled set of patches with different η used at test time, ranging in $\{0, 0.25, 0.5, 0.75\}$. **Columns 3 & 5**: the unordered inputs to the model, with the context patch tokens outlined in green. **Columns 4 & 6**: each patch was placed in the predicted position, and patches falling in the same position were averaged. Coherent dog-like animal can be seen in the final reconstructions with the background placed around the dog.

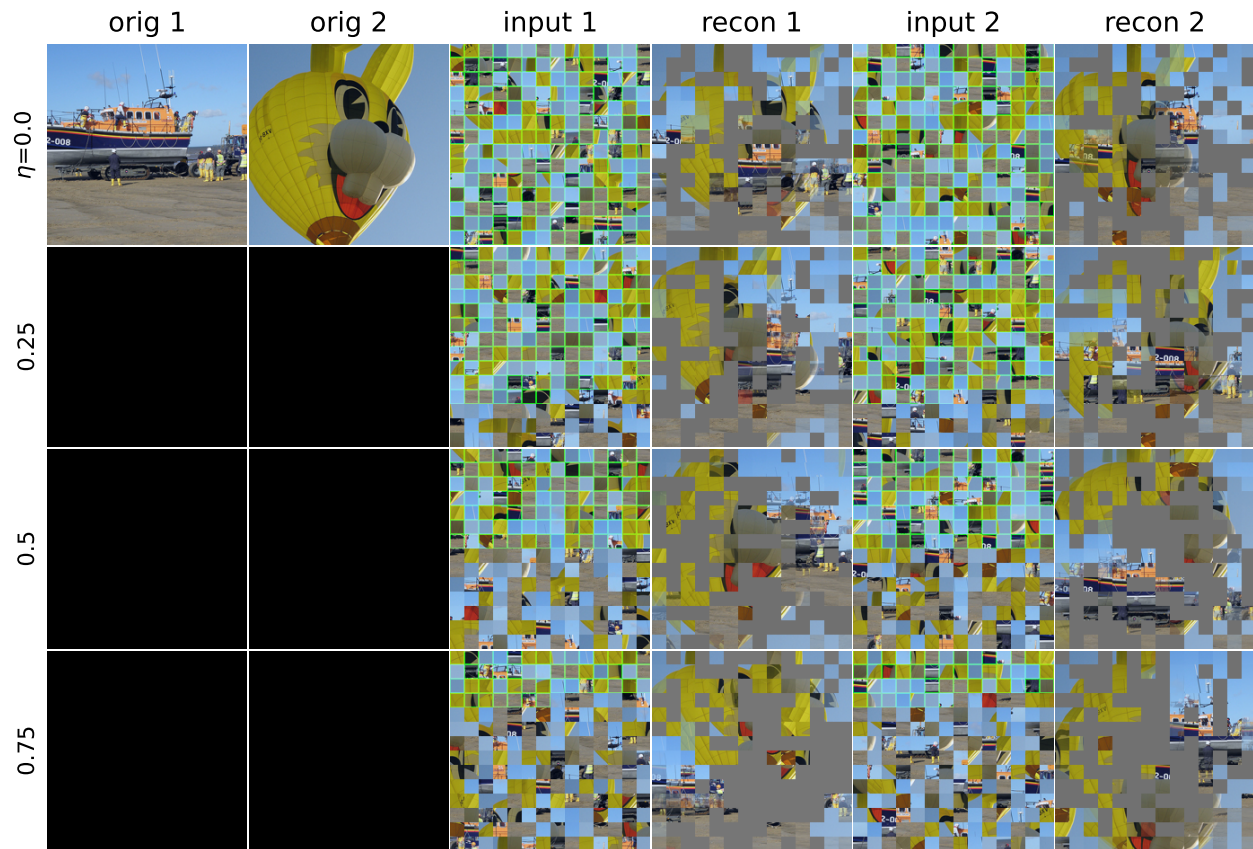


Figure 15. Patches from two images from the ImageNet validation set (top left images) were shuffled together and separated into two distinct sets. **Rows 1–4**: positions were predicted using the shuffled set of patches with different η used at test time, ranging in $\{0, 0.25, 0.5, 0.75\}$. **Columns 3 & 5**: the unordered inputs to the model, with the context patch tokens outlined in green. **Columns 4 & 6**: each patch was placed in the predicted position, and patches falling in the same position were averaged. In this example, similar patches were placed closer together. In the last column of Row 3, a coherent boat-like structure was reconstructed in the lower left region of the image.