MixMAE: Mixed and Masked Autoencoder for Efficient Pretraining of Hierarchical Vision Transformers

Jihao Liu[1,2] Xin Huang[2] Jinliang Zheng[2] Yu Liu[2] ✉ Hongsheng Li[1,3]
[1] CUHK MMLab
[2] SenseTime Research
[3] CPII under InnoHK

Abstract

In this paper, we propose Mixed and Masked AutoEncoder (MixMAE), a simple but efficient pretraining method that is applicable to various hierarchical Vision Transformers. Existing masked image modeling (MIM) methods for hierarchical Vision Transformers replace a random subset of input tokens with a special [MASK] *symbol and aim at reconstructing original image tokens from the corrupted image. However, we find that using the* [MASK] *symbol greatly slows down the training and causes pretraining-finetuning inconsistency, due to the large masking ratio (e.g., 60% in SimMIM). On the other hand, MAE does not introduce* [MASK] *tokens at its encoder at all but is not applicable for hierarchical Vision Transformers. To solve the issue and accelerate the pretraining of hierarchical models, we replace the masked tokens of one image with visible tokens of another image, i.e., creating a mixed image. We then conduct dual reconstruction to reconstruct the two original images from the mixed input, which significantly improves efficiency. While MixMAE can be applied to various hierarchical Transformers, this paper explores using Swin Transformer with a large window size and scales up to huge model size (to reach 600M parameters). Empirical results demonstrate that MixMAE can learn high-quality visual representations efficiently. Notably, MixMAE with Swin-B/W14 achieves 85.1% top-1 accuracy on ImageNet-1K by pretraining for 600 epochs. Besides, its transfer performances on the other 6 datasets show that MixMAE has better FLOPs / performance tradeoff than previous popular MIM methods.*

1. Introduction

Utilizing unlabeled visual data in self-supervised manners to learn representations is intriguing but challenging. Following BERT [12] in natural language processing, pretraining with masked image modeling (MIM) shows great success in learning visual representations for various downstream vision tasks [4, 16, 33, 39, 40], including image classification [11], object detection [25], semantic segmentation [42], video classification [15], and motor control [39]. While those state-of-the-art methods [4, 16] achieved superior performance on vanilla Vision Transformer (ViT) [14, 34], it is still an open question that how to effectively pretrain hierarchical ViT to purchase further efficiencies [8, 10, 26, 28, 35] on broad vision tasks.

In general, existing MIM approaches replace a portion of input tokens with a special [MASK] symbol and aim at recovering the original image patches [4, 40]. However, using [MASK] symbol leads to two problems. On the one hand, the [MASK] symbol used in pretraining never appears in the finetuning stage, resulting in pretraining-finetuning inconsistency [12]. On the other hand, the pretrained networks waste much computation on processing the less informative [MASK] symbols, making the pretraining process inefficient. Those problems become severer when a large masking ratio is used [4, 16, 33, 40]. For example, in SimMIM [40], a masking ratio of 60% is used during the pretraining, i.e., 60% of the input tokens are replaced with the [MASK] symbols. As a result, SimMIM needs relatively more epochs (i.e., 800) for pretraining. In addition, as the high masking ratio causes much pretraining-finetuning inconsistency, the performances of SimMIM on downstream tasks are limited.

In contrast, MAE [16] does not suffer from the above problems by discarding the masked tokens in the encoder and uses the [MASK] symbols only in the lightweight decoder. MAE utilizes the vanilla ViT [14] as the encoder, which can process the partial input efficiently with the self-attention operation. However, the design also limits the application of MAE on hierarchical ViTs as the hierarchical ViTs cannot process 1D token sequences with arbitrary lengths [28, 35].

In this work, we propose MixMAE, a generalized pretraining method that takes advantage of both SimMIM [40] and MAE [40] while avoiding their limitations. In particular, given two random images from the training set, MixMAE creates a mixed image with random mixing masks as input

✉ Corresponding author.

Approach	Compatible with hierarchical ViT	Pretraining efficient	Pretrain-finetune consistent
BEiT [4]	✓	✗	✗
SimMIM [40]	✓	✗	✗
MAE [16]	✗	✓	✓
MixMAE	✓	✓	✓

Table 1. Key differences between MixMAE and related works.

and trains a hierarchical ViT to reconstruct the two original images to learn visual representations. From one image's perspective, instead of replacing the masked tokens of the image with the special $[\mathrm{MASK}]$ symbols, the masked tokens are replaced by visible tokens of the other image. MixMAE adopts an encoder-decoder design. The encoder is a hierarchical ViT and processes the mixed image to obtain hidden representations of the two partially masked images. Before the decoding, the hidden representations are unmixed and filled with the $[\mathrm{MASK}]$ tokens. Following MAE [16], the decoder is a small ViT to reconstruct the two original images. We illustrate the proposed MixMAE in Figure 1.

MixMAE can be widely applied to pretrain different hierarchical ViTs, such as Swin Transformer [28], Twins [8], PVT [35], etc. Thanks to the utilization of the hierarchical architecture, we can naturally apply the pretrained encoder to object detection and semantic segmentation tasks. Empirically, with similar model sizes and FLOPs, MixMAE consistently outperforms BEiT [4] and MAE [16] on a wide spectrum of downstream tasks, including image classification on iNaturalist [18] and Places [41], object detection and instance segmentation on COCO [25], and semantic segmentation on ADE20K [42]. By abandoning using $[\mathrm{MASK}]$ tokens in the encoder, MixMAE shows much better pretraining efficiency than SimMIM [40] on various hierarchical ViTs [8, 28, 35].

2. Related Works

Inspired by BERT [12] for Masked Language Modeling, Masked Image Modeling (MIM) becomes a popular pretext task for visual representation learning [2, 4, 16]. MIM aims to reconstruct the masked tokens from a corrupted input. Current MIM approaches can be divided into two categories by the reconstruction targets. SimMIM [40] points out that raw pixel values of the randomly masked patches are a good reconstruction target and a lightweight prediction head is sufficient for pretraining. Different from SimMIM, MAE [16] only takes the visible patches as the input of the encoder. Mask tokens are added in the middle of the encoder and the decoder. Such an asymmetric design greatly reduces the computation overhead of the encoder. To further enhance the feature extraction capability of the encoder, CAE [6] separates the encoder and decoder explicitly by adding a feature alignment module in the middle of them. Jean-Baptiste et al. [1] propose to learn representations by reconstructing original videos from synthetically mixed ones.

Instead of building the reconstruction target manually, using a network to generate the reconstruction target has also been widely applied. In such works, an image tokenizer is used to transform an image into visual tokens. BEiT [4] utilizes a pretrained discrete VAE (dVAE) [30, 31] as the tokenizer. However, the originally used MSE loss in dVAE is insufficient to force the tokenizer to capture high-level semantics. PeCo [13] proposed to apply perceptual similarity loss on the training of dVAE can drive the tokenizer to generate better semantic visual tokens, which helps pretraining. Moreover, the tokenizer in BEiT [4] needs to be offline pretrained, which limits the model's adaption ability. To address the problem, iBOT [43] proposed to use an online tokenizer to generate the visual tokens.

There are also concurrent works that explore using MAE on hierarchical Vision Transformers. UM-MAE [22] proposed a new masking strategy for adapting MAE to pretrain pyramid-based ViTs (e.g., PVT [35], Swin [28]). GreenMIM [20] also adapt MAE on hierarchical architectures. It partitions the local windows into several equal-sized groups and proposes an optimal grouping algorithm to find the optimal group size. Instead of designing specific masking or grouping strategies, we focus on rearranging the inputs and targets. Empirical evaluation shows that MixMAE can obtain better performance with various hierarchical architectures.

3. Methodology

In this section, we introduce the proposed MixMAE for learning visual representations via Masked Image Modeling (MIM). We start by briefly revisiting MIM, and then introduce how MixMAE creates training inputs and performs image reconstruction, as well as the hierarchical Transformer architecture. Finally, we present how to reduce the difficulty of the pretext task to improve the pretraining efficiency.

3.1. A Revisit of Masked Image Modeling

Following BERT [12], recent works [4, 16, 40] proposed MIM for learning visual representations. Given an input image x, MIM firstly divides the image into non-overlapping image patches x^p, following ViT [14]. It then samples a random mask M to mask a portion of the image patches, and fills the masked place with a special symbol $[\mathrm{MASK}]$, $\hat{x}^p = x^p \odot M + [\mathrm{MASK}] \odot (1 - M)$, where $\odot$ denotes element-wise multiplication. The masked image $\hat{x}^p$ is processed by an image encoder to produce the latent representations, and a lightweight decoder (head) is utilized to reconstruct the original image based on the latent representations. The reconstruction target can be chosen as the normalized raw pixel [16, 40] or visual tokens [4, 13]. MIM computes the mean squared error (MSE) between the reconstructed image patches y^p and the original image patches x^p as the

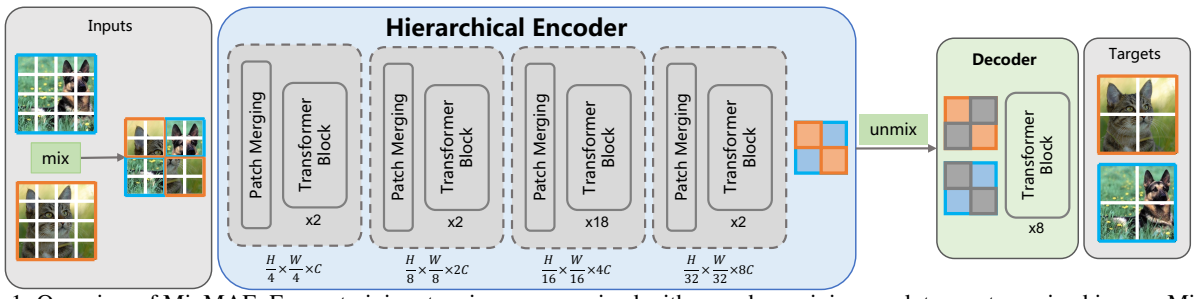

Figure 1. Overview of MixMAE. For pretraining, two images are mixed with a random mixing mask to create a mixed image. MixMAE takes the mixed image as input and reconstructs the two original images. Right before decoding, the token embeddings are unmixed and filled with mask tokens for dual reconstruction of the two original images.

reconstruction loss, $\mathcal{L}_{rec} = \|(y^p - x^p) \odot (1 - \mathrm{M})\|_2^2$, which is only calculated on masked patches [16]. After pretraining, the decoder is discarded and the encoder is used for further finetuning on downstream visual tasks.

3.2. Mixed and Masked Autoencoder (MixMAE)

While previous MIM works achieved great progress in self-supervised visual representation pretraining, they usually require a large number of epochs for pretraining. One reason is that they waste much computation on processing the less informative [MASK] symbols. Besides, using the [MASK] symbol also causes pretraining-finetuning inconsistency as those symbols never appear during finetuning. To tackle the issues, we create mixed images as training inputs from pairs of unlabelled training images, which are generated by mixing two groups of visible tokens from two images, for pretraining. The mixed input is processed by MixMAE to reconstruct original images simultaneously. For better transferring the learned multi-scale representations to downstream tasks, we utilize the popular Swin Transformer with larger-window size as the encoder of the proposed MixMAE [27, 28]. Figure 1 illustrates the proposed framework.

Mixed Training Inputs. Given two sets of image patches $\{x_1^p, x_2^p\}$ of two random training images, we create a mixed image by filling each spatial location with the corresponding visual token from either x_1^p or x_2^p. The mask notation M is slightly abused and we denote $M = 1$ as choosing a token from x_1^p and vice versa. The mixed training image $\hat{x}_m^p$ is therefore formulated as:

$$\hat{x}_m^p = x_1^p \odot \mathrm{M} + x_2^p \odot (1 - \mathrm{M}). \tag{1}$$

MixMAE then takes the mixed image as input for reconstruction during pretraining. The mixed image no longer consists of the extra [MASK] symbol and only actual visual tokens, leading to better performances on downstream tasks. The design shares the same principle of MAE [16], but our approach does not disassemble the structure of the 2D image, making it more flexible for adapting to various visual backbones, such as PVT [35] and Swin Transformer [28]. We can conduct better pretraining based on various hierarchical vision architectures. We follow common practices to use random masking strategy [16, 40].

Hierarchical Vision Transformer. For better encoding multi-scale representations, we build the encoder of MixMAE with popular Swin Transformer [28] and use larger window size [27] to encode more context for better reconstruction.

Following Swin Transformer, the input is split into non-overlapping image patches and processed by a linear projection layer. Then the image patches added with positional embeddings are processed by 4 stages of Transformer blocks to produce hierarchical representations, with a downsampling layer between every two successive stages. However, we do not use the complicated shifted window for information propagation across non-overlapping windows. Instead, we use a relatively large window size (i.e., 14×14), and only conduct global self-attention in stage-3 and -4 [1]. The larger window size brings negligible computation overhead, but can better integrate the global context. As we usually use a large masking ratio in MIM, the global context is important for better reconstruction.

We scale the encoder of MixMAE following [27] with configuration parameters listed bellow:

- Base: $C = (128, 256, 512, 1024)$, $H = (4, 8, 16, 32)$, $B = (2, 2, 18, 2)$,
- Large: $C = (192, 384, 768, 1536)$, $H = (6, 12, 24, 48)$, $B = (2, 2, 18, 2)$,
- Huge: $C = (352, 704, 1408, 2816)$, $H = (11, 22, 44, 88)$, $B = (2, 2, 18, 2)$,

where C, H, and B denote the channel numbers, numbers of the attention heads, and the numbers of blocks for each stage. The window size is set to 14×14 / 7×7 for stage-1, -2, and -3 / -4 during pretraining. A linear layer is added between the encoder and the decoder to convert the embedding dimension of the encoder's output to 512.

[1] The feature map resolution is 14×14 / 7×7 for stage-3 / -4. A 14×14 / 7×7 window attention is equivalent to global self-attention.

Dual Reconstruction. After encoding the mixed input, we *unmix* the token embeddings into two groups according to the binary mask M. We then add the [MASK] tokens to reconstruct the original two images from the two groups with the decoder, which has 8 Transformer blocks with an embedding dimension of 512. The loss is therefore set as

$$\mathcal{L}_{rec} = \|(y_1^p - x_1^p) \odot (1 - \mathrm{M})\|_2^2 + \|(y_2^p - x_2^p) \odot \mathrm{M}\|_2^2, \quad (2)$$

where y_1^p and y_2^p are the reconstructed images corresponding to x_1^p and x_2^p, respectively. The intuition behind is that as the mixed input contains tokens from two images, we can fully utilize them by reconstructing both images to pretrain the neural network. The computation overhead of reconstructing both images is negligible as the decoder is lightweight. Our approach demonstrates much higher efficiency than previous works, as to be introduced in Section 5.

3.3. Reducing the Difficulty of the Pretext Task

Although the dual reconstruction (Eq. (2)) enjoys several benefits, it is a much more challenging optimization problem due to the mixing of image tokens, which causes slow convergence in our preliminary experiments. To reduce the optimization difficulty, we facilitate the dual reconstruction by exploring the following approaches.

- **Mix embedding:** Besides the positional embeddings, we add two mix embeddings to the visual tokens to implicitly differentiate the two mixing groups. Each mix embedding is a vector and is shared for tokens from the same image. In practice, we use different mix embeddings for the 4 stages of the encoder and add the embedding at the beginning of each stage.
- **Masked self-attention:** Thanks to the flexibility of the self-attention mechanism, we can also differentiate two mixing images explicitly by masking the self-attention fields. Specifically, for each token, it is only allowed to aggregate information from the tokens belonging to the same image (group). We implement the masked self-attention by reusing the mixing mask M described in Section 3.2. Note that we upsample the mask M by nearest interpolation at different stages to match the resolution of the feature map.

Both approaches do not introduce much computation overhead or extra parameters. The empirical results show that both approaches can help obtain better results. However, the second approach also leads to a faster convergence speed, which is crucial for large-scale pretraining. We use the second approach by default and ablate the design in Section 6.

4. Experimental Setup

We validate our proposed MixMAE by conducting experiments with pretraining-then-finetuning strategy, following previous practices [4, 16]. In particular, we use ImageNet-1K [11] as the training set for self-supervised pretraining. We then finetune the encoder of MixMAE to downstream tasks, including image classification on ImageNet-1K [11], iNaturalist [18], and Places [41], object detection and instance segmentation on COCO [25], and semantic segmentation on ADE20K [42].

Pretraining on ImageNet-1K. We conduct self-supervised pretraining on ImageNet-1K [11]. By default, we pretrain for 600 epochs with the input size of 224×224. The window size is set as 14×14 for the first 3 stages and 7×7 for stage-4. The patch size of the mask is set to 32×32 as our hierarchical encoder eventually downsamples the input to $\frac{1}{32}$ of the input resolution. Following MAE [16], a masking ratio of 75% is used by default, which is implemented by mixing 4 images. We follow all other pretraining hyperparameters of those in MAE [16] for a fair comparison.

Finetuning for image classification. We conduct supervised finetuning with the pretrained encoder of our MixMAE on image classification tasks, including ImageNet-1K [11], Places [41], and iNaturalist [18]. We follow previous practices [4, 16] and use a layer-wise learning-rate decay strategy [9] for finetuning. We sweep the decay rate in $\{0.7, 0.75, 0.8\}$, and report the best-performing results. We use drop path regularization [19], and set the drop rate to 0.15/0.2/0.3 for Swin-B/L/H, respectively. We finetune Swin-B/L/H for 100/50/50 epochs following MAE [16].

Finetuning on COCO. We perform supervised finetuning on COCO [25] for object detection and instance segmentation using the Mask RCNN framework [17] with our pretrained encoder as the backbone. We reuse the training setup in MAE [16] for a fair comparison. We change the window size to 16×16 for being divisible by the input 1024×1024 resolution. Besides, we change the window sizes of the 6th-, 12th-, and 18th-block in stage-3 to 32×32 for cross-window interactions following the previous practice [23].

Finetuning on ADE20K. We perform supervised finetuning on ADE20K [42] for semantic segmentation. We use the UperNet [38] framework with our pretrained encoder as its backbone. We also change the window size as mentioned above. We reuse the training setup in BEiT [4] for a fair comparison.

We include more details about pretraining and finetuning in the Appendix.

5. Main Results

In this section, we compare our MixMAE to prior arts on various visual benchmarks. We present the results on ImageNet-1K [11] in Section 5.1, and then show the results on the other 6 benchmarks in Section 5.2. Note that all the results of MixMAE are obtained by conducting supervised finetuning of the encoder with self-supervised pretraining, without extra intermediate finetuning [4].

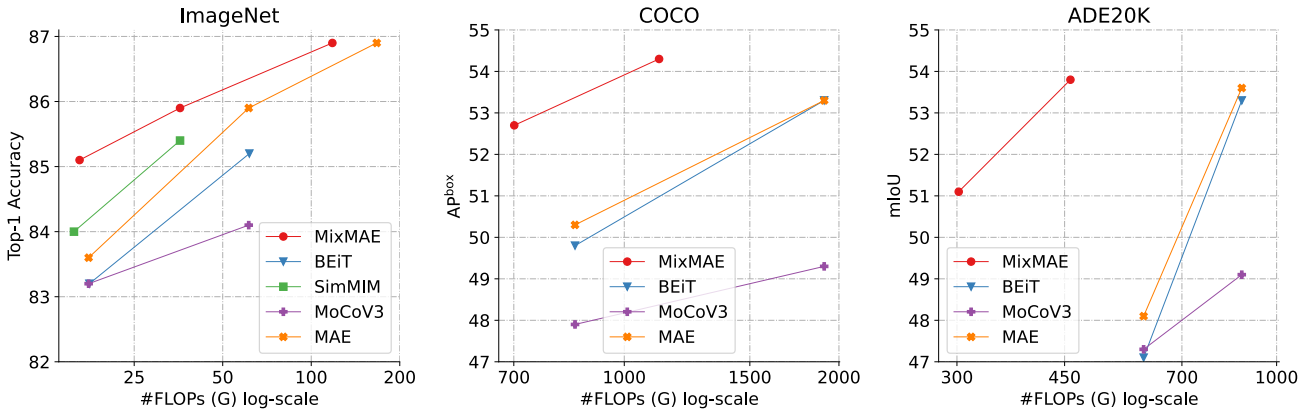

Figure 2. Tradeoffs of FLOPs vs. (left) top-1 accuracy on ImageNet-1K, (middle) AP^{box} on COCO, (right) and mIoU on ADE20K. All results are from various self-supervised pretraining methods followed by supervised finetuning. All entries on COCO [25] use Mask RCNN [17] framework. All entries on ADE20K [42] use UperNet [38] framework. Note that this comparison confounds differences in architecture and pretraining strategy.

Method	Backbone	FLOPs (G)	Param. (M)	Supervision	Pretrain Epochs	FT	LIN
ViT [14]	ViT-B	17.5	86	RGB	14 †	79.9	-
BEiT [4]	ViT-B	17.6	87	DALL-E	800	83.2	37.6
CAE [6]	ViT-B	17.5	86	DALL-E	800	83.6	68.6
MaskFeat [36]	ViT-B	17.5	86	HOG	300	83.6	-
data2vec [3]	ViT-B	17.5	86	Feature	800	84.2	-
iBOT [43]	ViT-B	17.5	86	Momentum	1600	84.0	79.5
PeCo [13]	ViT-B	17.5	86	MoCo v3	800	84.5	-
MAE [16]	ViT-B	17.5	86	RGB	1600	83.6	67.8
MAE$^\diamond$ [16]	Swin-B/W14	16.3	88	RGB	600	84.4	61.0
EsViT‡ [21]	Swin-B/W14	16.3	87	Momentum	300	83.9	**81.3**
SimMIM [40]	ViT-B	17.5	86	RGB	800	83.8	56.7
SimMIM [40]	Swin-B	15.6	88	RGB	800	84.0	-
SimMIM$^\diamond$ [40]	Swin-B/W14	16.3	88	RGB	300	84.1	20.2
GreenMIM [20]	Swin-B	15.6	88	RGB	800	83.8	-
GreenMIM [20]	Swin-B/W14	16.3	88	RGB	800	84.1	-
MixMAE	Swin-B	15.6	88	RGB	600	84.6	61.2
MixMAE	Swin-B/W14	16.3	88	RGB	300	84.8	63.8
MixMAE	Swin-B/W14	16.3	88	RGB	600	**85.1**	71.0

Table 2. Comparison with state-of-the-art MIM methods. All entries are results of base-level models and have comparable model sizes. We report the finetuning accuracy on ImageNet-1K. The FLOPs and Params. are calculated for the encoders. † denotes the number of epochs is based on JFT [32]. ‡ results are from [37]. $\diamond$ denotes our implementation with the official code. FT and LIN denote top-1 accuracy on ImageNet-1K after finetuning and linear probing respectively.

5.1. Results on ImageNet-1K

Comparisons with other MIM approaches. Table 2 presents the comparison between MixMAE and state-of-the-art Masked Image Modeling (MIM) works. Our MixMAE can obtain higher accuracy while requiring fewer epochs for pretraining. In particular, we achieve 84.8% top-1 accuracy with 300 epochs of pretraining, 1.6% better than BEiT [4] with 62.5% fewer epochs for pretraining. Besides, our MixMAE also enjoys longer pretraining as previous methods [16]. Specifically, we obtain strong 85.1% top-1 accuracy with only 600 epochs of pretraining with Swin-B/W14.

While previous works design various reconstruction targets to speed up the pretraining process [4, 36], our MixMAE reconstructs simply normalized pixels [16] and demonstrates strong pretraining efficiency. Compared to MaskFeat [36], our MixMAE obtains +1.2% better accuracy with the same pretraining epochs. PeCo [13] proposed to recon-

Pretrain Method	Backbone	Pretrain Epochs	Finetune Epochs	Top-1 Acc.
Supervised [28]	Swin-B	-	300	83.5
SimMIM [40]	Swin-B	800	100	84.0
GreenMIM [20]	Swin-B	800	100	83.8
MixMAE	Swin-B	600	100	**84.6**
Supervised [40]	Swin-L	-	300	83.5
SimMIM [40]	Swin-L	800	100	85.4
GreenMIM [20]	Swin-L	800	100	85.1
MixMAE	Swin-L	600	50	**85.9**
Supervised [35]	PVT-L	-	300	81.7
SimMIM†	PVT-L	800	100	82.0
MixMAE	PVT-L	600	100	**83.4**
Supervised [8]	Twins-SVT-L	-	300	83.7
SimMIM†	Twins-SVT-L	800	100	83.3
GreenMIM [20]	Twins-SVT-L	800	100	**83.9**
MixMAE	Twins-SVT-L	600	100	**83.9**

Table 3. Comparison with state-of-the-art MIM methods using the same encoder. We report the finetuning accuracy on ImageNet-1K. † denotes our implementation with the official code using input size of 224×224.

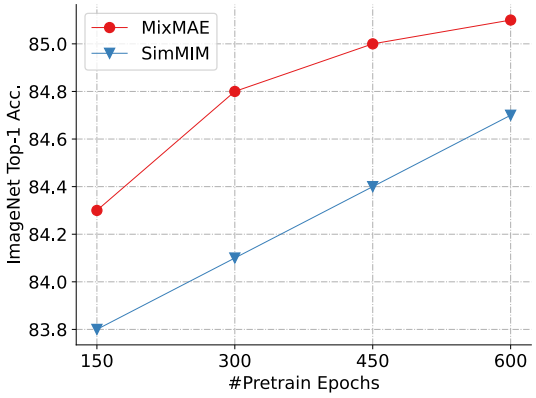

Figure 3. Efficiency comparison between MixMAE and SimMIM. We report the finetuning accuracy on ImageNet-1K. The encoder is Swin-B/W14 with input size of 224×224.

struct the perceptual codebook from a pretrained MoCo v3 network [7] and can achieve better performance to some extent. In comparison, our MixMAE obtains even better performance (+0.6%) than PeCo with fewer pretraining epochs (-200). The superior performance of MixMAE comes from our mixed pretraining as well as the hierarchical Vision Transformer. However, SimMIM [40] also utilizes a hierarchical Swin Transformer [28], but its performance is worse (-1.1%) than MixMAE with more pretraining epochs (+200). We conduct a thorough comparison with SimMIM by using the same encoder Swin-B/W14 and show the results in Figure 3. Our proposed MixMAE shows much better pretraining efficiency than SimMIM. We list the training time in the Appendix.

We test with scaling MixMAE up to 600M parameters, as shown in Figure 2 (Left). MixMAE has better FLOPs vs. accuracy tradeoff than other approaches. In particular, MixMAE pretrained Swin-L and -H achieve 85.9% and 86.9% top-1 accuracy, respectively, being comparable with MAE-L (85.9%) and -H (86.9%) but requiring fewer FLOPs for inference (-40% for -L and -30% for -H).

Integrating MixMAE to other backbones. While previous works [16,36] may be restricted to a specific architecture, our proposed MixMAE can generalize to various visual backbones, including Swin Transformer [28], Twins [8], and PVT [35]. We conduct a thorough comparison with other MIM approaches with fixed encoders. As shown in Table 3, MixMAE consumes the same or fewer epochs for pretraining but obtains consistently better performance on hierarchical ViTs. In particular, our MixMAE achieves 84.6% top-1 accuracy with Swin-B, +0.6% better than SimMIM [40] while requiring 200 fewer epochs for pretraining. With Swin-L, our MixMAE obtains 85.8% top-1 accuracy with 600 epochs of pretraining and 50 epochs of finetuning, showing higher efficiency than SimMIM. Besides, our MixMAE achieves 83.2% top-1 accuracy with PVT-L [35], improving the supervised baseline by a non-trivial margin.

5.2. Results of Transferring to Downstream Tasks

To further demonstrate the effectiveness of the visual representations learned by MixMAE, we transfer MixMAE to various visual benchmarks with settings described in Section 4.

Object detection and instance segmentation. We show the results on COCO [25] in Table 4. By pretraining for 600 epochs on ImageNet-1K, we achieve 52.7 AP^{box} and 47.0 AP^{mask} with Swin-B, surpassing previous self-supervised approaches with fewer FLOPs and parameters. Compared to BEiT [4], our MixMAE obtains higher AP^{box} (+2.9) and AP^{mask} (+2.6) with less pretraining epochs (-200). Note that our hierarchical backbone can naturally be transferred to object detection without re-designing [23] network architectures such as FPN [24].

Our MixMAE can also scale up to larger models in object detection task and obtains better performance. As shown in Table 4, we achieve 54.3 AP^{box} (48.2 AP^{mask}) with Swin-L, +1.0 (+1.1) better than BEiT while requiring 200 fewer epochs for pretraining and 41% fewer FLOPs for inference. We further evaluate the tradeoff of FLOPs vs. AP^{box} in Figure 2 (Middle). We also found that MixMAE outperforms other approaches by large margins.

Semantic segmentation. Table 4 also presents the results of MixMAE on ADE20K [42]. We compare its Mean Intersection over Union (mIoU) on ADE20K with other self-supervised approaches. Our pretrained Swin-B achieves 51.1 mIoU, +4.0 better than BEiT while requiring only half of FLOPs for inference. Besides, we obtain 53.8 mIoU by

Method	Backbone	Pretrain Epochs	FLOPs (G)	Params. (M)	COCO AP^{box}	COCO AP^{mask}	FLOPs (G)	Params. (M)	ADE20K mIoU
MoCo v3 [7]	ViT-B	300	853	116	47.9	42.9	606	164	47.3
BEiT [4]	ViT-B	800	853	116	49.8	44.4	606	164	47.1
MAE [16]	ViT-B	1600	853	116	50.3	44.9	606	164	48.1
iBOT [43]	ViT-B	1600	-	-	51.2	44.2	-	-	50.0
EsViT [40]	Swin-B	300	-	-	-	-	-	-	47.3
SimMIM [40]	Swin-B	800	-	-	52.3	-	-	-	52.8†
SimMIM ◇ [40]	Swin-B/W14	300	701	110	51.1	45.4	302	122	48.9
GreenMIM [20]	Swin-B	800	-	-	50.0	44.1	-	-	-
MixMAE	Swin-B/W14	300	701	110	52.3	46.4	302	122	49.9
MixMAE	Swin-B/W14	600	701	110	**52.7**	**47.0**	302	122	**51.1**
MoCo v3 [7]	ViT-L	300	1907	339	49.3	43.9	877	392	49.1
BEiT [4]	ViT-L	800	1907	339	53.3	47.1	877	392	53.3
MAE [16]	ViT-L	1600	1907	339	53.3	47.2	877	392	53.6
SimMIM [40]	Swin-L	800	-	-	53.8	-	-	-	53.5†
MixMAE	Swin-L	600	1119	319	**54.3**	**48.2**	460	236	**53.8**

Table 4. Comparison with other self-supervised approaches on COCO and ADK20K. We report AP^{box} and AP^{mask} on COCO, and mIoU on ADE20K. The results of BEiT and MoCo v3 are from MAE [16]. The results of EsViT are from [37]. † denotes using supervised finetuning on ImageNet. ◇ denotes our implementation with the official code.

Method	Backbone	FLOPs (G)	Params. (M)	INat2018	INat2019	Places205	Places365	Average
DINO [5]	ViT-B	17.5	86	72.6	78.2	-	-	-
MAE [16]	ViT-B	17.5	86	75.4	80.5	63.9	57.9	69.4
MixMAE	Swin-B/W14	16.3	88	**78.2**	**83.3**	**68.6**	**59.0**	**72.3**
MAE [16]	ViT-L	61.3	304	80.1	83.4	65.8	59.4	72.1
MixMAE	Swin-L	35.8	235	**80.6**	**84.4**	**69.3**	**59.6**	**73.5**

Table 5. Comparison with other self-supervised approaches on classification tasks. We report the top-1 accuracy and average accuracy of all datasets. We also report the average accuracy over the 4 datasets.

scaling up the model to Swin-L. Thanks to the hierarchical design, our MixMAE consumes much fewer FLOPs for inference compared to other approaches with plain ViT. In Figure 2 (Right), our MixMAE outperforms other approaches by large margins.

Image classification. We further transfer MixMAE to other 4 classification datasets and show the results in Table 5. These datasets are challenging as the accuracies are relatively low, e.g., 57.9% top-1 accuracy on Places365 [41] for MAE-B [16]. However, our MixMAE can still outperform previous self-supervised approaches. In particular, we achieve an average +2.9% performance gain compared to MAE-B with Swin-B. Besides, our pretrained Swin-L has an average +1.4% performance gain over MAE-L while requiring only 58% FLOPs for inference.

6. Ablation Studies

In this section, we ablate the key designs of MixMAE and report the transferring results of each ablation. Unless otherwise specified, we pretrain Swin-B/W14 for 300 epochs with a masking ratio of 50%. By default, we report the top-1 accuracy on ImageNet-1K [11] and mIoU on ADE20K [42]. We also show the results on COCO [25] in Appendix.

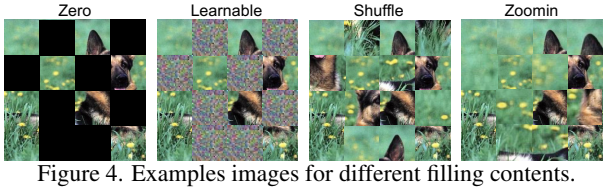

Figure 4. Examples images for different filling contents.

Type	Top-1 Acc.	mIoU
Mix	**84.6**	**49.9**
Zero	84.1	48.0
Learnable	84.1	48.9
Shuffle	82.6	43.0
Zoomin	83.5	44.9

Table 6. Filling content.

# Images (ratio)	Top-1 Acc.	mIoU
2 (0.5)	84.6	49.9
2 w/ [M] (0.75)	84.4	49.0
3 (0.67)	84.7	49.9
4 (0.75)	**84.8**	**49.9**
5 (0.8)	84.5	49.5

Table 8. Number of mixing images.

# Epochs	Top-1 Acc.	mIoU
300	84.6	49.9
600	**85.1**	50.3
900	**85.1**	**51.0**

Table 7. Pretraining epochs.

Dual	Top-1 Acc.	mIoU
✓	**84.6**	**49.9**
✗	84.0	47.3

Table 9. Dual reconstruction.

Approach	300	600	900
Ours w/o unmixing	84.4	84.4	84.4
Ours w/ mix embedding	84.4	84.6	84.8
Ours w/ masked self-attention	84.6	**85.1**	**85.1**

Table 10. Ablation on reducing the difficulty of the pretext task. We report the top-1 accuracy on ImageNet-1K for each approach with different pretraining epochs. Ours w/o unmixing denotes that we do not reduce the difficulty of the pretext task. Details of the other two approaches are described in Section 3.3.

Content to filling. While MixMAE default fills the masked tokens of one image with visible tokens from another image, we also explore more design choices. Specifically, we try to fill the masked tokens with the following contents.

- **Zero:** Filling the masked tokens with zeros. This approach causes serious mismatches between the masked tokens and the visible tokens.
- **Learnable:** Following previous works [4, 40], we fill the masked tokens with a shared learnable token. The difference between the zero approach is that learnable tokens can be adapted to visible tokens to match the distribution of the training set.
- **Shuffle:** We randomly shuffle the masked tokens, and then fill the masked locations with the shuffled tokens. We note that this approach is similar to solving jigsaw puzzles [29] with the difference that we need to fully regress the pixels.
- **Zoomin:** We zoom in the original image and randomly crop an image patch with the size of the original image. We then fill the masked tokens with tokens from the cropped image. This approach also provides masking tokens that are similar to visible ones but is harder than the shuffle approach.

We visualize the four approaches in Figure 4. We compare the performances of the four approaches in Table 6. Our default choice Mix performs best in terms of accuracy on ImageNet-1K and mIoU on ADE20K. We find the learnable approach has a similar performance on ImageNet-1K but better performance on ADE20K compared to Zero. We hypothesize that the training-finetuning inconsistency has a larger impact on tasks without a lot of labeled images. The shuffle and zoomin approaches perform much worse than other approaches. Those two strategies cause easier pretext task and have lower pretraining loss. However, the learned representation quality is lower.

Dual reconstruction. We ablate the proposed dual reconstruction in Table 9. We find that dual reconstruction greatly boosts the performance on downstream tasks. The performance gap is larger on ADE20K, where we observe +2.6 mIoU with the dual reconstruction. Note that the computation overhead of dual reconstruction is negligible as the decoder is lightweight.

Masking ratio. Our MixMAE implements different masking ratios by mixing more images at inputs. In addition, we also experiment to add $[\mathrm{MASK}]$ tokens for a higher masking ratio. We ablate the masking ratios in Table 8. We find that using a masking ratio 75% by mixing 4 images performs best. In contrast, adding $[\mathrm{MASK}]$ tokens for a 75% masking ratio has worse performance, demonstrating the effectiveness of the proposed mixing approach.

Pretraining epochs. Thanks to the dual reconstruction, our MixMAE can achieve strong performance with few pretraining epochs. We ablate the pretraining epochs in Table 7. We find that the mIoU on ADE20K can be further improved with more pretraining epochs. We achieve 51.0 mIoU with 900 epochs of pretraining. In contrast, the accuracy on ImageNet-1K does not improve after 600 epochs. It might be because the finetuning on ImageNet-1K is more adequate.

Reducing the difficulty. As stated in Section 3.3, directly performing reconstruction with the mixed input is a much more challenging optimization problem. Hence, we provide two practical approaches to reduce the difficulty. We ablate the design in Table 10. We note that all the approaches do not bring nonnegligible FLOPs or parameters. We find that the performance of the approach without unmixing is worst even when trained for more epochs. In contrast, using mix embedding alleviates the problem and improves its performance with longer pretraining. However, using masked self-attention in our final solution is much more efficient, and has better performance.

7. Discussion and Conclusion

This paper proposes Mixed and Masked AutoEncoder (MixMAE) for efficient visual representation learning. Our MixMAE uses a mixed input created by mixing two (or more) images with random masks, and applies dual reconstruction to recover the original two (or more) images from the mixed hidden representations. We further explore using Swin Transformer with a larger window size for efficient representation learning. Empirical results on 7 visual benchmarks demonstrate MixMAE can learn high-quality visual representations efficiently and has better FLOPs / performance tradeoff than previous MIM works. While this paper focuses on the vision field, we hope our work will inspire future works in other modalities, such as text and audio.

Acknowledgement This project is funded in part by National Key R&D Program of China Project 2022ZD0161100, by the Centre for Perceptual and Interactive Intelligence (CPII) Ltd under the Innovation and Technology Commission (ITC)'s InnoHK, by General Research Fund of Hong Kong RGC Project 14204021. Hongsheng Li is a PI of CPII under the InnoHK.

References

[1] Jean-Baptiste Alayrac, Joao Carreira, and Andrew Zisserman. The visual centrifuge: Model-free layered video representations. In *CVPR*, 2019.

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

[3] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv:2202.03555*, 2022.

[4] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2021.

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[6] Xiaokang Chen, Mingyu Ding, Xiaodi Wang, Ying Xin, Shentong Mo, Yunhao Wang, Shumin Han, Ping Luo, Gang Zeng, and Jingdong Wang. Context autoencoder for self-supervised representation learning. *arXiv:2202.03026*, 2022.

[7] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021.

[8] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *NeurIPS*, 34:9355–9366, 2021.

[9] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *ICLR*, 2020.

[10] Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *NeurIPS*, 34:3965–3977, 2021.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[13] Xiaoyi Dong, Jianmin Bao, Ting Zhang, Dongdong Chen, Weiming Zhang, Lu Yuan, Dong Chen, Fang Wen, and Nenghai Yu. Peco: Perceptual codebook for bert pre-training of vision transformers. *arXiv:2111.12710*, 2021.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[15] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fründ, Peter N. Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *ICCV*, 2017.

[16] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Doll'ar, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2021.

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[18] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist species classification and detection dataset. In *CVPR*, 2018.

[19] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016.

[20] Lang Huang, Shan You, Mingkai Zheng, Fei Wang, Chen Qian, and Toshihiko Yamasaki. Green hierarchical vision transformer for masked image modeling. *arXiv preprint arXiv:2205.13515*, 2022.

[21] Chunyuan Li, Jianwei Yang, Pengchuan Zhang, Mei Gao, Bin Xiao, Xiyang Dai, Lu Yuan, and Jianfeng Gao. Efficient self-supervised vision transformers for representation learning. In *ICLR*, 2022.

[22] Xiang Li, Wenhai Wang, Lingfeng Yang, and Jian Yang. Uniform masking: Enabling mae pre-training for pyramid-based vision transformers with locality. *arXiv:2205.10063*, 2022.

[23] Yanghao Li, Saining Xie, Xinlei Chen, Piotr Dollar, Kaiming He, and Ross Girshick. Benchmarking detection transfer learning with vision transformers. *arXiv:2111.11429*, 2021.

[24] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[25] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[26] Jihao Liu, Xin Huang, Guanglu Song, Hongsheng Li, and Yu Liu. Uninet: Unified architecture search with convolution, transformer, and mlp. In *ECCV*, pages 33–49. Springer, 2022.

[27] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *CVPR*, pages 12009–12019, 2022.

[28] 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. In *ICCV*, 2021.

[29] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016.

[30] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021.

[31] Jason Tyler Rolfe. Discrete variational autoencoders. In *ICLR*, 2016.

[32] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017.

[33] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *arXiv:2203.12602*, 2022.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017.

[35] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

[36] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. *arXiv:2112.09133*, 2021.

[37] Yixuan Wei, Han Hu, Zhenda Xie, Zheng Zhang, Yue Cao, Jianmin Bao, Dong Chen, and Baining Guo. Contrastive learning rivals masked image modeling in fine-tuning via feature distillation. *arXiv:2205.14141*, 2022.

[38] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018.

[39] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv:2111.11429*, 2022.

[40] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *CVPR*, 2021.

[41] Bolei Zhou, Àgata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NeurIPS*, 2014.

[42] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[43] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2021.