PROGRESSIVE COMPRESSED AUTOENCODER FOR SELF-SUPERVISED REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Masked Image Modeling (MIM) methods are driven by recovering all masked patches from visible ones. However, patches from the same image are highly correlated and it is redundant to reconstruct all the masked patches in MIM. This redundancy is neglected by existing methods and causes non-negligible overheads in computation and storage that do not necessarily benefit self-supervised learning. In this paper, we present a novel approach named Progressively Compressed AutoEncoder (PCAE) to address this problem by progressively compacting tokens and retaining the least necessary information for representation. In particular, we propose to mitigate the performance degradation caused by token reduction through exploiting the vision transformer to leak information from discarded tokens to the retained ones. Besides, we also propose the progressive discarding strategy to achieve a better trade-off between performance and efficiency. Identifying redundant tokens plays a key role in redundancy reduction. We resolve this issue using a simple yet effective criterion, *i.e.*, we identify redundant tokens according to their similarity to the mean of token sequence. Thanks to the flexible strategy, PCAE can be employed for both pre-training and downstream finetuning and, consequently, reduces the computing overhead non-trivially throughout the training pipeline. Experiments show that PCAE achieves comparable performance while at most accelerates 1.9 times throughput compared with MAE for self-supervised learning, and accelerates 15%-57% throughput while the performance drop is within 0.6% for downstream classification.

1 INTRODUCTION

Self-supervised learning has witnessed great progress in recent years and even outperforms its supervised counterpart in downstream tasks (Chen et al., 2020b; Caron et al., 2021; Grill et al., 2020). However, self-supervised methods usually take more epochs and computational overheads for considerable performance, which limits their applications. Recently, Masked Image Modeling (MIM) becomes a popular topic for its scalability as well as promising performance (Bao et al., 2022; He et al., 2022; Dong et al., 2021; Chen et al., 2022; Xie et al., 2022), especially for MAE (He et al., 2022) which significantly accelerates training via only operating on 25% visible patches in the encoder. MIM methods are driven by recovering all masked patches from visible patches, but this reconstruction task could be a huge burden for them. For example, MAE He et al. (2022) masks 75% input patches and recovers them in the decoder. As the encoder only handles 25% tokens, the overhead of the decoder (5.28G Flops) exceeds that of the encoder (4.3G Flops) during training, even though the decoder is relatively lightweight. Therefore, the efficiency bottleneck *de facto* lies on the decoder. Note that MIM methods assume that the input image is redundant, and this redundancy suppose to be consistent for the input and the reconstruction targets, yet neglected by previous methods. This observation motivates us to relax the efficiency bottleneck of MIM methods by reducing the redundancy in reconstruction targets.

In this paper, we present a novel approach named Progressive Compressed Auto-Encoder (PCAE) to reduce the redundancy in reconstruction targets. The core idea is that: instead of recovering all masked patches, we neglect redundant ones and only reconstruct representative targets. However, naively removing part of the reconstruction patches suffers serious performance degradation (please refer to Table 9 for more details) as useful information is discarded along with redundancy. Therefore, we propose to mitigate this issue by exploiting the information propagation capability of the

vision transformer to leak information from the discarded patches to the retained ones. Specifically, we discard tokens produced by the vision transformer, *i.e.*, the momentum encoder, rather than the original reconstruction targets. To further alleviate information loss, we propose the progressive discarding strategy where redundant tokens are progressively discarded at different layers of the momentum encoder (please refer to Table 4 for more details). Identifying redundant tokens lays the foundation of this discarding strategy. Matrix decomposition precisely identifies the redundant component of token sequence but exhibits prohibitive complexity in practice. As an alternative, we propose a simple yet effective criterion where we identify the redundant token according to their similarity to the mean of the token sequence. Experimental results and visualizations demonstrate that this criterion is very effective and introduces negligible overhead.

Benefiting from the proposed strategy, PCAE reduces the reconstruction targets from 147 tokens to 18 tokens, which significantly improves the efficiency. Experiments over multiple benchmarks demonstrate the effectiveness of the proposed method. Specifically, PCAE accelerates 1.9 times throughput (624 img/s vs 328 img/s) or saves nearly 2/3 memory usage (10.82G vs 28.12G) under the same batch size compared with MAE (He et al., 2022) while enjoying much faster converge speed (PCAE 300 ep 83.6 vs MAE 1600 ep 83.6) or higher performance (PCAE 800 ep 83.8 vs MAE 1600 ep 83.6). Besides, we extend the proposed strategy to inference as well as downstream fine-tuning for classification tasks, and the results show this strategy still works well. In practical deployments, it is important to provide a continuous performance-overhead curve. Thanks to the flexibility, we could provide models with different inference speeds by adjusting the number of discarded tokens. Specifically, compared with the vanilla model, we accelerate 15% to 57% throughput during inference while the drop of accuracy is within 0.6%.

In summary, we list our contributions as follows:

- We revisit the reconstruction in MIM and point out that the computational overhead is redundant, which is ignored by previous methods.
- We propose a novel self-supervised learning approach, termed PCAE to accelerate the self-supervised pretraining with negligible performance loss.
- Our proposed PCAE can be adapted to downstream fine-tuning and achieves unified acceleration among pre-training and downstream tasks.

2 Related Work

Self-supervised learning. Self-supervised learning builds the learning target from the input signal itself and expects to generalize the obtained knowledge to downstream tasks. One taxonomy for self-supervised learning is to divide it into contrastive and generative Liu et al. (2021). Contrastive learning methods Chen et al. (2020a); Wu et al. (2018); He et al. (2020); Chen et al. (2020b); Caron et al. (2020; 2021); Zbontar et al. (2021); Li et al. (2021); Grill et al. (2020) have been widely studied and achieve great progress over multiple downstream tasks including classification, object detection, and semantic segmentation Chen et al. (2020b). Generative self-supervised learning methods include auto-regressive model Van Oord et al. (2016); Van den Oord et al. (2016), flow-based model Dinh et al. (2014) and the auto-encoding model Ballard (1987). Among these methods, the denoising auto-encoding (DAE) model Vincent et al. (2008) inspires both MLM and MIM methods. The core idea of DAE is corrupting the input signal and recovering the input signal via auto-encoder which is similar to current MIM methods. Recent studies Chen et al. (2022); He et al. (2022); Wei et al. (2022) show that the MIM methods outperform previous contrastive learning methods in downstream fine-tuning tasks based on the modern vision transformer. In this paper, we mainly focus on MIM methods.

Masked image modeling. Predicting the masked part of the image is a classic self-supervised task, and has been researched by previous works Doersch et al. (2015); Pathak et al. (2016) based on the convolutional neural network. However, due to their limited performance, these methods are located in a non-mainstream position for a long time, especially after the advent of contrastive learning. Recently, the vision transformer Dosovitskiy et al. (2021) bridges the architecture gap between vision and language, and more MIM methods Bao et al. (2022); Dong et al. (2021); Xie et al. (2022); He et al. (2022); Wei et al. (2022); Chen et al. (2022) are proposed inspired by Masked Language Modeling. BEiT Bao et al. (2022), PeCo Dong et al. (2021) and CAE Chen et al. (2022) rely on the pre-trained dVAE to produce the prediction target. MAE He et al. (2022) and SimMIM

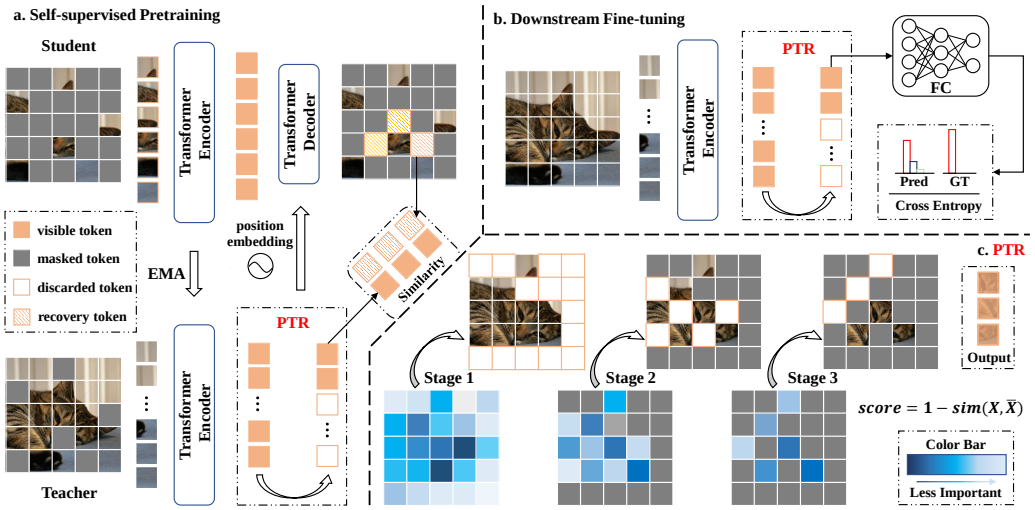


Figure 1: The overall framework of PCAE. (a) PCAE for self-supervised pre-training. PCAE consists of the student, the teacher which is the exponential moving average (EMA) model and the decoder. The student operates on visible patches while the teacher operates on masked patches. The teacher progressively discard redundant tokens and ultimately output much fewer tokens for reconstruction. The decoder reconstruct the output of the teacher according to the output of the student. (b) PCAE for downstream fine-tuning. PCAE could be easily adapted to downstream fine-tuning by transferring the behavior of the teacher to the supervised setting. (c) Progressive Token Reduction (PTR). The token reduction strategy used in the teacher or the backbone for downstream fine-tuning.

Xie et al. (2022) directly predict the raw pixel value, and MaskFeat Wei et al. (2022) predicts the HOG Dalal & Triggs (2005) feature of masked patches. Despite predicting different kinds of targets, these methods reconstruct all masked patches at the output end, which still be caught in the low information density problem.

Dynamic Vision Transformer. Under the supervised setting, several methods Rao et al. (2021); Meng et al. (2022); Yin et al. (2022); Liang et al. (2022) have been proposed to dynamically discard inattentive tokens for efficiency in vision transformers. Dynamic ViT (Rao et al., 2021), A-ViT (Yin et al., 2022) and Ada-ViT Meng et al. (2022) propose to measure token importance through extra learnable parameters and only accelerates the inference. Besides, these methods require to initialize from pre-trained models, thereby inaccessible for self-supervised learning. E-ViT does not need extra parameters and (Liang et al., 2022) measures token importance via the attention to the class token, but it relies on ground truth to supervise the class token which is absent in self-supervised learning. In summary, although dynamic vision transformers have been explored in supervised learning, they are inaccessible for self-supervised learning and the strategy that identifies the importance of tokens without prior knowledge is absent so far.

3 METHOD

This section is organized as follows: in section 3.1, we present the overall framework and objective of PCAE. In section 3.2, we elaborate on how we build the compact reconstruction targets including identifying redundant tokens and the progressive discarding strategy. In section 3.3, we demonstrate the effectiveness of PCAE through illustrating the diversity of token sequence at different layers and visualizing the remaining tokens at each stage. In section 3.4, we extend PCAE to downstream classification task so that the proposed strategy could improve efficiency across pre-training, fine-tuning and final inference.

3.1 OVERVIEW

The framework of PCAE is illustrated in Figure 1 (a), and the overall objective of PCAE is:

$$\max_{\theta,\phi,\omega} \mathop{\mathbb{E}}_{x\sim\mathcal{X}} \mathcal{M}(\bar{f}((1-M)\odot x);k), g_\phi(h(f_\theta(M\odot x),\omega+p)) \tag{1}$$

where $\odot$ is the element-wise product; M is the random mask and $M \odot x$ denotes unmasked patches while $(1 - M) \odot x$ denotes masked patches; f_θ is the encoder to be pre-trained while $\bar{f}$ is the exponential moving average model of f_θ; k is the number of tokens we expect to keep. ω is a learnable mask token and p is the positional embedding of tokens requested to reconstruct; $h(\cdot, \cdot)$ is the concatenation operation; g_ϕ is the decoder network; $\mathcal{M}(\cdot, \cdot)$ is the similarity measurement, and we instantiate it with cosine similarity here.

The EMA model $\bar{f}$ operates on masked patches and progressively discards redundant tokens from the token sequence, thereby ultimately output much fewer tokens. On the other end, the encoder process unmasked patches and output $f_\theta(M \odot x)$. Then, the decoder g_ϕ concatenates the output of the encoder and the position embedded mask tokens to reconstruct $\bar{f}((1 - M) \odot x)$. Note that, as a reminiscent method, MAE (He et al., 2022) expects to reconstruct all patches, so the number of mask tokens is 147. In contrast, PCAE only reconstructs 18 tokens, and the computational overhead on the decoder could be largely reduced.

3.2 Progressively token reduction

In this section, we elaborate on the $\bar{f}$ where we progressively discard redundant tokens. We discuss the progressive discarding strategy and redundant token identification in the following.

Progressive discarding strategy. As vision transformers are stacked by self-attention blocks, it is important to decide which layer to discard tokens. Intuitively, for a given keep ratio, discarding all tokens at once suffers serious information loss, and progressive discarding them at different layers could largely alleviate it through exploiting the information leakage brought by the self-attention mechanism. Formally, we decompose $\bar{f}$ into $\bar{f}_1 \circ \bar{f}_2 \circ \cdots \circ \bar{f}_L$ where $\bar{f}_i$ indicates the ith layer in $\bar{f}$. Then, for the output of ith layer $x_i \in \mathbb{R}^{N \times D}$ we have:

$$x_i = \begin{cases} D(\bar{f}_i(x_{i-1}); k_i), & \text{if } i \in I \\ \bar{f}_i(x_{i-1}), & \text{if } i \notin I \end{cases} \tag{2}$$

where I is the set of layer index we expect to discard tokens; $D(\cdot; \cdot)$ is the redundant token identification function and we will discuss it in later; k_i is the number of tokens we expect to keep in i^{th} layer. In practice, we apply the consistent drop ratio τ for all stages, so $k_i = \lfloor (1 - \tau) * N \rfloor$. Note that as we perform the token reduction in the momentum encoder, the gradient backpropagation is not taken into consideration. Thus, once a token is discarded, it will never be used in later layers, which accelerates the inference of the momentum encoder.

Redundant token identification. Matrix decomposition precisely identifies the redundant component in token sequence but exhibits prohibitive complexity in practice (slow down training five times in our implementation). As an alternative, we turn to identify redundant tokens according to their similarity to the average token. We are motivated by two reasons. First, removing tokens most similar to the average token has the smallest influence on the average of the new token sequence. Specifically, considering the average of the original token sequence $\bar{a}$, the average of remained tokens a_r and the average of discarded tokens a_d, it is obvious that $\|\bar{a} - a_r\| \propto \|\bar{a} - a_d\|$. Therefore, this strategy has minimal impact on the overall distribution of tokens. Second, according to the Discrete Fourier Transform (DFT), the average token corresponds to the low-frequency component of the image which usually presents as the background, and it is reasonable to remove them as they contribute little to visual perception. Formally, for a token sequence $x \in \mathbb{R}^{N \times D}$, the redundant token identification can be given as:

$$\begin{aligned} S &= rank(\mathcal{M}(x, \sum_N x^j / N)) \in \mathbb{R}^N, \\ D(x; k) &= \mathbb{1}[S > k] \in \mathbb{R}^N \end{aligned} \tag{3}$$

where $\mathcal{M}$ is the similarity measurement and instantiated with cosine similarity; $rank(\cdot)$ denotes the rank function in ascending order; S is the rank of tokens; k is the number of tokens we expect to keep. Eq 3 shows that we determine whether a token is redundant according to its relative similarity to the average token. Tokens that are more similar to the average token than other k tokens are regarded as redundant tokens, and vice versa.

The output of $\bar{f}$ could be given by combining Eq 3 and Eq 2.

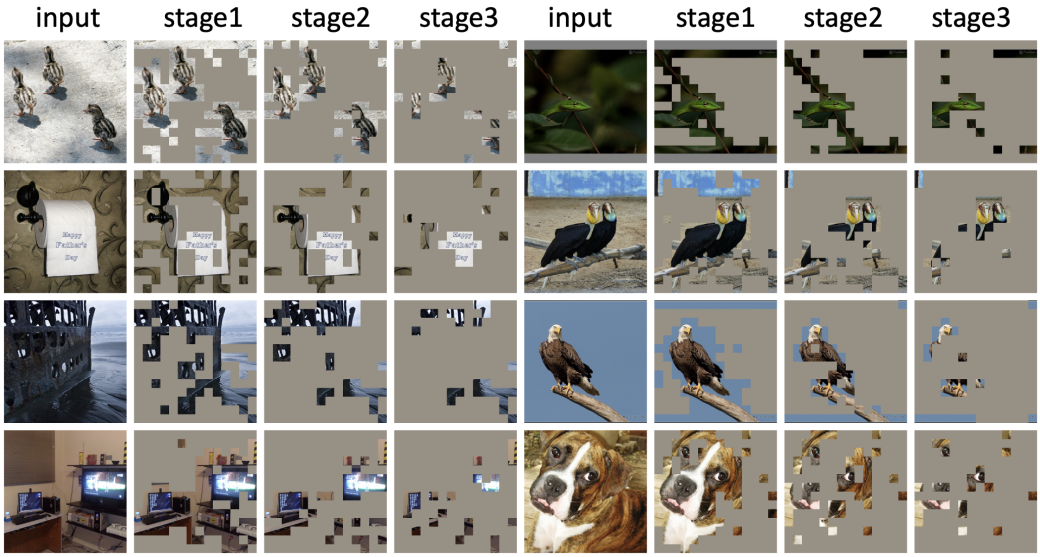


Figure 2: The visualization of the proposed state-wise discarding method. Images are arranged from left to right, top to bottom, and images 1-8 refer to the original image. We forward images with PCAE pre-trained model and record the position of the remaining tokens. Then, we mask patches corresponding to the discarded tokens at each stage. These visualizations show that PCAE prefer to keep the high-frequency component of images.

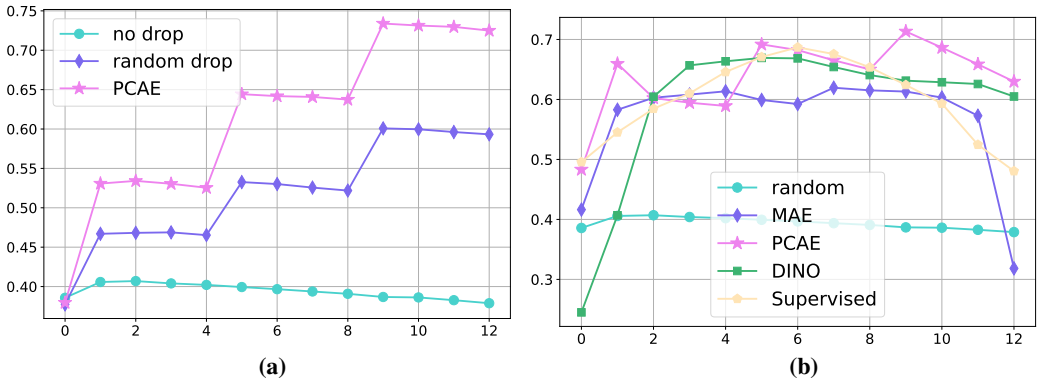


Figure 3: We measure the diversity of token sequences during the forward process. The vertical axis indicates the value of the nuclear norm, and the horizontal axis indicates the index of the layer (0 indicates the input token sequences). (a) We run the randomly initialized ViT-Base network on ImageNet validation set with three different settings to show their differences in token diversity. (b) We compare the curve which describes the relationship between the layer number and the nuclear norm between models initialized by different methods.

3.3 Analyze

3.3.1 Diversity enhancement by PCAE

We run randomly initialized ViT-B networks on ImageNet validation set under different settings and measure the average diversity at each layer (0 indicates the input token sequences) with the nuclear norm (more details please refer to the supplementary) over the whole ImageNet validation set. As illustrated in Figure 3a, the proposed strategy effectively enhances the diversity of token sequence compared with cases where we do not discard tokens or randomly discard tokens.

In addition, we also apply this setting to pre-trained checkpoints to reveal the differences between pre-training methods. As illustrated in Figure 3b, we find that all pre-training methods no matter supervised or unsupervised enhance the diversity of token sequence at intermediate layers. We hypothesize that meaningful pre-training makes the network learn to focus on important things and

Table 1: Image classification results on the ILSVRC-2012 ImageNet dataset with top 1 accuracy. "Epochs" refers to the number of pre-training epochs. MoCo v3 and DINO adopt multi-crop augmentation for pre-training. MoCo v3: 2 global crops of 224 × 224. DINO: 2 global crops of 224 × 224 and 10 local crops of 96 × 96.† indicates this method needs pre-trained dVAE. The GPU day: All methods are run under 1024 batch size for fair comparison.

Method	Epochs	Crops	Gpu day	Accuracy
Train from Scratch	300	-	-	81.8
MoCo v3	300	2	-	83.2
DINO	400	12	122.5	83.3
BEiT†	300	1	-	83.0
MAE	300	1	15.4	82.9
CAE†	300	1	-	83.6
MAE	1600	1	82.1	83.6
SimMIM	800	1	54.1	**83.8**
PCAE (ours)	300	1	10.1	83.6
PCAE (ours)	800	1	26.9	**83.8**

neglect redundant components in input signals. Another interesting finding is that MAE suffers low diversity at both the input end (index 0) and the output end (index 12). We hypothesize that it is caused by the symmetry of its input and output. In contrast, PCAE overcome the low diversity problem at its output end.

3.3.2 VISUALIZATION OF PCAE

We visualize patches corresponding to the remaining tokens after each stage, and the results are presented in Figure 2. We find that this visualization is exactly in accord with the frequency perspective we discussed above. Taking the fifth image in the third row as an example, PCAE filters most of the sky background and eventually keep the detail-rich patches such as the beak and claws. It also holds for other images, such as the three chicks in the first image, the snake's head in the second image, and the text in the third image. All these details are well preserved.

We also find that PCAE not only focuses on the salient object, but retains a variety of objects within the image. As illustrated in Figure 2, tokens kept by PCAE are diverse and distributed throughout the image, which preserves the diversity within the image. We attribute it to the stage-wise discard strategy. Take the sixth image as an example, the first two stages discard most patches belonging to the sky. Therefore, in the third stage, patches belonging to the sky are no longer redundant, thereby being preserved eventually. The stage-wise strategy helps preserve the diversity within images which is important for the MIM-based method.

3.4 DOWNSTREAM FINE-TUNING

As the proposed strategy is easy to implement and does not require any extra parameters, it can be seamlessly incorporated into inference as well as downstream fine-tuning. Specifically, we transfer the implementation of $\bar{f}$ that we presented in section 3.2 to downstream tasks and discard tokens both in fine-tuning and inference stages.

We observe two important differences between pre-training acceleration and downstream acceleration. First, downstream tasks are more sensitive to token reduction. The number of remaining tokens can be as less as 18 in the pre-training stage, but we observe serious performance degradation in downstream tasks in the such extreme case. Therefore, we at most discard 40% tokens each stage and discard tokens after 0th, 7th, 10th layer. Second, the discarded tokens could be used to improve performance. The discarded tokens are not used in the pre-training stage, but we find they are useful for downstream tasks. Therefore, we append the average of discarded tokens to the retained tokens in downstream tasks. However, we remove these appended tokens before final global pooling. Besides, we apply a warmup strategy on the drop ratio parameter in fine-tuning following Liang et al. (2022). Specifically, we do not discard any token in the previous 10 epochs and then gradually increase the drop ratio to the target value in the following 25 epochs. Then, we keep the target drop ratio until fine-tuning ends.

Table 2: Object detection and instance segmentation on COCO. All the results are based on the same implementations. Mask R-CNN is adopted and trained with the 1× schedule. "Epochs" refers to the number of pre-training epochs on ImageNet-1K.

Methods	Epochs	Object Detection			Instance Segmentation		
		AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
Supervised methods:							
DeiT	300	46.9	68.9	51.0	41.5	65.5	44.4
Contrastive learning methods:							
MoCo v3	300	45.5	67.1	49.4	40.5	63.7	43.4
DINO	400	**46.8**	68.6	50.9	**41.5**	65.3	44.5
Masked Image Modeling methods:							
BEiT	300	39.5	60.6	43.0	35.9	57.7	38.5
BEiT	800	42.1	63.3	46.0	37.8	60.1	40.6
MAE	300	45.4	66.4	49.6	40.6	63.4	43.7
MAE	1600	48.4	69.4	53.1	42.6	66.1	45.9
PCAE (ours)	300	47.6	68.3	52.4	42.0	65.3	45.6
PCAE (ours)	800	**48.7**	69.4	53.5	**42.8**	66.1	46.2

4 EXPERIMENTS

In this section, we first compare PCAE with other MLM methods on multiple benchmarks including classification, object detection, and segmentation. Then, we extend PCAE to downstream tasks for acceleration. After that, we ablate on the hyper-parameters of PCAE. We follow MAE He et al. (2022) for the settings in data augmentation and optimization. We apply 75% mask ratio on the input image across all experiments, and discard 50% tokens after 0th, 4th, and 8th layer respectively in default. For more details please refer to the supplementary material.

4.1 COMPARISON WITH OTHER METHODS

Fine-tuning on ImageNet-1k. We follow the standard MLM evaluation protocol that fine-tunes the pre-trained model on the ILSVRC-2012 ImageNet for 100 epochs. We follow the fine-tuning schedule and hyper-parameters in MAE He et al. (2022) out of fair comparison. Under this setting, we compare PCAE with its main counterpart MAE and other self-supervised methods.

Referring to Table 1, PCAE largely outperforms MAE with the same number of epochs but saves around 35% computational cost. For the same performance, PCAE only takes 1/8 overhead compared with MAE. Although BEiT and CAE can benefit from DALL-E Ramesh et al. (2021) which is pre-trained on 250M images, PCAE still reach a better or comparable fine-tuning performance. Contrastive learning methods rely on complex data augmentation which badly increases the computational cost, especially for DINO which relies on the multi-crop operation. As presented in Table 1, PCAE outperforms DINO in the fine-tuning benchmark at a much lower cost (around 1/12).

Object detection and segmentation. Besides the classification task, we evaluate PCAE on object detection and segmentation to demonstrate its potential in dense prediction tasks. In this setting, we fine-tune Mask R-CNN He et al. (2017) with the pre-trained parameters in an end-to-end manner on COCO Lin et al. (2014). The ViT backbone is adapted for use with FPN Lin et al. (2017). We report the box AP for object detection and the mask AP for instance segmentation in Table 4.1. The results show that PCAE consistently outperforms other methods both in object detection and instance segmentation.

Throughput comparison. As the training may be influenced by many factors, we show the throughput comparison between PCAE and MAE in Table 4.2. The results show that PCAE is still much more efficient than MAE even though MAE is already a very efficient method. Specifically, PCAE at most accelerate 1.9 times throughput or save 2/3 memory usage compared with MAE. We also adapt PCAE to SimMIM, termed PCAE-SimMIM, which accelerates the throughput 2.37 times without compromising performance. For more details please refer to the supplementary.

Table 3: The throughput comparison with MAE. We run PCAE and MAE on the same machine equipped with V100 GPU under different batch size. The acceleration is calculated under the same batch size if possible and compared with the maximum throughput otherwise.

Batch size	PCAE			MAE		
	Memory	Throughput	Accelerate	Memory	Throughput	Accelerate
64	4.71G	412.6	1.56×	8.82G	265.1	1.0×
128	6.75G	498.7	1.65×	16.28G	302.6	1.0×
256	10.82G	572.9	1.74×	28.12G	328.4	1.0×
512	18.98G	606.2	1.85×	/	/	/
768	27.11G	624.1	1.90×	/	/	/

Table 4: We fix the drop ratio and ablate on different drop stage (include 0). "Exp id" indicates the experimental identity.

Exp id	drop tokens	drop stage	drop ratio (%)	total (%)	GPU Day	Accuracy (%)
Exp id = 1	×	0	0	0	5.7	**82.3**
Exp id = 2	✓	1	50	50	3.9	81.0
Exp id = 3	✓	2	50	75	3.6	81.8
Exp id = 4	✓	3	50	87.5	3.3	**82.3**
Exp id = 5	✓	4	50	93.75	3.2	81.3

4.2 Downstream acceleration

Model acceleration is critical for the practical deployment of vision transformers. Due to the simplicity and flexibility of the proposed progressive token reduction strategy, it can be easily incorporated into the supervised setting and benefit the inference speed. With the pre-trained model, we fine-tune it on ImageNet under different drop ratios for 100 epochs and report the Top-1 accuracy. As presented in Table 6, PCAE balance the acceleration and performance well. Specifically, PCAE accelerates 57% throughput while the performance drop is within 0.6%. Even for the extreme case where only 42 tokens are kept out of 196 tokens, the performance slightly drops 1.2% but the throughput improves 88%.

These results further demonstrate the effectiveness of PCAE as it manages to accelerate scenarios including self-supervised pre-training, supervised fine-tuning, and inference.

4.3 Ablation Study

We apply the same setting for all ablation studies unless other specific. Specifically, we pre-train PCAE on ImageNet for 100 epochs with the batch size of 1024 and fine-tune pre-trained parameters on ImageNet for 100 epochs with the default schedule of MAE for a fair comparison.

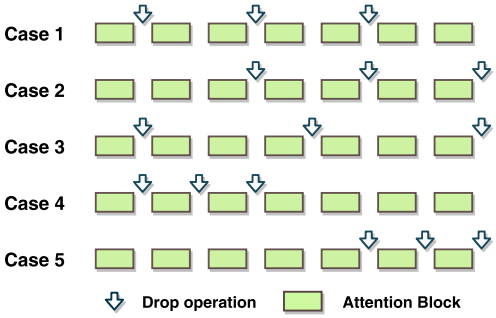
Initially, we highlight critical hyper-parameters of PCAE for clarity. Referring to Eq 2, our discarding strategy is determined by I which decides where tokens are discarded and the drop ratio hyper-parameter. As the value of I is too huge, we decompose it into two parameters, i.e., drop stage and drop case. The drop stage indicates how many times we should discard tokens, i.e., $card(I)$. Drop case indicates where we prefer to discard tokens, i.e., shallower layers, deeper layers, e.t.c. (refer to Table 5 for more details). In summary, drop stage, drop case, and drop ratio are three critical hyper-parameters for PCAE, and their effects are elaborated in the following. The core operation in PCAE is discarding tokens during forwarding, so we elaborate this operation by answering two questions: how many tokens do we need to discard and where should we discard tokens with different settings of drop stage, drop ratio and drop case.

How many tokens should we discard? We first fix the drop ratio and drop case, and ablate on the drop stage. As presented in Table 4, the results show that removing redundant tokens from the prediction target significantly improves efficiency (exp 1 and others). Among these experiments, we find that the reconstruction targets also prefer to high drop ratio, which is similar to the input end, and discarding 87.5% tokens (exp 4) achieves the best balance between performance and efficiency.

We further control the total drop ratio equal to 87.5% and ablate on the drop stage, i.e., we need to drop tokens in few times with a large drop ratio or more times with a smaller drop ratio. A

Table 5: We fix the drop stage and drop ratio, and ablate on different drop case (include the comparison to randomly drop *abbr* RD). The figure layout in right shows the drop position.

Exp id	RD	drop case	GPUd	Acc
Exp id = 1	✓	1	3.3	81.0
Exp id = 2	×	1	3.3	**82.3**
Exp id = 3	×	2	3.7	82.0
Exp id = 4	×	3	3.4	82.0
Exp id = 5	×	4	3.1	81.5
Exp id = 6	×	5	4.8	80.7



large drop ratio improves efficiency but may suffer performance degradation, and vice versa. The empirical results in Table 7 show that dropping 3 times is the best choice to balance them.

Then, we fix the drop stage and grid search for the best total drop ratio from 75% to 90%. As presented in Table 8, totally discarding 80% tokens reaches the best performance. However, compared with experiment 4, we find that discarding 50% tokens three times is more simple and efficient and only slightly degrades the performance. Thus, we apply the configuration of exp 4 in default.

Table 6: The performance of models under different drop ratio.

Drop ratio	Throughput	MACs	Acc
0	286	17.57	83.8
0.1	330 (↑15%)	15.34 (↓13%)	83.7 (↓0.1)
0.2	385 (↑35%)	13.19 (↓25%)	83.5 (↓0.3)
0.3	449 (↑57%)	11.29 (↓36%)	83.2 (↓0.6)
0.4	538 (↑88%)	9.53 (↓46%)	82.6 (↓1.2)

Where should we discard tokens? We ablate on the drop case parameter here. We propose five different cases to discard tokens and their details are illustrated at the right of Table 5. The results in Table 5 show that evenly allocating the discarding operation among blocks performs better than continuously discarding tokens (compare exp 2, 3, 4 with exp 5, 6). The drop case 1 performs best compared with case 2 and case 3 (exp 2, 3, 4), which indicates that token reduction supposes to be avoided at the last layer.

Table 7: We fix the total drop ratio and ablate on drop stage.

Exp id	drop stage	drop ratio	GPUd	Acc
Exp id = 1	1	87.5	3.0	81.5
Exp id = 2	3	50	3.3	**82.3**
Exp id = 3	6	29	3.5	82.0
Exp id = 4	12	16	3.6	81.5

Table 8: We fix the drop stage and ablate on the totally drop ratio.

Exp id	drop ratio	total	GPUd	Acc
Exp id = 1	63	75	3.6	81.8
Exp id = 2	58.5	80	3.5	**82.4**
Exp id = 3	53	85	3.4	81.9
Exp id = 4	50	87.5	3.3	82.3
Exp id = 5	46.4	90	3.3	81.7

We also compare with the random drop in Table 5, and the result shows that the performance of randomly discarding tokens far falls behind that of our strategy (exp 1 and exp 2). Besides, we provide more performance of random drop under different configurations in the supplementary, and results show the proposed strategy outperforms it in all cases.

5 Conclusion

In this paper, we present PCAE which addresses the problem of the redundant reconstruction target in MIM methods. The results show that PCAE outperforms MAE over multiple benchmarks in cases where PCAE even only reconstructs 18 tokens. The lightweight reconstruction task accelerates training (at most 1.9 times throughput acceleration compared with MAE) without compromising performance. We visualize the remained tokens and find that PCAE manages to retain semantic-abundant components in images and mainly discards tokens belonging to the background. PCAE exhibits elegant symmetry at the input and output end as both them are compressed. What beyond the proposed method is the insight that MIM-based methods benefit from more compact prediction targets. We wish it can inspire future works.

REFERENCES

Dana H Ballard. Modular learning in neural networks. In *Aaai*, volume 647, pp. 279–284, 1987.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEit: BERT pre-training of image transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=p-BhZSz59o4.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 9912–9924, 2020.

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 - International Conference on Computer Vision*, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML 2020: 37th International Conference on Machine Learning*, volume 1, pp. 1597–1607, 2020a.

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 preprint arXiv:2202.03026*, 2022.

Xinlei Chen, Haoqi Fan, Ross B. Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020b.

Shuhao Cui, Shuhui Wang, Junbao Zhuo, Liang Li, Qingming Huang, and Qi Tian. Towards discriminability and diversity: Batch nuclear-norm maximization under label insufficient situations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3941–3950, 2020.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pp. 886–893. Ieee, 2005.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1422–1430, 2015.

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 preprint arXiv:2111.12710*, 2021.

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: The Ninth International Conference on Learning Representations*, 2021.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In *International Conference on Machine Learning*, pp. 3015–3024. PMLR, 2021.

Maryam Fazel. *Matrix rank minimization with applications*. PhD thesis, PhD thesis, Stanford University, 2002.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284, 2020.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Junnan Li, Pan Zhou, Caiming Xiong, and Steven Hoi. Prototypical contrastive learning of unsupervised representations. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=KmykpuSrjcq`.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=BjyvwnXXVn_`.

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 David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pp. 740–755. Springer, 2014. doi: 10.1007/978-3-319-10602-1_48.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12309–12318, 2022.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2536–2544, 2016.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pp. 8821–8831. PMLR, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International conference on machine learning*, pp. 1747–1756. PMLR, 2016.

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

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.

Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14668–14678, 2022.

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3733–3742, 2018.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9653–9663, 2022.

Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10809–10818, 2022.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pp. 12310–12320. PMLR, 2021.

APPENDIX

A DIVERSITY MEASUREMENT

Initially, we show how we measure the diversity. We start by looking at the input token sequence from a patchify image. Denote the number of token as N, and the dimension as D, the token sequence can be written as a matrix $A \in \mathbb{R}^{N \times D}$. Following the common practice in self-supervised learning Wang & Isola (2020); Ermolov et al. (2021), we analyze the token representations on the unit sphere, i.e., we normalize the length of each token to one. As the convex envelope of matrix rank, the nuclear norm is widely used to measure to diversity of matrix rows Cui et al. (2020). Specifically, according to the theorem in Fazel (2002), $\|A\|_*$ is the convex envelope to $rank(A)$ when $\|A\|_F \leq 1$. Since $\|A\|_F = \sqrt{N}$ (note that each row vector in A is unit vector), the convex envelope of $rank(A)$ turn to $\|A\|_*/\sqrt{N}$. According to the property of the nuclear norm, $\|A\|_*/\sqrt{N}$ is bounded by $\sqrt{k}$ where $k = min(N, D)$. In practical implementation, N is the length of token sequence and usually be 196, and D is the embedding dimension and usually be 768 or 512. Thus, $min(N, D) = N$ and $\|A\|_*/\sqrt{N} \leq \sqrt{N}$. Since the length of token sequences is various during forward process and the upper bound of $\|A\|_*/\sqrt{N}$ depends on N, we instead apply $\|A\|_*/N$ which has a clear upper bound and get rid of the influence from N to measure the diversity in token sequences.

B IMPLEMENTATION DETAILS

Architecture. We study the ViT base architecture (12 transformer blocks with dimension 768) and follow the implementation of Dosovitskiy et al. (2021). The teacher and the student branch share the same architecture. We follow the implementation of He et al. (2022) for the decoder part. Specifically, we apply an FC (fully-connected layer) as the dimension adapter between the encoder and decoder and two transformer blocks to decode the masked tokens with the output of the student. At last, we apply another FC to convert the decoder dimension into the encoder dimension for reconstruction. What is different from MAE is that we only reconstruct tokens output by the teacher.

Hyper-parameters. We follow the setting of hyper-parameters of MAE (He et al., 2022). For the EMA operation, we apply the fixed momentum equal to 0.9999.

C THE NAIVE APPROACH OF DISCARDING PART OF THE TARGETS.

We present the naive approach of discarding part of the reconstruction targets of MAE here. We discard patches with the proposed strategy and only reconstruct remanent patches with MAE. We apply two settings for the base MAE. In the first setting, we simply mask gradients from discarded patches while we only decode remanent tokens in the decoder in the second setting. We observe that both settings suffer serious performance degradation as presented in Table 9. This demonstrates what we stated in the main text that simply discarding part of the reconstruction target damages useful information as well as eliminates redundancy.

Table 9:

Setting	Acc
Base	**82.1**
Setting id = 1	79.6
Setting id = 2	79.9

D ADAPT TO SIMMIM

For SimMIM, which forwards visible tokens together with the mask tokens at the encoder, we can regard the encoder in SimMIM as the decoder in MAE, and thereby speed up it via discarding most of the mask tokens. Specifically, the length of the token sequence is 196 (78 visible tokens and 118 mask tokens) in vanilla SimMIM, and we can speed up it by building a much more compact target for it. Then, the length of the token sequence can be reduced to 96 (78 visible and 18 mask tokens), thereby improving the efficiency. We adapt PCAE to SimMIM with minimum modification, and we find that PCAE-SimMIM achieves comparable performance (PCAE-SimMIM 82.3 vs SimMIM

Table 10: The throughput comparison between PCAE-SimMIM and SimMIM. We adapt PCAE to SimMIM and compare its throughput with the original SimMIM on the same machine equipped with V100 GPU under different batch size. The acceleration is calculated under the same batch size if possible and compared with the maximum throughput otherwise.

Batch size	PCAE-SimMIM			SimMIM		
	Memory	Throughput	Accelerate	Memory	Throughput	Accelerate
64	5.63G	310.2	1.71×	11.32G	181.1	1.0×
128	8.43G	383.4	1.87×	20.82G	204.9	1.0×
164	9.99G	417.7	2.91×	26.16	143.5	1.0×
256	14.08G	432.8	2.11×	/	/	/
512	25.37G	485.4	2.37×	/	/	/

82.1 100 epoch) to SimMIM with 55% training hours (PCAE-SimMIM 11.1h vs SimMIM 20.3h on 8 V-100 machine). Besides, we provide the throughput comparison between PCAE-SimMIM and SimMIM below, which shows PCAE still works well when transferred to other methods.

E MORE COMPARISONS TO RANDOM DROP.

We supplement more experiments here to show our advantage over randomly drop (RD). For a fair comparison, we fix the total drop ratio equal to 87.5% and search for the best drop strategy for randomly discarding by the drop case and drop stage. Specifically, we fix the drop stage and run the randomly drop strategy with different drop case (we select drop cases with better results by referring to Table 5). The results in Table 11 show that the randomly drop strategy much worse than PCAE. Besides, we also fix the drop case and search on the drop stage, but the results still show that PCAE outperform the randomly drop strategy by a large margin.

Table 11: The comparison between the proposed redundant token identification strategy and randomly drop (RD)

Method	drop stage	drop ratio	drop case	FT acc
PCAE	3	0.5	1	**82.3**
RD	3	0.5	1	81.0
RD	3	0.5	2	81.1
RD	3	0.5	3	80.3
RD	6	0.29	1	80.4
RD	12	0.16	1	81.0