RIFormer: Keep Your Vision Backbone Effective But Removing Token Mixer

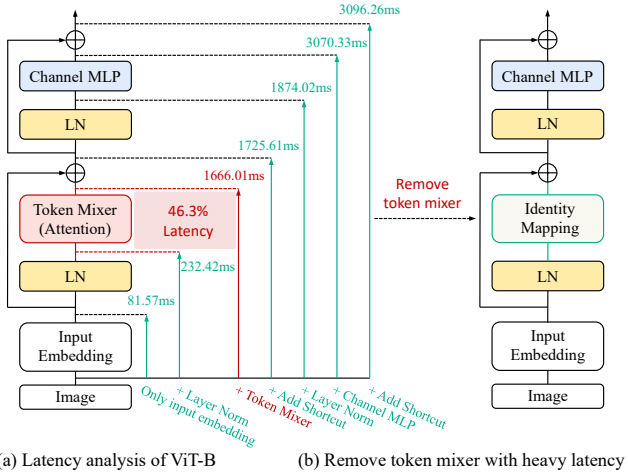
Jiahao Wang[1,2] Songyang Zhang[1*] Yong Liu[3] Taiqiang Wu[3] Yujiu Yang[3]
Xihui Liu[2] Kai Chen[1*] Ping Luo[2] Dahua Lin[1]
[1]Shanghai AI Laboratory [2]The University of HongKong
[3]Tsinghua Shenzhen International Graduate School
wang-jh19@tsinghua.org.cn {zhangsongyang,chenkai,lindahua}@pjlab.org.cn
yang.yujiu@sz.tsinghua.edu.cn xihuiliu@eee.hku.hk pluo@cs.hku.hk

Abstract

This paper studies how to keep a vision backbone effective while removing token mixers in its basic building blocks. Token mixers, as self-attention for vision transformers (ViTs), are intended to perform information communication between different spatial tokens but suffer from considerable computational cost and latency. However, directly removing them will lead to an incomplete model structure prior, and thus brings a significant accuracy drop. To this end, we first develop an RepIdentityFormer base on the reparameterizing idea, to study the token mixer free model architecture. And we then explore the improved learning paradigm to break the limitation of simple token mixer free backbone, and summarize the empirical practice into 5 guidelines. Equipped with the proposed optimization strategy, we are able to build an extremely simple vision backbone with encouraging performance, while enjoying the high efficiency during inference. Extensive experiments and ablative analysis also demonstrate that the inductive bias of network architecture, can be incorporated into simple network structure with appropriate optimization strategy. We hope this work can serve as a starting point for the exploration of optimization-driven efficient network design.

1. Introduction

The monumental advance in computer vision in the past few years was partly brought by the revolution of vision backbones, including *convolutional neural networks (ConvNets)* [13, 17, 25, 30] and *vision transformers (ViTs)* [14, 38]. Both of them have particular modules in their basic building blocks that aggregate information between different spatial locations, which are called *token mixer* [46], such as self-attention for ViTs. Although the effectiveness of token mixer has been demonstrated on many vision tasks [5, 6, 24, 45], its computational complexity typically takes up a significant portion of the network. In practice, heavy token mixers make the vision backbone limited especially on the edge-side devices due to the issue of speed and computation cost.



(a) Latency analysis of ViT-B (b) Remove token mixer with heavy latency

Figure 1. Latency analysis of different components in ViT-Base [14]. (a) For token mixer (self-attention), the latency occupies about 46.3% of the backbone. (b) Our motivation was to remove the token mixer while striving to keep the performance.

There have been several attempts in the literature to investigate efficient token mixers for slimming vision backbones [29, 31, 46]. Although those works have already achieve competitive performance with light-weight design, they do retain the token mixers, which brings non-negligible increase in latency, as illustrated in Fig. 1. The recent work [47] finds that removing the token mixer is possible but leads to performance degeneration. Those explorations in efficient token mixers inspire us to think that *can we keep the vision backbone effective but removing the token mixer?* The resulting token mixer free vision backbone is expected to be efficient and effective for the realistic application.

* Corresponding author.

In this work, we first review the current model architectures and learning paradigms. Most of the previous works concentrate on the improvement of the architecture while adopting the conventional supervised learning to optimize the model from scratch. Differently, we propose to adopt the simplified model architecture, and explore the learning paradigm design to fully exploit the potential of the simple model. We aim to simultaneously maintain the efficiency and efficacy of token mixer free vision backbone (namely IdentityFormer, in Fig. 1-(b)). To this end, we investigate the simple and yet effective learning strategy, *knowledge distillation (KD)* [18] thoroughly in the following sections.

Our main idea is distilling the knowledge from powerful teacher model (with token mixer) to the student model (token mixer free). We instantiate the re-parameterizing idea to enlarge the modeling capacity of student network but retain its efficiency, as shown in Fig. 2. Specifically, the simple affine transformation is introduced into student model, to replace the token mixer for training. The parameters of affine transformation can be merged into LayerNorm [2] during inference, which makes the student token mixer free finally.

We empirically summarize the our learning strategy as the following guidelines, hope to shed light on how to learn the extremely simple model. Concretely, **1**) soft distillation without using ground-truth labels is more effective; **2**) using affine transformation without distillation is difficult to tailor the performance degeneration; **3**) the proposed block-wise knowledge distillation, called *module imitation*, helps leveraging the modeling capacity of affine operator; **4**) teacher with large receptive field is beneficial to improve receptive field limited student; **5**) loading the pre-trained weight of teacher model (except the token mixer) into student improve the convergence and performance.

Based on the above guidelines, we finally obtain a token mixer free vision model with competitive performance enjoying the high efficiency, dubbed as *RepIdentityFormer (RIFormer)*. RIFormer shares nearly the same macro and micro design as MetaFormer [46], but safely removing all token mixers. The quantitative results show that our networks outperform many prevailing backbones with faster inference speed on ImageNet-1K [8]. And the ablative analyses on the feature distribution and *Effective Receptive Fields (ERFs)* also demonstrate that the inductive bias brought by an explicit token mixer, can be implicitly incorporated into the simple network structure with appropriate optimization strategies. In summary, the main contributions of our work are as the following:

- We propose to explore the vision backbone by developing advanced learning paradigm for *simple model architecture*, to satisfy the demand of realistic application.
- We instantiate the re-parameterizing idea to build a token mixer free vision model, RIFormer, which owns the improved modeling capacity for the inductive bias while enjoying the efficiency during inference.
- Our proposed practical guidelines of distillation strategy has been demonstrated effective in keeping the vision backbone competitive but removing the token mixer.

2. Related Work

2.1. Vision Transformer Acceleration

Vision transformer is a inference slow, energy intensive backbone due to its quadratic computational cost of the self-attention, and therefore unfriendly to deploy on resource-limited edge devices, calling for compression techniques. Various technology route are designed for vision transformer slimming, such as distilling an efficient transformer with fewer depths and embedding dimensions [16,38,39,44, 51], pruning or merging unimportant tokens [3, 21, 28, 29], applying energy efficient operations [23, 33], or designing efficient attention alternatives [4,24,31], *etc*. Different from these lines, our work propose a novel angle of totally removing the complicated token mixer in a backbone while keep satisfactory performance.

2.2. Structual Re-parameterization

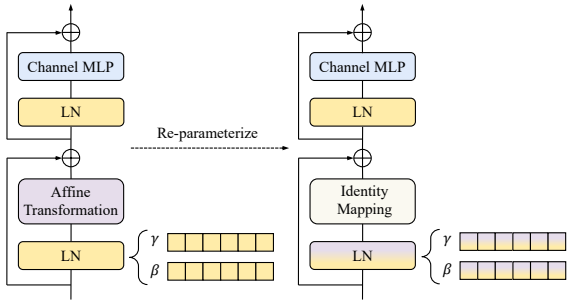
Structual re-parameterization [12, 13, 49] is a training technique which decouples the training-time and inference-time architectures. For example, RepVGG [13] is a plain VGG-style architecture with attractive performance and speed during inference, and a powerful architecture with manually added 1×1 branch and identity mapping branch during training. Similarly, such technique can be further extended to super large kernel ConvNets [12], MLP-like models [9], network pruning [11] and special optimizer design [10]. In this paper, we follow the technique to introduce parameters and equivalently absorb them into LN layer after training. The extra weights after proper optimization can help the model learn useful representations.

3. Preliminary and Motivation

In this section, we first briefly recap the concept of token mixer. Then, we revisit their inevitable side effects on inference speed through an empirical latency analysis, and thus introduce the motivation of our paper.

3.1. Preliminary: The Concept of Token Mixer

The concept token mixer is a structure that perform token mixing functions in a given vision backbone. It allows information aggregation from different spatial positions [46]. For instance, self-attention module serves as the token mixer in ViT [14] by performing the attention function in parallel between components in queries, keys and values matrices, which are linearly projected from the input feature. Moreover, ResMLP [37] applies a cross-patch linear sublayer by treating Spatial MLP as token mixer. The



(a) RepIdentityFormer Training (b) RepIdentityFormer Inference

Figure 2. Structural re-parameterization of a RIFormer block.

computational and memory costs of the aforementioned token mixers are quadratic to the image scale.

3.2. Motivation

In this section, we take our eyes on the side effects of token mixers through a quantitative latency analysis on the ViT [14] model. We start with a modified 12-layer ViT-Base model containing only input embedding, without any operation in each of its basic building blocks. Then we gradually add the operation component (*e.g.*, LN, Attention, Channel MLP, *etc*.) to each basic block, and the model finally comes to ViT-Base without the global average pooling layer and the classifier head. For each model, we take a batch size of 2048 at 224^2 resolution with one A100 GPU and calculate the average time over 30 runs to inference that batch. The whole process is repeated for three times and we take the medium number as the statistical latency. As shown in Fig. 1, after stacking the regular number of 12 layers, token mixer can bring an additional latency of 1433.6ms, occupying about 46.3% of the backbone.

According to the above analysis, token mixer brings significant side effects on the latency to the model, which makes it limited for realistic application. The observation naturally raises a question: *can we keep the backbone effective but removing token mixer?* Specifically, a recent work [47] introduces the MetaFormer model without any token mixer in its basic building block and finds that it raises a non-negligible performance degeneration. Based on those findings, we propose to exploit the *full potential* of the extremely simple model by incorporating the inductive bias with the advanced optimization strategies, such as *knowledge distillation* [18, 38, 51], structural re-parameterization [12, 13], *etc*. And we present all the exploration details in the remaining of this work.

4. Exploring RIFormer: A Roadmap

In this section, we present a trajectory going from a fully supervised approaches for RIFormer to more advanced training paradigms. During the journey we investigate and develop different optimization schemes for transformer-like models, while maintaining the inference-time model as the same. The baseline RIFormer we use has exactly the same macro architecture and model size as recently-developed MetaFormer [46], the difference only lies in the fact that no token mixer is used in its basic building blocks during inference. We control the computational complexity of RIFormer-S12 models comparable to PoolFormer-S12 [46], with about 12M parameters and 1.8G MAC. All RIFormer-S12 models in this section are trained and evaluated on ImageNet-1K for 120 epochs. The details of hyper-parameters are shown in Sec.1 of the appendix. The roadmap of our exploration is as follows.

Token Mixer	Training recipe	ImageNet top-1 acc (%)
Pooling	CE Loss	**75.01**
Identity	CE Loss	72.31

Table 1. Results of different token mixers on MetaFormer using cross-entropy loss.

TM	Label	Teacher	ImageNet top-1 acc (%)
Identity	✓	✗	72.31
Identity	✓	hard	73.51
Identity	✗	hard	72.86
Identity	✓	soft	73.64
Identity	✗	soft	**74.05**

Table 2. Results of different teacher type in normal/label-free RIFormer-S12 with identity mapping as token mixer.

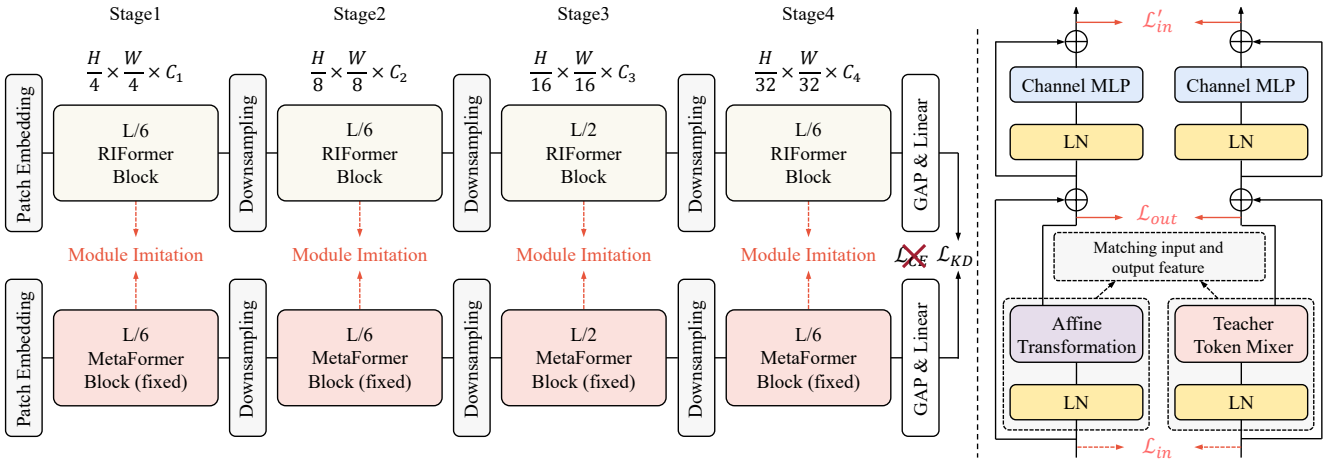
TM	Label	KD type	ImageNet top-1 acc (%)
Affine	✓	✗	72.25
Affine	✓	hard	73.44
Affine	✗	hard	72.77
Affine	✓	soft	72.10
Affine	✗	soft	**74.07**

Table 3. Results of different distillation type in normal/label-free RIFormer-S12 with affine transformation as token mixer.

4.1. Vision Backbone Without Token Mixer

Our exploration is directed to remove token mixer in each basic block of a inference-time model vision backbone to obtain a higher inference speed while striving to keep the performance. Thus, we start with a RIFormer-S12 model with a fully supervised training scheme using CE loss, mainly follows [46]. As a performance reference, we compare the results with PoolFormer-S12, since it use only basic pooling operation as token mixer and the performance gap can thus, be attributed to the absence of basic token mixing function. As shown in Tab. 1, RIFormer-S12 with a trivial supervised training can lead to an unacceptable performance drop (2.7% top-1 accuracy) compared to PoolFormer-S12. The results show that without token mixer in each building block, it is limited for regular supervised learning in helping the model learn useful information from images, calling for advanced training procedure.

We then investigate and modify a series of training paradigms to improve the inferior baseline performance,



(a) Overall training procedure with L RIFormer blocks (b) Module imitation

Figure 3. **(a) Overall training procedure of RIFormer.** Following the macro and micro design of [46], RIFormer removes token mixer in each block. **(b) Module imitation technique** aims to mimic the behavior of token mixer via a simple affine transformation.

which can be summarized as 1) knowledge distillation, 2) teacher type influence, 3) structural re-parameterization, 4) the proposed module imitation technique, 5) load partial parameters from teacher. Since we aim at exploring the influence of different advanced training recipes instead of network architecture, inference-time model architecture is always kept the same at intermediate steps. Next, we share 5 useful guidelines for training RIFormer.

4.2. Distillation Paradigm Design

We now study the knowledge distillation [18, 38] of a RIFormer student by a general vision backbone teacher with token mixer, and summarize how to effectively utilize the "soft" labels coming from the strong teacher network.

Guideline 1: soft distillation without using ground-truth labels can be effective for student without token mixer. Basically, most of the existing KD methods are designed for models with token mixer. For example, it is common practice to help a student *convnet* by learning from both ground-truth labels and the soft labels predicted by a teacher convnet. Moreover, some observations from DeiT [38] show that using the hard labels instead of soft labels as a supervised target, can improves *transformer* significantly. In contrast, the token mixer free backbone do not have explicit patch aggregating modules in its basic block. The distillation of it is should be thus, different from that of conventional backbones. Specifically, although RIFormer shares the same macro structure as transformer, it still cannot be treated as a student *transformer* because we have deliberately removed the token mixer from each building block. However, we also do not prefer viewing it as a pure *convnet* since RIFormer bears a resemblance to transformer in terms of macro/micro-level architecture design. Therefore, we are motivated to explore a suitable KD method for RIFormer with promising performance.

Typically, the cross-entropy objective is to assist a student network reproduce the hard accurate label, and we argue that the process may be unsuitable for RIFormer. First, the ground-truth hard label can be transformed to a soft distribution by label-smoothing regularization [35], with weights $1-\varepsilon$ for the true label and ε shared each classes. The unlearned uniform distribution across the negative classes is less informative, and may interfere with the learned soft distribution given by teacher. Second, 1×1 convolutions actually dominate basic building block in RIFormer, "mixing" only the per-location features but not spatial information. Such a simplified design may require richer information in the supervised labels. To demonstrate this, Tab. 2 compare the performance of four different settings. The default teacher is a GFNet-H-B [31] (54M parameters). Hard distillation with true labels improve the accuracy from 72.31% to 73.51%. It shows that a teacher with token mixer has a positive effect on a student without token mixer. In fact, the combination of using a soft distillation without true labels performs the best, improving the network performance to 74.05%.

Remark 1. Supervised learning with true label does not seem to be the most suitable way for a crude model without token mixer. A teacher with token mixer can help to guide the training, but still fails to fully recover the performance gap from removing token mixer, calling for other strategies.

4.3. Re-parameterization for Identity Mapping

Guideline 2: using affine transformation without tailored distillation, is hard to recover the performance degredation. In this part, we adopt the idea of *Structural Reparameterization* [9, 12, 13] methodology, which usually takes a powerful model for training and equivalently converts to a simple model during inference. Specifically, the inference-time token mixer module in RIFormer can be

TM	Feat	Rel	Layer	ImageNet top-1 acc (%)
Affine	0	0	-	74.07
Affine	40	0	6	74.49
Affine	60	0	6	74.77
Affine	80	0	6	74.81
Affine	80	10	6	75.08
Affine	80	20	6	74.82
Affine	80	40	6	75.00
Affine	80	20	4	**75.13**

Table 4. Results of different module imitation setting.

viewed as an identity mapping following a LN layer. Thus, the training-time module should satisfy at least two basic requirements: 1) *per-location* operator for allowing equivalent transformation; 2) *parametric* operator for allowing extra representation ability. Accordingly, we apply an affine transformation operator to replace the identity mapping during training, which only performs channel-wise scaling and shifts, as shown in Fig. 2. The affine operator and its preceding LN layer can be converted into a LN with modified weights, thus it can be equivalently converted into an identity mapping during inference. Denote the input feature as $\mathrm{M} \in \mathbb{R}^{N\times C\times H\times W}$, the affine operator can be expressed as:

$$\mathrm{Affine}(\mathrm{M}, \mathbf{s}, \mathbf{t})_{:,i,:,:} = \mathbf{s}_i \mathrm{M}_{:,i,:,:} + \mathbf{t}_i - \mathrm{M}_{:,i,:,:}, \tag{1}$$

where $\mathbf{s} \in \mathbb{R}^C$ and $\mathbf{t} \in \mathbb{R}^C$ are learnable weight vectors. We follow [46] to add a subtraction of the input during implementation due to the residual connection, and thus does not merge the first and third terms in Eq. 1. Then, we describe how to merge the affine transformation into its preceding LN layer, so the training-time model can be equivalently converted to model for deploy but no longer has token mixer in its blocks. We use $\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\gamma}, \boldsymbol{\beta}$ as the mean, standard deviation and learned scaling factor and bias of the preceding LN layer. Denote $\mathrm{T}^{(a)} \in \mathbb{R}^{N\times C\times H\times W}$, $\mathrm{T}'^{(a)} \in \mathbb{R}^{N\times C\times H\times W}$ respectively as the input and output of an affine residual sub-block in Fig. 2-(a). During the training time, we have:

$$\mathrm{T}'^{(a)} = \mathrm{Affine}(\mathrm{LN}(\mathrm{T}^{(a)}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\gamma}, \boldsymbol{\beta}), \mathbf{s}, \mathbf{t}) - \mathrm{T}^{(a)} \tag{2}$$

where LN is the LN function, which is implemented by GroupNorm API in PyTorch (setting the group number as 1) following [46]. During inference time, there only exists an identity mapping followed by a LN layer in the residual sub-block. Thus, we have:

$$\mathrm{T}'^{(a)} = \mathrm{LN}(\mathrm{T}^{(a)}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\gamma}', \boldsymbol{\beta}'), \tag{3}$$

where $\boldsymbol{\gamma}'$ and $\boldsymbol{\beta}'$ are the weight and bias parameters of the merged LN layer. Based on the equivalence of Eq. 2 and Eq. 3, for $\forall 1 \leq i \leq C$, we have:

$$\boldsymbol{\gamma}'_i = \boldsymbol{\gamma}_i(\mathbf{s}_i - 1), \quad \boldsymbol{\beta}'_i = \boldsymbol{\beta}_i(\mathbf{s}_i - 1) + \mathbf{t}_i, \tag{4}$$

The proof and PyTorch-like code of the affine transformation and re-parameterization process is shown in Sec.2 and Sec.3 of the appendix, respectively. Since the LN layer does not have a pre-computed mean and standard deviation during inference time, their specific values are input adaptive, which do not affect the equivalence of transform.

Teacher (T)	T.acc (%)	MI	ImageNet top-1 acc (%)
PoolFormer-M48 [46]	82.5	✗	73.63
Swin-B* [24]	85.2	✗	73.12
Pyramid ViG-B [15]	83.7	✗	73.25
GFNet-H-B [31]	82.9	✗	74.07
PoolFormer-M48 [46]	82.5	✓	74.83
Swin-B* [24]	85.2	✓	74.52
Pyramid ViG-B [15]	83.7	✓	74.25
GFNet-H-B [31]	82.9	✓	**75.13**

Table 5. Results of different teachers on RIFormer-S12 w/ or w/o module imitation (MI). * indicates ImageNet-22K pre-training.

Remark 2. Compare Tab. 3 with Tab. 2, directly applying structural re-parameterization method shows no advantages. We attribute this phenomenon to the fact that the affine transformation in the LN layer is a *linear transformation* that can be directly merged with the extra affine operator we introduced (if do not add any nonlinear function in between). Therefore, if both are supervised only by the output of the model, the potential of the additional parameters may not be fully exploited. Meanwhile, the isomorphic design of teacher and student inspires us to explore suitable methods for knowledge transfer of *modules* at each layer.

4.4. Module Imitation

Guideline 3: the proposed block-wise knowledge distillation, called module imitation, helps leveraging the modeling capacity of affine operator. The previous KD methods we tried only focus on the output of between teacher and student networks. We propose *module imitation (MI)* method, which present to utilize the useful information in the teacher's token mixer. Specifically, a pre-trained PoolFormer-S12 [46] is utilized as a teacher network. As shown in Fig. 3, we expect the simple affine operator (with its preceding LN layer) to approximate the behavior of that of a basic token mixer during training. Denote $f(\cdot), \mathrm{T}^{(a),m} \in \mathbb{R}^{N\times C\times H\times W}, m \in \mathcal{M}$ as the affine operator and the input of the m-th layer of RIFormer in which $\mathcal{M}$ is the intermediate layers set we used, and $g(\cdot), \mathrm{T}^{(t),m} \in \mathbb{R}^{N\times C\times H\times W}, m \in \mathcal{M}$ are that of the teacher network, respectively. We abbreviate $\mathrm{LN}(\cdot, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\gamma}, \boldsymbol{\beta})$ as $\mathrm{LN}(\cdot)$ for simplicity. The mean squared error (MSE) of the inputs between the LN layer of affine operator and token mixer can be calculated as:

$$\mathcal{L}_{in} = \alpha_1 \|\mathrm{LN}(\mathrm{T}^{(a),m}) - \mathrm{LN}(\mathrm{T}^{(t),m})\|_F^2, \tag{5}$$

where $\alpha_1 = 1/NCHW$. Note that the input feature of the current layer is the output feature of the previous one. Therefore, we propose to match the output features of this

block (*i.e.*, the input features of the next subsequent block) in practice, which can be seen as a hidden state distillation in transformers [16, 19, 40, 41, 51].

$$\mathcal{L}'_{in} = \alpha_1 \|\mathrm{T}^{(a),m+1} - \mathrm{T}^{(t),m+1}\|_F^2, \tag{6}$$

The hidden-state distillation based on relation matrices [16, 51] is then applied on the output feature:

$$\mathcal{L}_{rel} = \alpha_2 \|\mathcal{R}(\mathrm{T}^{(a),m+1}) - \mathcal{R}(\mathrm{T}^{(t),m+1})\|_F^2, \tag{7}$$

where $\alpha_2 = 1/NH^2W^2$, $\mathcal{R}(T) = \tilde{T}\tilde{T}^{\mathsf{T}}$, $\tilde{T}$ denotes normalize T at the last dimension. Considering the MSE of the outputs between affine operator and token mixer:

$$\mathcal{L}_{out} = \alpha_1 \|f(\mathrm{LN}(\mathrm{T}^{(a),m})) - g(\mathrm{LN}(\mathrm{T}^{(t),m}))\|_F^2, \tag{8}$$

Combining Eq. 6, Eq. 7 and Eq. 8, the final loss function with module imitation is defined as:

$$\mathcal{L} = \mathcal{L}_{soft} + \lambda_1 \mathcal{L}'_{in} + \lambda_2 \mathcal{L}_{out} + \lambda_3 \mathcal{L}_{rel}, \tag{9}$$

where $\mathcal{L}_{soft}$ is the soft logit distillation target in Sec. 4.2, λ_1, λ_2, λ_3 is the hyper-parameter for seeking the balance between loss functions. In Tab. 4, Feat and Rel are number of epochs of using $(\mathcal{L}'_{in}, \mathcal{L}_{out})$ and $\mathcal{L}_{rel}$, Layer represents the number of intermediate layers we used. The results show positive effect of module imitation on the student RIFormer in different circumstances. With a 4 layer setting and the usage of affine operator, we get the best result of 75.13%, already surpassing the PoolFormer-S12's result of 75.01% in Tab. 1. From now on, we will use this setting.

Remark 3. We deem a reason for that phenomenon might be that module imitation helps the affine operator implicitly benefit from the supervision of the teacher's token mixer, while not losing the convenience of explicitly merging the preceding LN layer. Besides, we find module imitation can effectively shift the feature distribution closer to the teacher network and show larger *Effective Receptive Fields (ERFs)*. Please refer to Sec. 5.3 for details.

Guideline 4: teacher with large receptive field is beneficial to improve student with limited receptive field. Tab. 5 compares student performance with different teacher architectures. Although GFNet-H-B [31] does not show the highest ImageNet top-1 performance among teachers, it can still serves as a better choice, no matter whether module imitation is used or not.

Remark 4. The fact is probably attributed to the receptive field gap between teacher and student. As explained by [1], inductive bias can be transferred from one model to another through distillation. According to this study, model with large receptive field (*e.g.*, GFNet with learnable *global filters* in the frequency domain) can be better teacher for student RIFormer with limited receptive field.

Guideline 5: loading the pre-trained weight of teacher model (except the token mixer) into student improve the convergence and performance. Our method can be categorized as a model compression technique that aims at removing the token mixer in basic blocks for acceleration. Inspried by previous methods, including knowledge distillation [32, 34], quantization [22, 26], and model acceleration [29] that initialize the weights of the light-weight network using (or partly using) the corresponding weights of the pre-trained heavy network, we explore a suitable initialization method. Since our goal is to remove only the token mixer, the weights of the remaining part still remain and are not paid enough attention in the previous journey. We observe that initializing the weights of RIFormer (except the affine operator) with the corresponding teacher network further boost the performance from 75.13% to 75.36%. This brings us to the final paradigm for training RIFormer.

Closing remarks. So far, we have finished our exploration and discovered a suitable paradigm for training the RIFormer. It has the approximately the same macro design with MetaFormer [46], but does not require any token mixer. Equipped with the propsoed optimization methods, RIFormer can outperform complicated models with token mixers for ImageNet-1K classification. These encouraging findings inspire us to answer the following questions in the next section. 1) The scaling behavior of such extremely simple architecture with our training paradigm. 2) The generalizability of the paradigm on different teachers.

5. Experiments

5.1. Image classification

Setup. For ImageNet-1K [8] with 1.2M training images and 50000 validation images, we generally apply the training scheme in [46] while following the guidelines in Sec. 4. The data augmentation contains MixUp [50], CutMix [48], CutOut [52] and RandAugment [7]. As a model compression work on removing token mixer, bridging the performance gap caused by removing the token mixer is definitely our first priority, instead of proposing a strong baseline. Therefore, we use a prolonged the training epochs of 600. We also finetune the pre-trained models for 30 epochs with input resolution of 384^2. More details are in the appendix.

Main results. Tab. 6 shows the results of RIFormer on ImageNet classification. We pay main attention to the *throughput* metrics since our primary consideration is to satisfy latency requirements for edge devices. As expected, favorable speed advantage is achieved since RIFormer does not contain any token mixer in its building block, compared with other type of backbones. Surprisingly, with such fast inference, RIFormers successfully remove all token mixers using our training approach without affecting the performance. For example, RIFormer-M36 can pro-

Token Mixer	Outcome Model	Image Size	Params (M)	MACs (G)	Throughput (images/s)	Top-1 (%)
Convolution	RSB-ResNet-34 [17, 43]	224	22	3.7	6653.75	75.5
	RSB-ResNet-50 [17, 43]	224	26	4.1	2732.85	79.8
	RSB-ResNet-101 [17, 43]	224	45	7.9	1856.48	81.3
	RSB-ResNet-152 [17, 43]	224	60	11.6	1308.26	81.8
Attention	DeiT-S [38]	224	22	4.6	3092.02	79.8
	DeiT-B [38]	224	86	17.5	1348.76	81.8
	PVT-Small [42]	224	25	3.8	1622.53	79.8
	PVT-Medium [42]	224	44	6.7	1190.48	81.2
	PVT-Large [42]	224	61	9.8	865.33	81.7
Spatial MLP	MLP-Mixer-B/16 [36]	224	59	12.7	1855.45	76.4
	ResMLP-S24 [37]	224	30	6.0	3228.75	79.4
	ResMLP-B24 [37]	224	116	23.0	298.94	81.0
	Swin-Mixer-T/D6 [24]	256	23	4.0	1625.59	79.7
	Swin-Mixer-B/D24 [24]	224	61	10.4	1131.60	81.3
2D FFT	GFNet-H-Ti [31]	224	15	2.1	1979.56	80.1
	GFNet-H-S [31]	224	32	4.6	1434.19	81.5
	GFNet-B [31]	224	43	7.9	1771.07	80.7
	GFNet-H-B [31]	224	54	8.6	939.20	82.9
Pooling	PoolFormer-S12 [46]	224	12	1.8	4160.18	77.2
	PoolFormer-S24 [46]	224	21	3.4	2140.20	80.3
	PoolFormer-S36 [46]	224	31	5.0	1440.37	81.4
	PoolFormer-M36 [46]	224	56	8.8	1009.45	82.1
	PoolFormer-M48 [46]	224	73	11.6	761.93	82.5
None	RIFormer-S12$^{\diamond}$	224	12	1.8	4899.80 (↑ 17.8%)	76.9
	RIFormer-S24$^{\diamond}$	224	21	3.4	2530.48 (↑ 18.2%)	80.3
	RIFormer-S36$^{\diamond}$	224	31	5.0	1699.94 (↑ 18.0%)	81.3
	RIFormer-M36$^{\diamond}$	224	56	8.8	1185.33 (↑ 17.4%)	82.6
	RIFormer-M48$^{\diamond}$	224	73	11.6	897.05 (↑ 17.7%)	82.8
	RIFormer-S12‡	384	12	5.4	1586.51	78.3
	RIFormer-S24‡	384	21	10.0	819.40	81.4
	RIFormer-S36‡	384	31	14.7	552.07	82.2
	RIFormer-M36‡	384	56	25.9	403.15	83.4
	RIFormer-M48‡	384	73	34.1	304.43	83.7

Table 6. **Results of models with different types of token mixers on ImageNet-1K.** $^{\diamond}$ denotes training with prolonged 600 epochs. ‡ denotes fine-tuning from the ImageNet pre-trained model for 30 epochs.

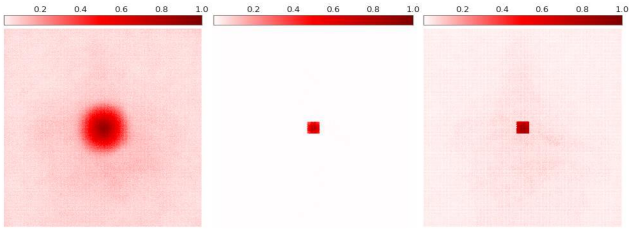
cess more than 1185 images at 224^2 resolution per second, with the top-1 accuracy of 82.6%. In comparison, the recent baseline PoolFormer-M36 [46] with a Pooling token mixer, can process 1009 images of the same size with a worse 82.1% accuracy. We also compare with an efficient backbone, GFNet [31]. It conducts token mixing via a global filter, which consists an FFT, an element-wise multiplication, and an IFFT, with a total computational complexity $\mathcal{O}(N \log N)$. With a 939 throughput, GFNet-H-B gets 82.9% accuracy while our RIFormer-M48 can still reaches a comparable 82.8% accuracy with on par throughput of 897. Meanwhile, the body of inference-time RIFormer is dominated by only 1×1 conv following LN, without complex 2D FFT or attention, friendly for hardware specialization.

Notably, without token mixer, RIFormer cannot even perform basic token mixing operation in its building blocks. However, the ImageNet experiments demonstrate that with the proposed training paradigm, RIFormer still shows promising results. We can only deem the reason behind the fact might be that optimization strategy plays a key role. RIFormer is readily a starting recipe for the exploration of optimization-driven efficient network design, and rest assured of the performance with advanced training schemes.

5.2. Ablation studies

Effectiveness of module imitation. As an important way for the extra affine operator to learn suitable weights, module imitation is based on distillation. Therefore, we compare it with the hidden state feature distillation approach (with relations). Taking the paradigm in Sec. 4.2 by soft distillation without CE loss, we get the results in Tab. 7. More details for Sec. 5.2 can be found in Sec.4 of the appendix. With feature distillation, the accuracy is 0.46% lower than that of module imitation, showing module imitation's positive effect on the optimization of the extra weights.

Comparisons of different acceleration strategy. Next, we discuss whether the token removing is better than other sparsification strategies. Based on the PoolFormer [46]



(a) PoolFormer-S12 (b) RIFormer-S12 (c) RIFormer-S12 (MI)

Figure 4. The *Effective Receptive Field (ERF)* of PoolFormer-S12 and RIFormer-S12 with/without using module imitation.

Token Mixer	Feature distillation scheme	Top-1 (%)
Identity	None	74.05
Identity	Feature distill	74.90
Affine	Module imitation	**75.36**

Table 7. Ablation study of the effectiveness of module imitation.

baseline, we first construct a slim PoolFormer-S9 and PoolFormer-XS12 by reducing the depth to 9 and by maintaining about $\frac{5}{6}$ of its width, *i.e.*, embedding dimension, to obtain comparable inference speed with our RIFormer-S12. We also follow the soft distillation paradigm in Sec. 4.2. Tab. 8 shows the results. Directly pruning depths or width cannot render a better performance than ours without latency-hungry token mixer.

Generalization to different teachers. In order to verify the proposed training paradigm a general compression technique, we adopt the architecture modifications in [47] for student and change teacher to the other 4 MetaFormer baselines [47], with teacher token mixer as rand matrices, pooling, separable depthwise convolutions, and attention, respectively. Tab. 9 shows that our method has a positive effect in different depth settings and teacher circumstances.

5.3. Analysis of Module Imitation.

Module imitation (MI) shifts the feature distribution of the RIFormer model to be closer to the teacher. The effect of the module imitation is explicitly shown in Fig. 5. It can be observed that PoolFomer-S12 and RIFormer-S12 show a clear difference in feature distribution of Stage 1 and Stage 4. After applying the proposed module imitation, the distribution of RIFormer-S12 are basically shifted toward that of the PoolFomer-S12, demonstrating its effect on helping student learn useful knowledge from the teacher.

Module imitation helps showing larger Effective Receptive Field (ERF). ERF [27] reflects how large an area of the image the trained model can respond to or capture information about how large an object. We follow [12, 20] to visualize the ERF via measuring the aggregated contributions of each pixel of the input to the central points of the output feature. Since RIFormer removes all token mixers, it exhibits an expectedly much smaller ERF than PoolFormer, as shown in Fig. 4. There is only a square of pixels emerging with red color in the whole region, much smaller than PoolFormer. However, surprisingly, we can observe that red color widely distributed in all locations after training with module imitation. It seems that although there's no explicit structural change, module imitation still help change the learned weights and show larger ERF.

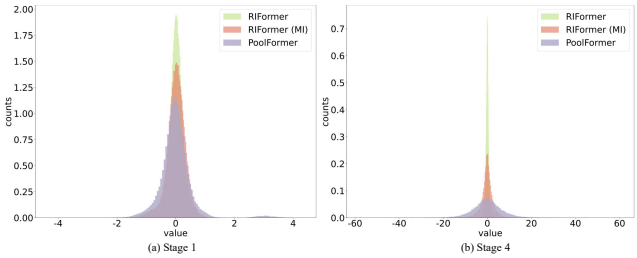


Figure 5. Visualization of the feature distribution of the first and last stage of PoolFormer-S12 and RIFormer-S12.

Model	Type	Throughput	Top-1 (%)
PoolFormer-S12	None	4160.18	75.01
PoolFormer-S9	Depth	5025.71	74.78
PoolFormer-XS12	Width	4780.28	75.11
RIFormer-S12	TM	4899.80	**75.36**

Table 8. Results of comparison with depth or width slimming.

Token Mixer	Teacher	Top-1 (%)
Affine (12 layers)	None	72.75
Affine (12 layers)	RandFormer-S12 [47]	**75.62**
Affine (12 layers)	PoolFormer V2-S12 [47]	**75.87**
Affine (18 layers)	None	75.01
Affine (18 layers)	ConvFormer-S18 [47]	**77.53**
Affine (18 layers)	CAFormer-S18 [47]	**77.26**

Table 9. Results of generalization to other teachers.

6. Limitations and Conclusion

This paper investigates removing token mixer of the basic building block in a vision backbone, motivated by their considerable latency cost. To keep the remaining structure still effective, we thoroughly revisits the training paradigm. We observe that appropriate optimization strategy can effectively help a token mixer-free model learn useful knowledge from another model, boosting its performance and bridge the gap caused by incomplete structure. Limitations are that more vision tasks, including detection, deblurring, *etc*. are not discussed, and we will work on them in the future.

Acknowledgement

This paper is partly supported by the National Key R&D Program of China No.2022ZD0161000, the General Research Fund of HK No.17200622, Shanghai Postdoctoral Excellence Program (No.2022235), the National Natural Science Foundation of China (Grant No.61991450) and the Shenzhen Science and Technology Program (JCYJ20220818101001004).

References

[1] Samira Abnar, Mostafa Dehghani, and Willem Zuidema. Transferring inductive biases through knowledge distillation. *arXiv preprint arXiv:2006.00555*, 2020. 6

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2

[3] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022. 2

[4] Han Cai, Chuang Gan, and Song Han. Efficientvit: Enhanced linear attention for high-resolution low-computation visual recognition. *arXiv preprint arXiv:2205.14756*, 2022. 2

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1

[6] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *CVPR*, 2021. 1

[7] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR*, 2020. 6

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2, 6

[9] Xiaohan Ding, Honghao Chen, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Repmlpnet: Hierarchical vision mlp with re-parameterized locality. In *CVPR*, 2022. 2, 4

[10] Xiaohan Ding, Honghao Chen, Xiangyu Zhang, Kaiqi Huang, Jungong Han, and Guiguang Ding. Re-parameterizing your optimizers rather than architectures. *arXiv preprint arXiv:2205.15242*, 2022. 2

[11] Xiaohan Ding, Tianxiang Hao, Jianchao Tan, Ji Liu, Jungong Han, Yuchen Guo, and Guiguang Ding. Resrep: Lossless cnn pruning via decoupling remembering and forgetting. In *ICCV*, 2021. 2

[12] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *CVPR*, 2022. 2, 3, 4, 8

[13] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *CVPR*, 2021. 1, 2, 3, 4

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020. 1, 2, 3

[15] Kai Han, Yunhe Wang, Jianyuan Guo, Yehui Tang, and Enhua Wu. Vision gnn: An image is worth graph of nodes. In *NeurIPS*, 2022. 5

[16] Zhiwei Hao, Jianyuan Guo, Ding Jia, Kai Han, Yehui Tang, Chao Zhang, Han Hu, and Yunhe Wang. Learning efficient vision transformers via fine-grained manifold distillation. In *NeurIPS*, 2022. 2, 6

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 7

[18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 3, 4

[19] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. In *EMNLP*, 2020. 6

[20] Bum Jun Kim, Hyeyeon Choi, Hyeonah Jang, Dong Gu Lee, Wonseok Jeong, and Sang Woo Kim. Dead pixel test using effective receptive field. *arXiv preprint arXiv:2108.13576*, 2021. 8

[21] Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Xuan Shen, Geng Yuan, Bin Ren, Hao Tang, et al. Spvit: Enabling faster vision transformers via latency-aware soft token pruning. In *ECCV*, 2022. 2

[22] Yanjing Li, Sheng Xu, Baochang Zhang, Xianbin Cao, Peng Gao, and Guodong Guo. Q-vit: Accurate and fully quantized low-bit vision transformer. In *NeurIPS*, 2022. 6

[23] Jing Liu, Zizheng Pan, Haoyu He, Jianfei Cai, and Bohan Zhuang. Ecoformer: Energy-saving attention with linear complexity. In *NeurIPS*, 2022. 2

[24] 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. 1, 2, 5, 7

[25] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022. 1

[26] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *ECCV*, 2018. 6

[27] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *NeurIPS*, 2016. 8

[28] 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 *CVPR*, 2022. 2

[29] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, 2021. 1, 2, 6

[30] Yongming Rao, Wenliang Zhao, Yansong Tang, Jie Zhou, Ser-Nam Lim, and Jiwen Lu. Hornet: Efficient high-order spatial interactions with recursive gated convolutions. In *NeurIPS*, 2022. 1

[31] Yongming Rao, Wenliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. Global filter networks for image classification. In *NeurIPS*, 2021. 1, 2, 4, 5, 6, 7

[32] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 6

[33] Han Shu, Jiahao Wang, Hanting Chen, Lin Li, Yujiu Yang, and Yunhe Wang. Adder attention for vision transformer. In *NeurIPS*, 2021. 2

[34] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for bert model compression. In *EMNLP*, 2019. 6

[35] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 4

[36] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, et al. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021. 7

[37] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *TPAMI*, 2022. 2, 7

[38] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 1, 2, 3, 4, 7

[39] Jiahao Wang, Mingdeng Cao, Shuwei Shi, Baoyuan Wu, and Yujiu Yang. Attention probe: Vision transformer distillation in the wild. In *ICASSP*, 2022. 2

[40] Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *ACL*, 2021. 6

[41] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *NeurIPS*, 2020. 6

[42] 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. 7

[43] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021. 7

[44] Kan Wu, Jinnian Zhang, Houwen Peng, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. Tinyvit: Fast pretraining distillation for small vision transformers. In *ECCV*, 2022. 2

[45] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*, 2021. 1

[46] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *CVPR*, 2022. 1, 2, 3, 4, 5, 6, 7

[47] Weihao Yu, Chenyang Si, Pan Zhou, Mi Luo, Yichen Zhou, Jiashi Feng, Shuicheng Yan, and Xinchao Wang. Metaformer baselines for vision. *arXiv preprint arXiv:2210.13452*, 2022. 1, 3, 8

[48] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 6

[49] Sergey Zagoruyko and Nikos Komodakis. Diracnets: Training very deep neural networks without skip-connections. *arXiv preprint arXiv:1706.00388*, 2017. 2

[50] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 6

[51] Jinnian Zhang, Houwen Peng, Kan Wu, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. Minivit: Compressing vision transformers with weight multiplexing. In *CVPR*, 2022. 2, 3, 6

[52] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, 2020. 6