






AMixer: Adaptive Weight Mixing for Self-attention Free Vision Transformers

Yongming Rao^{1,2} , Wenliang Zhao^{1,2} , Jie Zhou^{1,2}, and Jiwen Lu^{1,2} 

¹ Department of Automation, Tsinghua University, Beijing, China
zhaowl20@mails.tsinghua.edu

² Beijing National Research Center for Information Science and Technology,
Beijing, China
{jzhou, lujiwen}@tsinghua.edu.cn

Abstract. Vision Transformers have shown state-of-the-art results for various visual recognition tasks. The dot-product self-attention mechanism that replaces convolution to mix spatial information is commonly recognized as the indispensable ingredient behind the success of vision Transformers. In this paper, we thoroughly investigate the key differences between vision Transformers and recent all-MLP models. Our empirical results show the superiority of vision Transformers mainly comes from the data-dependent token mixing strategy and the multi-head scheme instead of query-key interactions. Inspired by this observation, we propose a computationally and parametrically efficient operation named adaptive weight mixing to generate attention weights without token-token interactions. Based on this operation, we develop a new architecture named as AMixer to capture both long-term and short-term spatial dependencies without self-attention. Extensive experiments demonstrate that our adaptive weight mixing is more efficient and effective than previous weight generation methods and our AMixer can achieve a better trade-off between accuracy and complexity than vision Transformers and MLP models on both ImageNet and downstream tasks. Code is available at <https://github.com/raoyongming/AMixer>.

Keywords: Vision transformers · Adaptive weight mixing

1 Introduction

Recent advances on vision Transformers have pushed the state-of-the-art of various visual recognition tasks, including image classification [12, 28, 35, 54], semantic segmentation [8, 28, 55], object detection [4, 28] and action recognition [1, 2]. As a step towards less inductive bias in architecture designs, Vision Transformers

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-19803-8_4.

(ViT) [12] and its variants utilize the self-attention mechanism [45] to capture the interactions between different spatial locations by directly learning from the raw data, different from conventional CNNs that are largely relied on human prior knowledge and hand-made choices. More recently, the pure multi-layer perceptrons (MLP) models [41, 42] are proposed to further simplify the designs of vision Transformers. By replacing the self-attention layers with spatial MLPs, all-MLP models exhibit a simple and more efficient approach to mix spatial information for visual recognition tasks. However, empirical results suggest all-MLP models without self-attention usually perform inferiorly compared to vision Transformers [28, 42].

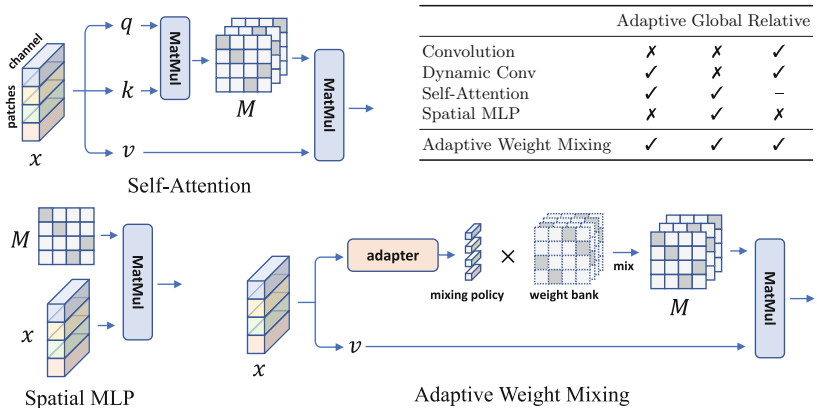


Fig. 1. The main idea of *AMixer*. By analyzing the differences between self-attention and spatial MLP, we propose a new operation named *adaptive weight mixing* to efficiently generate data-dependent spatial mixing weights without token-token interaction. We compare the proposed adaptive weight mixing with prevalent operations by considering three key factors: the input-dependent weight (*adaptive*), global interactions among spatial locations (*global*) and weights based on the relative positions (*relative*) “indicates that only a weak regularization of position-relative weight based on positional embeddings is imposed to self-attention”. Our method combines the advantages of previous operations.

The self-attention mechanism is commonly recognized as the key ingredient behind the success of vision Transformers. Self-attention fully considers the relationships among all spatial locations by generating the spatial mixing weights (*i.e.*, attention weights) using the dot product of two projected versions of the input (*i.e.*, *queries* and *keys*) and applying the weights to another projected input (*i.e.*, *values*). Benefiting from the communications among all tokens, vision Transformers can better model long-range dependencies with fewer blocks compared to convolutions [12, 51]. Since spatial MLP models also consider all possible interactions between any two spatial locations with learnable mixing weights, it is natural to ask: *which specific designs make self-attention more effective? Is there a more efficient way to learn the spatial mixing weights?*

In this paper, we thoroughly investigate the key differences between vision Transformers and recent all-MLP models. Starting from a simple and neat all-MLP architecture [42], we gradually add the designs in vision Transformers to verify their effects. Our empirical results show the superiority of vision Transformers mainly comes from the data-dependent spatial mixing strategy and the multi-head scheme instead of the query-key interactions in self-attention. Based on this observation, we propose a computationally and parametrically efficient operation called *Adaptive Weight Mixing* to generate attention weights without token-token interactions. Our operation is inspired by the spatial MLP in all-MLP models, where the spatial mixing weights are memory-like model parameters and are learned through standard back-propagation. Different from previous methods that generate the weights with dot-product (*e.g.*, self-attention) or another MLP (*e.g.*, dynamic convolution [15]), our solution makes the static memory *adaptive* to the input by predicting a small weight mixing matrix that linearly blends a set of static weights from a weight bank. By doing so, we avoid the heavy computation of dot-product ($\mathcal{O}(N^2D)$ for N tokens with D -dim features) and a large number of parameters for weight generation ($\mathcal{O}(N^2D)$ for generating $N \times N$ weights from a D -dim feature). Moreover, inspired by the relative positional embeddings [28, 37, 38], we propose to add the symmetric regularization to the weight bank, where the weight of two positions is only related to their relative position instead of their absolute positions. This strategy can largely reduce the number of parameters in the weight bank since only $\mathcal{O}(N)$ parameters need to be stored. The relative weight also adds the structural priors to the model and makes the input resolution of our model more flexible. Our key idea is illustrated in Fig. 1. We also summarize the differences between our method and other prevalent operations in deep vision models.

Our new adaptive weight mixing operation can serve as a plug-and-play module to replace the self-attention in vision Transformers or spatial MLP in all-MLP models. We propose a new architecture *AMixer* by replacing the self-attention in various vision Transformers with the new operation, which is able to adaptively capture both long-term and short-term spatial dependencies like vision Transformers while having a more efficient weight generation strategy. Equipped with our adaptive weight mixing operation, the enhanced all-MLP models ResMLP [42] and Swin-Mixer [28] can surpass their vision Transformers counterparts DeiT [43] and Swin Transformers [28]. We also scale up AMixer to obtain a series of hierarchical models with different complexities. Extensive experiments show that our adaptive weight mixing is more efficient and effective than previous weight generation methods. AMixer can attain a better trade-off between accuracy and complexity than vision Transformers and MLP models on both ImageNet [36] and downstream tasks.

2 Related Work

Vision Transformers. The Transformer architecture that is originally designed for the NLP tasks [45] has shown promising results on various vision problems since Dosovitskiy *et al.* [12] introduce it to the image classification problem

with a patch-based design. A large number of works aim to design more suitable architectures for vision tasks. Some methods are also proposed to improve the training strategy and inference efficiency. Although quite a few kinds of powerful models are designed, most of them are still based on the dot-product self-attention mechanism. In this work, we show that the query-key interaction in vision Transformer may not be necessary, which may provide a new view to design more efficient vision Transformers.

MLP-Like Models. Recently, there are several works that question the importance of self-attention in the vision Transformers and propose to use spatial MLP to replace the self-attention layer in the Transformers [27, 41, 42]. Although these models are more efficient than the counterpart vision Transformers by removing the dot-product self-attention, this simple modification may bring two drawbacks: (1) the memory-like static spatial MLP usually exhibit weaker expressive power compared to Transformers [28, 42]; (2) unlike Transformers, MLP-based models are hard to scale up to a new resolution since the weights of the spatial MLPs have fixed sizes. Our work aims to resolve the above issues in MLP-like models by introducing a new weight generation scheme. Since we store the relative weights instead of the absolute weights in the weight bank, our model can adapt to different input resolutions by interpolating the weight bank.

Dynamic Weights. The self-attention mechanism [45] can be viewed as one of the most popular methods to generate dynamic weights. Learning dynamic weights that are conditioned on the input is also a widely studied problem for CNN models. Previous efforts based on convolutional networks usually focus on predicting adaptive convolution kernels that are shared for the different locations [7, 14, 21]. Some works also propose to generate region or position-specific weights to better exploit the location information in CNNs [20, 26]. However, their methods are designed for generating relatively small kernels (*e.g.*, 3×3), which are widely used in CNN-based models. We argue that their generation methods are computational and parametrically expensive to generate large kernels, which makes them hard to scale up to the global interactions considered in this paper. The most related work is Synthesizer [40] which proposes to predict the attention weights using an MLP for NLP tasks. Different from them, we propose a new framework to generate weights by linearly blending weights in a bank, avoiding the large computation and storage cost.

3 Approach

3.1 Vision Transformers and MLP Models: An Unified View

Since vision Transformers [12, 28] start to dominate vision tasks, many Transformer-style architectures that pursue the same goal of reducing inductive bias emerged [33, 41, 42]. Among those, MLP models [27, 41, 42] replace the self-attention by MLPs that are applied across spatial locations, which are very

efficient on modern accelerators. However, the performances of current all-MLP models lag far behind their Transformer counterparts, *e.g.*, 76.6% (ResMLP-12 [42]) *vs* 79.8% (Deit-S [43]) on ImageNet. To examine where the performance drop comes from, we aim to provide an in-depth analysis of the two families of models from a unified perspective. Both vision Transformers and MLP models are built with the following basic block ($\mathbf{X} \in \mathbb{R}^{N \times D}$ is the input, $N = N_H N_W$):

$$\mathbf{X} \leftarrow \mathbf{X} + \text{Mixer}(\mathbf{X}), \quad (1)$$

$$\mathbf{X} \leftarrow \mathbf{X} + \text{MLP}(\mathbf{X}), \quad (2)$$

where the Mixer and the MLP perform spatial mixing and channel mixing, respectively. Generally, the Mixer function can be written in the following form:

$$\begin{aligned} \text{Mixer}(\mathbf{X}) &= \text{Concat}(\mathbf{M}_h(\mathbf{X})\mathbf{V}_h(\mathbf{X}))\mathbf{C}, \\ \mathbf{M}_h &\in \mathbb{R}^{N \times N}, \mathbf{V}_h \in \mathbb{R}^{N \times D_h}, \mathbf{C} \in \mathbb{R}^{D \times D}. \end{aligned} \quad (3)$$

We consider the multi-head occasion in the above formula. For the h -th head, the \mathbf{M}_h matrix characterizes the interactions among the tokens, and the \mathbf{V}_h projects input to $D_h = D/H$ dimension as the *value*. The $\{\mathbf{V}_h\}_{h=1}^H$ weighted by the $\{\mathbf{M}_h\}_{h=1}^H$ are then concatenated and projected by another matrix \mathbf{C} . For vision Transformers [12, 28, 43], the widely used multi-head self-attention (MHSA) can be derived from Eq. (3) by setting $\mathbf{V}_h(\mathbf{X}) = \mathbf{X}\mathbf{W}_h^V$, $\mathbf{C} = \mathbf{W}^O$, and

$$\begin{aligned} \mathbf{M}_h(\mathbf{X}) &= \text{Softmax}\left(\frac{\mathbf{Q}_h(\mathbf{X})\mathbf{K}_h(\mathbf{X})^\top}{\sqrt{D_h}}\right), \\ \mathbf{Q}_h(\mathbf{X}) &= \mathbf{X}\mathbf{W}_h^Q, \mathbf{K}_h(\mathbf{X}) = \mathbf{X}\mathbf{W}_h^K, \end{aligned} \quad (4)$$

where the $\mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V \in \mathbb{R}^{D \times D_h}$ and $\mathbf{W}^O \in \mathbb{R}^{D \times D}$ are the learnable parameters. For all-MLP models [41, 42], the spatial mixing operation is simply performed by a linear layer without the multi-head operation, *i.e.*,

$$H = 1, \mathbf{V}_1(\mathbf{X}) = \mathbf{X}, \mathbf{M}_1(\mathbf{X}) = \mathbf{W}, \mathbf{C} = \mathbf{I}. \quad (5)$$

3.2 Rethinking Self-attention in ViTs

From the analysis in Sect. 3.1, we can identify the differences between the vision Transformers and the all-MLP models. Specifically, vision Transformers contain four components that the all-MLP models do not have: (1) the multi-head scheme; (2) the softmax operation; (3) the V-projection (\mathbf{V}_h) and the C-projection (\mathbf{C}); (4) the dot-product between the query and the key. In this section, we will investigate whether these factors can bring improvements to the vanilla MLP models. The step-by-step results of our analysis are summarized in Table 1.

Multi-head Scheme. The multi-head self-attention [45] is originally proposed to help the model to attend to multiple positions. One nice property of the multi-head scheme is that it does not induce extra FLOPs. However, it is interesting

Table 1. Starting from ResMLP [42] and surpassing DeiT [43]. We gradually add key ingredients to an all-MLP model ResMLP [42] and show the final model (AMixer) outperform the Transformer DeiT [43] with fewer FLOPs and parameters.

Model	Params.	FLOPs	Acc. (%)
ResMLP-12 [42]	15 M	3.0 G	76.6
+ Multi-head scheme	19 M	3.0 G	77.4(+0.8)
+ Extra projections	22 M	3.7 G	78.0(+1.4)
+ Softmax	22 M	3.7 G	78.2(+1.6)
+ Adaptive weight mixing	26 M	3.9 G	79.9(+3.3)
+ Relative attention weight	19 M	3.9 G	80.3(+3.7)
DeiT-S [43]	22 M	4.6 G	79.8

to see that there are rare previous efforts that apply the multi-head scheme to MLPs except [28]. In our experiments, we implement a multi-head MLP model based on ResMLP [42] and find the accuracy on ImageNet can be improved from 76.6% to 77.4%.

Extra Projections. The V-projection and the C-projection are also important, especially when the multi-head scheme is used. Without these projections, interactions across different heads are cut off. We then add the extra projections and find they can further boost the performance by 0.6%.

Softmax. Another difference between self-attention and MLP is the softmax operation. Softmax makes the weighted sum of the row of $\mathbf{V}_h(\mathbf{X})$ bounded and we find it can improve the accuracy to 78.2%.

By far, the performance of the modified MLP is still lower than DeiT [43] and the only difference now is how \mathbf{M}_h is generated. In vision Transformers, the $\mathbf{M}_h(\mathbf{X})$ matrix is *adaptive* because the dot-product between \mathbf{Q}_h and \mathbf{K}_h is conditioned on the input \mathbf{X} while \mathbf{M}_h is simply a memory-like weight in MLPs. Then we ask: can we find another implementation of the adaptive $\mathbf{M}_h(\mathbf{X})$, which is better than the standard dot-product self-attention? To make an attempt towards this interesting question, we propose adaptive weight mixing, a better alternative to self-attention.

3.3 Adaptive Weight Mixing

To obtain the adaptive weights $\{\mathbf{M}_h\}_{h=1}^H$, self-attention requires $2ND^2$ FLOPs to compute the query and key and N^2D to perform the dot-product, such that the total FLOPs is

$$\text{FLOPs(SA)} = \underbrace{N^2D}_{\text{dot-product}} + \underbrace{2ND^2}_{\text{compute query and key}}. \quad (6)$$

In this paper, we propose a more efficient method named *adaptive weight mixing* to achieve a similar function to self-attention. Briefly speaking, our adaptive weight mixing learns a weight bank of size B and predicts a mixing policy for each token to generate the \mathbf{M}_h adaptively. Formally, let the weight bank be $\mathbf{W} \in \mathbb{R}^{B \times N \times N}$, which is learnable during training. We then use an adapter to predict the mixing policy based on the input \mathbf{X} :

$$\boldsymbol{\Pi}(\mathbf{X}) = \text{Adapter}(\mathbf{X}) \in \mathbb{R}^{N \times H \times B}. \quad (7)$$

The mixing policy $\boldsymbol{\Pi}(\mathbf{X})$ is sample-specific and head-specific, which largely increases the diversity. To reduce extra costs induced by predicting the policy, our adapter is designed to be very lightweight as a two-layer MLP.

$$\mathbf{Y} = \text{GELU}\left(\mathbf{X}\mathbf{W}_1^{D \times D'}\right)\mathbf{W}_2^{D' \times HB}, \text{Adapter}(\mathbf{X}) = \text{Reshape}(\mathbf{Y}), \quad (8)$$

where $D' = D/r$ is the hidden dimension with reduction ratio r to reduce computational costs following [21]. With the mixing policy, we can construct the \mathbf{M}_h by

$$\mathbf{M}_{h,i} = \text{Softmax}\left(\sum_{b=1}^B \boldsymbol{\Pi}_{i,h,b} \mathbf{W}_{b,i}\right), \quad (9)$$

where $\mathbf{M}_{h,i}$ denotes the weighting coefficient for the h -th head and the i -th token. The above equation also suggests that our $\mathbf{M}_{h,i}$ is taken from a B -dimension linear space spanned by $\mathbf{W}_{:,i}$. Distinct from self-attention, our method only requires

$$\text{FLOPs}(\text{Ada}) = \underbrace{\frac{ND^2}{r} + \frac{NDHB}{r}}_{\text{adapter}} + \underbrace{N^2BH}_{\text{mixing}} \quad (10)$$

to generate \mathbf{M}_h . In our experiments, the typical values of the hyper-parameters are: $B = 16, H = 8, r = 4$. It is easy to show our adaptive weight mixing is more efficient than self-attention since D is usually much larger than BH (e.g., $D = 384$ in DeiT-S [43]). With adaptive weight mixing, we can achieve 79.9% top-1 accuracy on ImageNet, surpassing DeiT-S [43].

Relative Attention Weight. The vanilla design of the weight bank contains too many parameters ($\mathcal{O}(BN^2)$). To make the weight bank more memory-friendly, we propose another variant that considers 2D relative position following the practice in [28]. Consider two points i, j with coordinates (i_h, i_w) and (j_h, j_w) , we assume the relation between i, j is only related to their relative position $(i_h - j_h, i_w - j_w)$ and irrelevant to their absolute positions. Therefore, there are only $(2N_H - 1) \times (2N_W - 1)$ possible relative positions and we only need to store a weight bank $\mathbf{W}^{\text{rel}} \in \mathbb{R}^{B \times (2N_H - 1) \times (2N_W - 1)}$ instead of the vanilla version $\mathbf{W} \in \mathbb{R}^{B \times N_H N_W \times N_H N_W}$. As is shown in Table 1, the relative attention weight can further bring an improvement of 0.4% on accuracy and significantly reduce the number of parameters of our model.

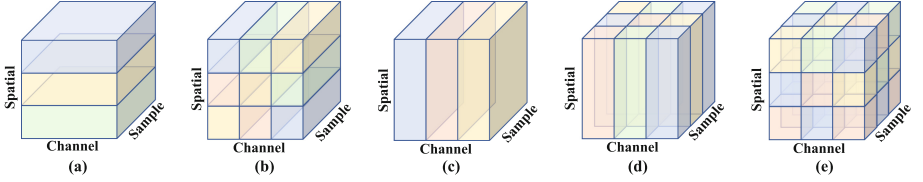


Fig. 2. Comparisons of different weight generation methods. We categorize the methods by whether the weights are data-dependent, spatial-aware, or channel-specific. Some typical operations in each category are: (a) Spatial MLP [42]; (b) Multi-head MLP (Sect. 3.2); (c) Convolution; (d) Dynamic Convolution [7, 14]; (e) Self-attention [45], Synthesizer [40], and Adaptive Weight Mixing (Sect. 3.3).

Discussions. In Fig. 2, we illustrate various weight generation methods in terms of whether they are data-dependent, spatial-aware, or channel-specific. Among those, Dynamic Convolution [7] aggregates the whole convolutional weights through predicted attention coefficients. Our adaptive weight mixing, however, allows more diverse weights since our mixing policies are different across the heads. Besides, convolution-based methods (Dynamic Convolution [7] and Dynamic Depth-wise Convolution [14]) generate spatial-shared weights while our method can produce unique weights for different spatial locations. Our method is also much more efficient than methods that directly predict the parameters using an MLP including Synthesizer [40] and Dynamic Depth-wise Convolution [14]. Our empirical results also show our method is more effective and efficient than existing dynamic weight generation models (see Sect. 4.3).

3.4 Architecture Variants

The adaptive weight mixing can be easily applied to various network architectures based on self-attention and spatial MLP. Apart from the ViT-like architectures, as is already shown in Table 1, our method is conceptually compatible with hierarchical architectures like Swin Transformers and Swin Mixer [28]. We directly apply our adaptive weight mixing to Swin Transformers to build our hierarchical models, including three variants named AMixer-T, AMixer-S, and AMixer-B. We follow the hierarchical design in Swin [28], where a 4-stage architecture is adopted with $[2, 2, n, 2]$ blocks in each stage. We adjust the number of blocks in the third stage, number of heads, and the MLP ratio to scale our model to have similar FLOPs with the Swin [28] series. Different from Swin, we find setting the MLP ratio to 3 can lead to a better trade-off between accuracy and complexity for our models. We also enlarge the window size to 14 for the third stage benefiting from the proposed highly efficient weight generation method without query-key interactions. We also further scale the ViT-like architecture to 17 layers to match the complexity of the original DeiT-S model. We fix the ratio of the bank size B to the number of heads H as 1.5 for our models due to the decent trade-off between complexity and performance. Note that our models enjoy more diversity with more heads but the performance of standard vision

Table 2. Main results on ImageNet. We apply our adaptive weight mixing to ViT-style and hierarchical architectures. We also include the results of multi-head MLP (MH-MLP) for comparison. We find the multi-head mechanism can bring notable improvements to MLPs and show our adaptive weight mixing can further enhance the performance of MLPs, surpassing self-attention.

Model	Params	FLOPs	Acc(%)	Model	Params	FLOPs	Acc(%)
DeiT-S [43]	22 M	4.6 G	79.8	Swin-T [28]	29 M	4.5 G	81.3
ResMLP-12 [42]	15 M	3.0 G	76.6	Swin-Mixer-T/D6 [28]	23 M	4.0 G	79.7
ResMLP-24 [42]	30 M	6.0 G	79.4	Swin-Mixer-B/D24 [28]	61 M	10.4 G	81.3
MH-MLP-DeiT-S	22M	4.5 G	79.2	MH-MLP-T	27 M	4.5 G	80.7
AMixer-DeiT-S	21M	4.5 G	80.8	AMixer-T	28 M	4.5 G	82.0

Transformers saturates with the number of heads increasing as shown in [44]. We follow the settings of Swin and do not add any extra convolutional layers except the patch embedding [12] and merging layers [28], while many previous works obtain significant improvement by adding more convolutional layers to capture local patterns [10, 22, 47, 49]. Moreover, we develop a series of pure MLP models that have identical architecture and designs (including all techniques discussed in Sect. 3.2 and relative attention weight) to AMixer series except the adaptive weight mixing strategy as the static counterpart of our models. We refer to them as “MH-MLP” (multi-head MLP) models. More details can be found in Supplementary Material.

4 Experiments

We conduct extensive experiments to verify the effectiveness of our AMixer models and the new adaptive weight mixing operation. We present the main results on ImageNet [36] and compare them with various state-of-the-art vision Transformers and MLP-like architectures. We also test our models on the downstream transfer learning task on relatively small datasets [24, 25, 31] and semantic segmentation task on the challenging ADE20K [56] dataset. Lastly, we investigate the effectiveness and efficiency of our new designs, and provide the visualization for a better intuitive understanding of our method.

4.1 ImageNet Classification

Setups. We conduct our main experiments on ImageNet [36], a large-scale benchmark dataset for image classification. We follow the standard protocol to train our model on the training set that contains 1.2 M images and evaluate the model on the 50,000 validation images reporting the single-crop top-1 classification accuracy over the 1,000 categories. To fairly compare with previous works on vision Transformers and MLP-like models, we follow the training strategy proposed in DeiT [43], where the model is trained for 300 epochs with the

Table 3. Comparisons with state-of-the-art vision Transformers and MLP-like models on ImageNet. We report the top-1 accuracy on ImageNet as well as the number of parameters and the theoretical complexity in FLOPs. We divide models into three groups based on their complexities. AMixer series exhibit very competitive performances with previous state-of-the-art methods.

Model	Params	FLOPs	Acc(%)	Model	Params	FLOPs	Acc(%)	Model	Params	FLOPs	Acc(%)
gMLP-S [27]	20 M	4.5 G	79.4	ResMLP-36 [42]	45 M	8.9 G	79.7	MLP-Mixer-B [41]	46 M	–	76.4
DeiT-S [43]	22 M	4.6 G	79.8	PVT-Large [47]	61 M	9.8 G	81.7	ViT-B [12]	86 M	17.5 G	79.7
PVT-M [47]	44 M	6.7 G	81.2	T2T-ViT _r -19 [53]	39 M	9.8 G	82.2	gMLP-S [27]	73 M	15.8 G	81.6
Swin-T [28]	29 M	4.5 G	81.3	CrossViT-18 [5]	43 M	9.0 G	82.5	DeiT-B [43]	86 M	17.5 G	81.8
CPVT-S [11]	23 M	4.6 G	81.5	GFNet-H-B [33]	54 M	8.6 G	82.9	CPVT-B [11]	88 M	17.6 G	82.3
GFNet-H-S [33]	32 M	4.6 G	81.5	Swin-S [28]	50M	8.7 G	83.0	T2T-ViT _r -24 [53]	64 M	15.0 G	82.6
T2T-ViT-14 [53]	22 M	5.2 G	81.5	CycleMLP-B4 [6]	52 M	10.1 G	83.0	Swin-B [28]	88 M	15.4 G	83.3
CycleMLP-B2 [6]	27 M	3.9 G	81.6	Twins-SVT-B [10]	56 M	8.3 G	83.2	Twins-SVT-L [10]	99 M	14.8 G	83.7
AMixer-T	26 M	4.5 G	82.0	AMixer-S	46 M	9.0 G	83.5	AMixer-B	83 M	16.0G	84.0

AdamW optimizer [29] and cosine learning rate scheduler. For ViT-style models, we directly adopt the data augmentation and training strategy of DeiT [43]. For hierarchical models, we follow the training techniques of Swin Transformers [28], where EMA model [32] and repeated augmentation [19] are not used during training. Note that we do not use any extra training tricks [22] to directly compare with baseline methods.

Comparisons with Baseline Models. We apply our new adaptive weight mixing operation to two types of widely used architectures: the original ViT [43] model enhanced with DeiT training strategy [43], and high-performance hierarchical models based on Swin Transformers [28]. We also include the results of our multi-head MLP (MH-MLP) for comparison. The results are presented in Table 2. With similar network architecture and identical training configurations, we see our AMixer achieves +1% performance improvement over DeiT model. By applying our operations to a Swin-Mixer-T/D6 and scaling the model to match the complexity, we also observe that our method outperforms the original Swin-T model by 0.7%. Besides, we find that the multi-head mechanism can bring notable improvements to MLPs, where our modified MLP models can largely improve the ResMLP-12 [42] and Swin-Mixer-T/D6 models by 2.6% and 1.0%. The performance gap between AMixer and MH-MLP (+1.6% and 1.3% for ViT and Swin style model respectively) also clearly shows the neat improvement brought by the adaptive weight mixing. These results strongly demonstrate that our adaptive weight mixing is a more efficient and effective method to generate attention weights than self-attention.

Comparisons with State-of-the-Art Models. By further scaling up AMixer models, we build a series of models based on Swin Transformers to compare with state-of-the-art vision Transformers and MLP-like models as shown in Table 3. We see our method achieves very competitive results compared to previous state-of-the-art networks with a relatively simple design. The building block of AMixer

Table 4. Results on transfer learning datasets. We report the top-1 accuracy on the four datasets and the FLOPs of the models.

Model	FLOPs	CIFAR10	CIFAR100	Flowers	Cars
ResNet50 [16]	4.1 G	–	–	96.2	90.0
EfficientNet-B7 [39]	37 G	98.9	91.7	98.8	94.7
ViT-L/16 [12]	190.7 G	97.9	86.4	89.7	–
DeiT-B/16 [43]	17.5 G	99.1	90.8	98.4	92.1
ResMLP-24 [42]	6.0 G	98.7	89.5	97.9	89.5
Swin-T [28]	4.5 G	98.7	88.7	99.6	91.6
AMixer-T	4.5 G	98.9	89.9	99.6	92.9
AMixer-B	16.0 G	99.1	91.0	99.8	92.9

consists of only MLPs and our adaptive weight mixing operations, while many previous works add convolutions [10, 47] to better capture local information or use a more sophisticated architecture [5] instead of the standard four-stage network. Since the mainstream research of developing more powerful vision Transformers still uses self-attention as an indispensable ingredient, we believe our method has the potential of applying to most vision Transformers variants and improving their efficiency.

4.2 Downstream Tasks

Transfer Learning. To evaluate the transferability of our AMixer architecture and the learned representation, we follow the commonly used experimental settings [12, 39, 43] to evaluate AMixer on a set of transfer learning benchmark datasets that contain a relatively small number of samples while having substantial domain gaps from the upstream ImageNet dataset. Specifically, we test our lightweight AMixer-T model and a more powerful model AMixer-B on CIFAR-10 [25], CIFAR-100 [25], Stanford Cars [24] and Flowers-102 [31]. Following the setting of previous works, we initialize the models with the ImageNet pre-trained weights and fine-tune them on the new datasets. The results are presented in Table 4. Our models generally have strong transferability on various downstream datasets. Our models also show competitive performance compared to state-of-the-art CNNs and large-scale vision Transformers with relatively low complexity.

Semantic Segmentation. Semantic segmentation is a widely used downstream task to verify the generality of vision Transformers on dense prediction tasks with high input resolution. We evaluate our AMixer model on the challenging ADE20K [56] dataset following previous works [28, 47], where we train two AMixer models that have similar computational complexities with the basic ResNet-50 and ResNet-101 models [16]. We see in Table 5 that our model outper-

forms the strong Swin models with a similar level of complexity, which suggests our method generalizes well to dense prediction tasks.

4.3 Analysis and Visualization

Robustness & Generalization Ability. We further perform experiments to show our AMixer also has better robustness and generalization ability. For robustness, we consider ImageNet-A, ImageNet-C, FGSM and PGD. ImageNet-A [18] (IN-A) is a challenging dataset that contains natural adversarial examples. ImageNet-C [17] (IN-C) is used to validate the robustness of the model under various types of corruption. We use the mean corruption error (mCE, lower is better) on ImageNet-C as the evaluation metric. FGSM [13] and PGD [30]

Table 5. Semantic segmentation on ADE20K. We report the mIoU on the validation set as well as the number of parameters and theoretical complexity in FLOPs. The models are equipped with the prevalent semantic segmentation method Semantic FPN [23] and UperNet [50]. The FLOPs are measured with 1024×1024 input. We divide the models into two groups that have the similar complexities of widely used ResNet-50 and ResNet-101 models [16] respectively.

Backbone	Semantic FPN [23] 80 k			UperNet [50] 160 k		
	FLOPs	Params	mIoU (%)	FLOPs	Params	mIoU (%)
ResNet50 [16]	183 G	29 M	36.7	952 G	67 M	42.1
Swin-T [28]	182 G	32 M	41.5	940 G	60 M	44.5
AMixer-T	169 G	31 M	43.7	927 G	58 M	46.0
ResNet101 [16]	260 G	48 M	38.8	1029 G	86 M	43.8
Swin-S [28]	274 G	53 M	45.2	1033 G	81 M	47.6
AMixer-S	249 G	51 M	45.9	1021 G	78 M	47.7

Table 6. Evaluation of robustness and generalization ability. We measure the robustness from different aspects, including the adversarial robustness by adopting adversarial attack algorithms including FGSM and PGD and the performance on corrupted/out-of-distribution datasets including ImageNet-A [18] (top-1 accuracy) and ImageNet-C [17] (mCE, lower is better). The generalization ability is evaluated on ImageNet-V2 [34] and ImageNet-Real [3].

Model	FLOPs	Params	ImageNet		Generalization		Robustness			
			Top-1 \uparrow	Top5 \uparrow	IN-V2 \uparrow	IN-Real \uparrow	FGSM \uparrow	PGD \uparrow	IN-C \downarrow	IN-A \uparrow
DeiT-S	4.6 G	22 M	79.8	95.0	68.4	85.6	40.7	16.7	54.6	18.9
Swin-T	4.5 G	28 M	81.2	95.5	69.6	86.6	33.7	7.3	62.0	21.6
AMixer-T	4.5 G	26 M	82.0	96.0	71.2	87.3	40.8	13.3	54.0	25.4
DeiT-B	17.6 G	87 M	82.0	95.6	70.9	86.7	46.4	21.3	48.5	27.4
Swin-B	15.4 G	88 M	83.4	96.5	72.5	87.8	49.2	21.3	54.4	35.8
AMixer-B	16.0 G	83 M	84.0	96.7	73.5	88.0	51.1	26.8	48.6	36.5

are two widely used algorithms that are targeted to evaluate the adversarial robustness of the model by single-step attack and multi-step attack, respectively. For generalization ability, we adopt two variants of ImageNet validation set: ImageNet-V2 [34] (IN-V2) and ImageNet-Real [3] (IN-Real). ImageNet-V2 is a re-collected version of ImageNet validation set following the same data collection procedure of ImageNet, while ImageNet-Real contains the same images as ImageNet validation set but has reassessed labels. We compare two of our models AMixer-T and AMixer-B to both the DeiT [43] and Swin [28] counterparts and find the AMixer models enjoy better generalization ability and robustness (Table 6).

Comparisons with Existing Weight Generation Methods. We first analyze the effectiveness of our new weight generation method compared to previous ones as shown in Table 7. We compare our method with self-attention (DeiT [43]), Synthesizer [40], location-shared weight generation with MLP (DyConv-G [14]) and weight selection (DyConv-S [7]). Under a carefully controlled setting (identical training method following DeiT [43] and same network configurations), we show that our method achieves the best performance among competitive baseline methods with high efficiency, where suggests our method is more suitable and efficient in the scenarios of vision Transformers.

Comparisons with Other Efficient Self-attentions. We compare our adaptive weight mixing with other methods to efficiently approximate self-attention, including Linformer [46], Performer [9], Nystroformer [52] and the linear attention in PVTv2 [48]. For fair comparisons, we use DeiT-S [43] as the basic archi-

Table 7. Comparisons with other weight generation and efficient attention methods. We fix the training strategy and the number of layers or FLOPs to test the effectiveness of different methods.

Model	Params	FLOPs	Acc (%)	Model	FLOPs	Acc (%)
DeiT-S [43]	22 M	4.6 G	79.8	Linformer [46]	4.5 G	77.8
Synthesizer [40]	19 M	3.9 G	78.4	Performer [9]	4.6 G	72.1
DyConv-G [14]	287 M	4.0 G	79.2	Nystroformer [52]	4.6 G	77.1
DyConv-S [7]	30 M	3.7 G	78.6	PVTv2 [48]	4.5 G	79.1
AMixer	19 M	3.9 G	80.3	AMixer	4.5 G	80.8

Table 8. Data efficiency & convergence speed. We compare the performance of our model and the baseline vision Transformer when trained with fewer data or fewer epochs on ImageNet.

Model	Data ratio			Training epoch		
	10%	50%	100%	50	100	300
DeiT-S [43]	34.0	72.9	79.8	65.7	74.4	79.8
AMixer-DeiT-S	48.7	76.3	80.8	69.2	76.9	80.8

ture and directly replace the standard self-attention with different efficient self-attentions. We also ensure the FLOPs of all the models to be $\sim 4.6\text{G}$ by stacking enough layers. The results are summarized in Table 7b. We find those efficient self-attentions all fail to bring improvement over the DeiT-S, while our AMixer-Deit-S can outperform the baseline by a significant margin.

Data Efficiency and Convergence Speed. We compare the performance of our model and the baseline vision Transformer when trained with fewer data or fewer epochs on ImageNet. The results are shown in Table 8. We see the advantage of our model becomes more significant when the model is trained with fewer data or epochs. The results indicate that our adaptive weight mixing operation performs better when data and computation resources are limited.

Adaptive Weights Visualization. To investigate how our \mathbf{M} matrices vary with the input images, we visualize our adaptive weights \mathbf{M} in Fig. 3. For each image, we first find the token (indicated by the red box) that has the highest classification score on the ground truth category, and visualize the weights corresponded to that token. We include the weights of the 1, 3, 5, 7, 10 and 12-th layer and weights of different heads are averaged. Firstly, we find the weights exhibit notable diversity across different images. Secondly, we show our model tends to attend to the most discriminative part of the images (*e.g.*, in the first row, the weights have higher values near the head of the dog). These visualizations show that our adaptive weights generated without token-token interactions have the similar behavior to the weights obtained by self-attention [12].

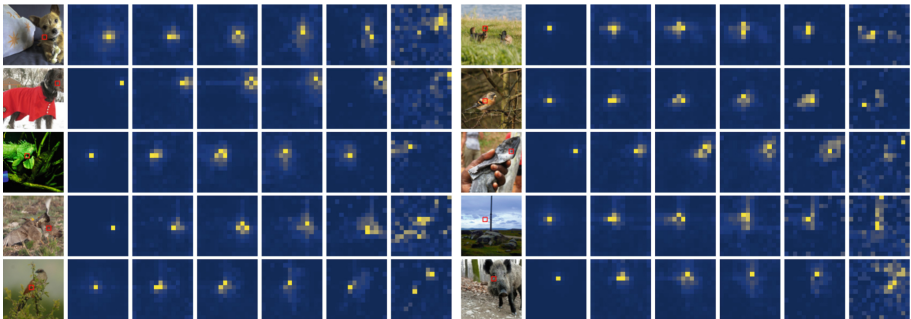


Fig. 3. Our method can generate data-dependent attention weights without token-token interactions. The attention weights in shallow layers usually focus on local regions while the weights of deeper layers adaptively attends to the shape of the whole object.

5 Conclusion

In this paper, we have thoroughly studied the key differences between vision Transformers and recent all-MLP models. Inspired by our empirical results, we

have proposed a new operation named adaptive weight mixing to generate attention weights without token-token interactions. Based on this operation, we have developed a new architecture AMixer that is computationally and parametrically more efficient than vision Transformers. Extensive experiments have shown that our adaptive weight mixing is more efficient and effective than previous weight generation methods. Our models achieve a better trade-off between accuracy and complexity than vision Transformers and MLP models on both ImageNet and downstream tasks.

Acknowledgment. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFA0700802, in part by the National Natural Science Foundation of China under Grant 62125603 and Grant U1813218, in part by a grant from the Beijing Academy of Artificial Intelligence (BAAI).

References

1. Arnab, A., Deghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. arXiv preprint [arXiv:2103.15691](https://arxiv.org/abs/2103.15691) (2021)
2. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? arXiv preprint [arXiv:2102.05095](https://arxiv.org/abs/2102.05095) (2021)
3. Beyer, L., Hénaff, O.J., Kolesnikov, A., Zhai, X., Oord, A.v.d.: Are we done with imagenet? arXiv preprint [arXiv:2006.07159](https://arxiv.org/abs/2006.07159) (2020)
4. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV. pp. 213–229. Springer (2020)
5. Chen, C.F., Fan, Q., Panda, R.: Crossvit: Cross-attention multi-scale vision transformer for image classification. arXiv preprint [arXiv:2103.14899](https://arxiv.org/abs/2103.14899) (2021)
6. Chen, S., Xie, E., Ge, C., Liang, D., Luo, P.: Cyclemlp: A mlp-like architecture for dense prediction. arXiv preprint [arXiv:2107.10224](https://arxiv.org/abs/2107.10224) (2021)
7. Chen, Y., Dai, X., Liu, M., Chen, D., Yuan, L., Liu, Z.: Dynamic convolution: Attention over convolution kernels. In: CVPR. pp. 11030–11039 (2020)
8. Cheng, B., Schwing, A.G., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. NeurIPS (2021)
9. Choromanski, K., Likhoshesterov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al.: Rethinking attention with performers. ICLR (2021)
10. Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., Shen, C.: Twins: Revisiting the design of spatial attention in vision transformers. In: NeurIPS 2021 (2021)
11. Chu, X., Tian, Z., Zhang, B., Wang, X., Wei, X., Xia, H., Shen, C.: Conditional positional encodings for vision transformers. arXiv preprint [arXiv:2102.10882](https://arxiv.org/abs/2102.10882) (2021)
12. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Deghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint [arXiv:2010.11929](https://arxiv.org/abs/2010.11929) (2020)
13. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint [arXiv:1412.6572](https://arxiv.org/abs/1412.6572) (2014)

14. Han, Q., Fan, Z., Dai, Q., Sun, L., Cheng, M.M., Liu, J., Wang, J.: Demystifying local vision transformer: Sparse connectivity, weight sharing, and dynamic weight. arXiv preprint [arXiv:2106.04263](https://arxiv.org/abs/2106.04263) (2021)
15. Han, Y., Huang, G., Song, S., Yang, L., Wang, H., Wang, Y.: Dynamic neural networks: A survey. arXiv preprint [arXiv:2102.04906](https://arxiv.org/abs/2102.04906) (2021)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
17. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. arXiv preprint [arXiv:1903.12261](https://arxiv.org/abs/1903.12261) (2019)
18. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. In: CVPR. pp. 15262–15271 (2021)
19. Hoffer, E., Ben-Nun, T., Hubara, I., Giladi, N., Hoefler, T., Soudry, D.: Augment your batch: Improving generalization through instance repetition. In: CVPR. pp. 8129–8138 (2020)
20. Hu, J., Shen, L., Albanie, S., Sun, G., Vedaldi, A.: Gather-excite: Exploiting feature context in convolutional neural networks. arXiv preprint [arXiv:1810.12348](https://arxiv.org/abs/1810.12348) (2018)
21. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: CVPR. pp. 7132–7141 (2018)
22. Jiang, Z., Hou, Q., Yuan, L., Zhou, D., Jin, X., Wang, A., Feng, J.: Token labeling: Training a 85.5% top-1 accuracy vision transformer with 56m parameters on imagenet. arXiv preprint [arXiv:2104.10858](https://arxiv.org/abs/2104.10858) (2021)
23. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: CVPR. pp. 6399–6408 (2019)
24. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: ICCVW. pp. 554–561 (2013)
25. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
26. Li, D., Hu, J., Wang, C., Li, X., She, Q., Zhu, L., Zhang, T., Chen, Q.: Involution: Inverting the inherence of convolution for visual recognition. In: CVPR. pp. 12321–12330 (2021)
27. Liu, H., Dai, Z., So, D.R., Le, Q.V.: Pay attention to mlps. arXiv preprint [arXiv:2105.08050](https://arxiv.org/abs/2105.08050) (2021)
28. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. arXiv preprint [arXiv:2103.14030](https://arxiv.org/abs/2103.14030) (2021)
29. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint [arXiv:1711.05101](https://arxiv.org/abs/1711.05101) (2017)
30. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint [arXiv:1706.06083](https://arxiv.org/abs/1706.06083) (2017)
31. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: ICVGIP. pp. 722–729 (2008)
32. Polyak, B.T., Juditsky, A.B.: Acceleration of stochastic approximation by averaging. SIAM journal on control and optimization **30**(4), 838–855 (1992)
33. Rao, Y., Zhao, W., Zhu, Z., Lu, J., Zhou, J.: Global filter networks for image classification. In: NeurIPS (2021)
34. Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do imagenet classifiers generalize to imagenet? In: ICML. pp. 5389–5400. PMLR (2019)
35. Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Pinto, A.S., Keysers, D., Houlsby, N.: Scaling vision with sparse mixture of experts. arXiv preprint [arXiv:2106.05974](https://arxiv.org/abs/2106.05974) (2021)

36. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. *IJCV* **115**(3), 211–252 (2015)
37. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. arXiv preprint [arXiv:1803.02155](https://arxiv.org/abs/1803.02155) (2018)
38. Srinivas, A., Lin, T.Y., Parmar, N., Shlens, J., Abbeel, P., Vaswani, A.: Bottleneck transformers for visual recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16519–16529 (2021)
39. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML. pp. 6105–6114. PMLR (2019)
40. Tay, Y., Bahri, D., Metzler, D., Juan, D.C., Zhao, Z., Zheng, C.: Synthesizer: Rethinking self-attention for transformer models. In: ICML. pp. 10183–10192. PMLR (2021)
41. Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Keysers, D., Uszkoreit, J., Lucic, M., et al.: Mlp-mixer: An all-mlp architecture for vision. arXiv preprint [arXiv:2105.01601](https://arxiv.org/abs/2105.01601) (2021)
42. Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Joulin, A., Synnaeve, G., Verbeek, J., Jégou, H.: Resmlp: Feedforward networks for image classification with data-efficient training. arXiv preprint [arXiv:2105.03404](https://arxiv.org/abs/2105.03404) (2021)
43. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. arXiv preprint [arXiv:2012.12877](https://arxiv.org/abs/2012.12877) (2020)
44. Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., Jégou, H.: Going deeper with image transformers. arXiv preprint [arXiv:2103.17239](https://arxiv.org/abs/2103.17239) (2021)
45. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS. pp. 5998–6008 (2017)
46. Wang, S., Li, B.Z., Khabsa, M., Fang, H., Ma, H.: Linformer: Self-attention with linear complexity. arXiv preprint [arXiv:2006.04768](https://arxiv.org/abs/2006.04768) (2020)
47. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions (2021)
48. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pvtv 2: Improved baselines with pyramid vision transformer. *Computational Visual Media* **8**(3), 1–10 (2022)
49. Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., Zhang, L.: Cvt: Introducing convolutions to vision transformers. arXiv preprint [arXiv:2103.15808](https://arxiv.org/abs/2103.15808) (2021)
50. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: ECCV. pp. 418–434 (2018)
51. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: Simple and efficient design for semantic segmentation with transformers. arXiv preprint [arXiv:2105.15203](https://arxiv.org/abs/2105.15203) (2021)
52. Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y., Singh, V.: Nyströmformer: A nyström-based algorithm for approximating self-attention (2021)
53. Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z., Tay, F.E., Feng, J., Yan, S.: Tokens-to-token vit: Training vision transformers from scratch on imagenet. *ICCV* (2021)
54. Zhai, X., Kolesnikov, A., Houlsby, N., Beyer, L.: Scaling vision transformers. arXiv preprint [arXiv:2106.04560](https://arxiv.org/abs/2106.04560) (2021)

55. Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. arXiv preprint [arXiv:2012.15840](https://arxiv.org/abs/2012.15840) (2020)
56. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: CVPR. pp. 633–641 (2017)