Mixing and Shifting: Exploiting Global and Local Dependencies in Vision MLPs

Huangjie Zheng[1,2], Pengcheng He[2], Weizhu Chen[2], Mingyuan Zhou[1]

huangjie.zheng@utexas.edu, penhe@microsoft.com,
wzchen@microsoft.com, mingyuan.zhou@mccombs.utexas.edu

The University of Texas at Austin[1]   Microsoft Azure AI[2]

Abstract

Token-mixing multi-layer perceptron (MLP) models have shown competitive performance in computer vision tasks with a simple architecture and relatively small computational cost. Their success in maintaining computation efficiency is mainly attributed to avoiding the use of self-attention that is often computationally heavy, yet this is at the expense of not being able to mix tokens both globally and locally. In this paper, to exploit both global and local dependencies without self-attention, we present Mix-Shift-MLP (MS-MLP) which makes the size of the local receptive field used for mixing increase in respect to the amount of spatial shifting. In addition to conventional mixing and shifting techniques, MS-MLP mixes both neighboring and distant tokens from fine- to coarse-grained levels and then gathers them via a shifting operation. This directly contributes to the interactions between global and local tokens. Being simple to implement, MS-MLP achieves competitive performance in multiple vision benchmarks. For example, an MS-MLP with 85 million parameters achieves 83.8% top-1 classification accuracy on ImageNet-1K. Moreover, by combining MS-MLP with state-of-the-art Vision Transformers such as the Swin Transformer, we show MS-MLP achieves further improvements on three different model scales, *e.g.*, by 0.5% on ImageNet-1K classification with Swin-B. The code is available at: https://github.com/JegZheng/MS-MLP.

1 Introduction

Showing promise in modeling visual dependencies, Vision Transformers (ViTs) have advanced the state of the art (SoTA) of many different visual tasks (Dosovitskiy et al., 2020; Touvron et al., 2020; Liu et al., 2021b). However, the self-attention module (Vaswani et al., 2017), which is key to the ViT success in capturing long-range visual dependencies, involves a computationally intensive operation that compares pairwise similarity between tokens. Inspired by self-attention, but without its heavy computation, several works show that building models solely on multi-layer perceptrons (MLPs) can achieve surprisingly promising results on ImageNet (Deng et al., 2009) classification with both spatial- and channel-wise token mixing (Tolstikhin et al., 2021; Touvron et al., 2021; Liu et al., 2021a). These MLP-based models are efficient in token mixing to aggregate the spatial information and model visual feature dependencies, achieving results competitive to previous models on several representative computer vision tasks, such as image classification, object detection, and semantic segmentation.

Extensive studies of MLPs can be categorized into two mainstream directions depending on whether capturing global or local visual dependencies. Inspired by ViTs, global-mixing MLP-based methods such as MLP-Mixer (Tolstikhin et al., 2021) and ResMLP (Touvron et al., 2021) achieve the global reception field with the communication between patch tokens through spatial-wise projections. In this direction, researchers explore to effectively handle all tokens with various techniques, such as gating, routing, and Fourier transforms (Liu et al., 2021a; Lou et al., 2021; Rao et al., 2021; Tang et al., 2021a,b). Apart from MLPs that explore the modeling of global visual dependencies, a large number of studies have also achieved progress in using MLP-based architectures to model local visual dependencies, as done in the classical convolution paradigm (LeCun et al., 1995). Different from global-mixing architectures, local-mixing MLPs sample nearby tokens for interactions. In this direction several studies achieve effective token sampling by exploiting spatial shifting, permutation,

and pseudo-kernel mixing (Yu et al., 2021; Hou et al., 2021; Mao et al., 2021; Lian et al., 2021; Chen et al., 2021b; Guo et al., 2021), *etc.*

Despite the success from both perspectives, MLPs still avoid the self-attention at the expense of not being able to mix tokens as flexibly and efficiently as self-attention. The global-mixing is less flexible in identifying the importance among all tokens, while local-mixing is not able to capture long-range dependencies. In this paper, we investigate whether MLPs can effectively capture both short- and long-range dependencies to further improve performance. Intuitively, the visual dependencies between neighboring regions are usually more significant and need more attention, while those far away are still not trivial at a glance. Therefore, we propose to mix tokens from fine- to coarse-levels, where we perform fine-grained mixing in neighboring regions to achieve token interactions locally, while coarse-grained mixing for distant tokens to capture long-range dependencies. Specifically, we propose a multi-scale regional mixing, where the size of the regional receptive field used for mixing is proportional to the relative distance with respect to the query token, and these multi-scale regions are aggregated with a shifting operation. We plug in such multi-scale regional mixing into MLP architecture as Mix-Shift-MLP (MS-MLP). We evaluate the performance of MS-MLP via a comprehensive empirical study on a series of representative computer vision tasks, including image classification, object detection, and segmentation. According to the results, given a similar model complexity, our MS-MLP consistently outperforms SoTA MLPs across various settings, *e.g.*, MS-MLP-B achieves 83.8% in ImageNet-1K classification, which is on par with Focal-Attention-B (Yang et al., 2021) with superior throughputs. In addition, we plug our MS-MLP module into SoTA Transformer models, which further improves both performance and efficiency. Notably, with MS-MLP, Swin Transformers (Liu et al., 2021b) and Focal Transformers (Yang et al., 2021) respectively get improved on average by 0.5-0.6% / 0.2-0.6% in image classification and 0.2-0.5% / 0.1-0.3% in object detection and segmentation.

2 Related works

Global token-mixing MLPs: Global token-mixing MLPs are first proposed as self-attention-free alternatives to Transformer architectures (Tolstikhin et al., 2021; Melas-Kyriazi, 2021; Touvron et al., 2021). MLP-Mixer (Tolstikhin et al., 2021) replaces the self-attention layer of ViT with a spatial-wise MLP projection of tokens, achieving results that are competitive with ViT. gMLP (Liu et al., 2021a), consisting of an MLP-based module with multiplicative gating, provides competitive results in both vision and natural language processing (NLP) tasks. Vision Permutator (Hou et al., 2021) focuses on global mixing along both the vertical and horizontal axes. Raft-MLP (Tatsunami & Taki, 2021) employs a hierarchical and serialized structure which continuously improves accuracy. Similar to the parameterization of query and key pairs in ViTs, Wave-MLP (Tang et al., 2021b) reweighs the importance of tokens with the amplitude and phase modules parameterized by two MLP projections.

Local token-mixing MLPs: Local token-mixing MLPs focus more on the token interactions at local regions and hence share more similarities with convolutional neural networks (CNNs) than with Transformers. They have also been proved to achieve good performance on computer vision tasks. For example, a spatially shifted MLP (S^2-MLP) (Yu et al., 2021; 2022) takes spatial shifts in four directions and mixes them in a channel-wise manner to gather information from neighboring tokens. Similar to S^2-MLP, an axial-shifted MLP (AS-MLP) (Lian et al., 2021) changes the spatial shifts in both the horizontal and vertical axes to gather local region information. CycleMLP (Chen et al., 2021b) takes pseudo-kernels and sample tokens from different spatial locations for mixing. ConvMLP (Li et al., 2021) incorporates convolution layers and a pyramid structure to achieve local token mixing. Hire-MLP (Guo et al., 2021) rearranges tokens across local regions to gain performance and computational efficiency.

ViTs and CNNs: Transformers (Vaswani et al., 2017), originated from the NLP area, have recently been applied to visual tasks. In ViTs, the input is processed as patch tokens and then self-attention is used to aggregate spatial information globally (Dosovitskiy et al., 2020). Touvron et al. (2020) explore how to train ViTs efficiently with a distillation strategy. The use of CNNs has a long history for visual tasks, with extensive works conducted on improving the design to aggregate the features from local convolution and enlarge the receptive field
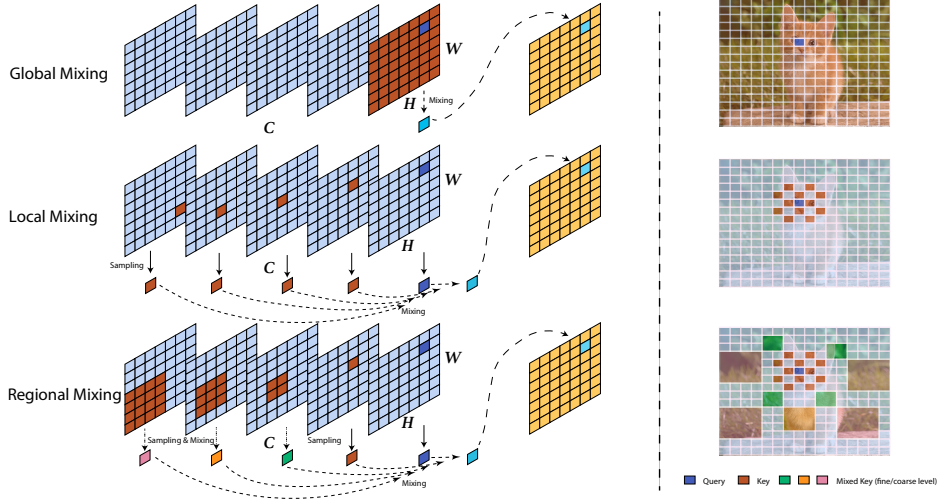
Figure 1: Illustrative comparison of typical operations in MLPs (best viewed in color). (**Left**) Different mixing strategies in a feature map: global mixing communicates within all the tokens to gather global spatial information; local mixing mechanism samples neighboring tokens to model local spatial dependency; regional mixing interacts tokens within regions of different scale that is proportional to the relative distance with regards to the query token. (**Right**) Corresponding visualization in view of attention mechanism: the global mixing captures dependency among all tokens, weighted by projection layer; the sparse mixing models local dependency with nearby tokens; our multi-scale regional mixing makes use of fine-grained level information for nearby tokens and coarse-level information for distant tokens.

(LeCun et al., 1998; Krizhevsky et al., 2012; Simonyan & Zisserman, 2014; Szegedy et al., 2015; He et al., 2016b). Recent works are proposed to marry the advantages of both global attention in ViTs and local attention in CNNs. PVT (Wang et al., 2021a) exploits a pyramid structure for gathering spatial information on dense prediction tasks such as object detection. TNT (Han et al., 2021) takes small Transformer blocks to capture local information. Swin Transformer (Liu et al., 2021b) proposes shifted window attention in order to aggregate spatial information from local regions. Focal-Transformer (Yang et al., 2021) extends the use of local shifted windows to different scales to efficiently capture both short- and long-range visual dependencies. Incorporating recent findings in Transformers, ConvNeXT (Liu et al., 2022) enlarges the receptive fields with larger kernels to capture global dependencies. Related to these recent progress, our work aims to improve MLP-based architectures to efficiently capture both short-range and long-range visual dependencies.

3   Method

Conventional MLPs are built upon a stacked architecture of multiple token-mixing blocks, where each token-mixing block consists of two sub-blocks, *i.e.*, a spatial-mixing module and a channel-mixing MLP to aggregate spatial and channel information, respectively. Given an input feature with height H, width W, and channel C, expressed as $X \in \mathbb{R}^{H \times W \times C}$, the token-mixing block is formulated as:

$$
\begin{aligned}
Z &= f_{\text{Spatial-Mixing}}(h_{\text{Spatial}}(X)) + X, \\
O &= f_{\text{Channel-Mixing}}(h_{\text{Channel}}(Z)) + Z,
\end{aligned}
\tag{1}
$$

where Z and O denote the intermediate feature and output feature of the block, respectively, and h denotes a normalization technique, such as batch normalization or layer normalization (Ioffe & Szegedy, 2015; Ba et al., 2016). The channel-mixing function $f_{\text{Channel-Mixing}}$ is usually parameterized with two MLPs, where the hidden dimension of the intermediate output is four times wider than the input dimension. Keeping this setting the same as previous ViTs and MLPs, we focus on investigating the spatial-mixing function $f_{\text{Spatial-Mixing}}$ in what follows.
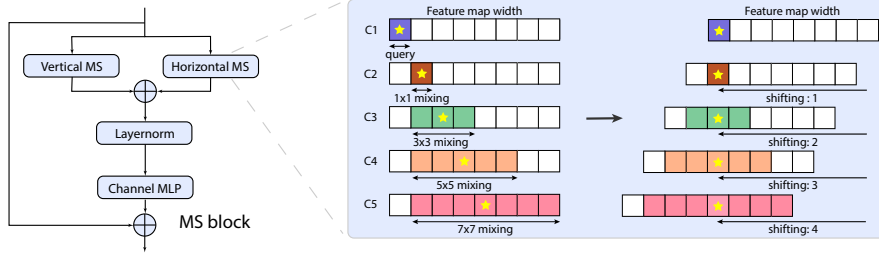
Figure 2: The plate notation of an MS-block (**Left**) and the illustration of the mixing-shifting operation in the horizontal direction of the feature map (**Right**). We first split the feature map into several groups (five groups here) along the channel dimension, with the first group regarded as the source of query tokens. In the other groups, as the centers of the attended regions (marked with yellow stars) become more and more distant, we gradually increase the mixing spatial range from 1×1 to 7×7. After the mixing operation, we shift the split channel groups to align their mixed center tokens with the query and then continue the channel-wise mixing with a channel MLP.

3.1 Multi-scale regional token-mixing

We first provide an illustration of the global and local token-mixing methods, as well as the proposed mixing that interacts tokens within regions on different scales. As shown in Fig. 1, given the query token patch (marked in cobalt blue), global mixing mixes all the tokens in the same channel together to get the output token (marked in cyan); local mixing first samples the tokens in nearby locations from all channels, then mixes them as the output. Correspondingly, in the right panel, the global mixing interacts with all locations and pays attention to all the other tokens, while the local mixing interacts with neighboring tokens of the query and pays attention to the nearby locations. Although global mixing gives interactions with all locations, it is more expensive in computation compared with local mixing, and none of the mixing schemes have the capability to give prioritization to the query. The receptive field of local mixing is also limited as the query only interacts with its neighbors. Here we consider a regional mixing with different region sizes, $i.e.$, a fine-grained mixing in nearby locations and a coarse-grained mixing in distant locations for a global view. As shown in Fig. 1, we propose to sample in the closest regions at the finest granularity of the given query. As it samples more distant locations, we gradually increase the granularity and mix the tokens in that region to produce a coarse-grained token. Finally, similar to local mixing, we mix the tokens from channels ranging from the fine-grained tokens to coarse-grained tokens to produce the output token. As a result, region mixing has the ability to capture both global and local dependencies efficiently, where it covers many more regions in the feature map than local mixing and emphasizes more important local regions as well as capturing the distant regions with a coarse granularity when compared to global mixing.

Mathematically, with $\boldsymbol{x}_c \in \mathbb{R}^{H \times W}$ be the c-th channel in X, the global mixing can be formulated as the function $f_{\text{global}} : \mathbb{R}^{H \times W} \to \mathbb{R}^{H \times W}$ such that $\boldsymbol{o}_c = f_{\text{global}}(\boldsymbol{x}_c)$, where f_{global} is also parameterized with an MLP along the flattened $H \times W$ dimension (Tolstikhin et al., 2021). In this way, token $x_{i,j,c}$ gathers spatial information from tokens $\{x_{i',j'}\}_{i' \neq i, j' \neq j}$ located elsewhere. Supposing $\mathcal{N}_{ij} = \mathbb{R}^{H' \times W' \times C} \subseteq \mathbb{R}^{H \times W \times C}$ denotes the feature map neighboring location of center coordinate (i, j), local mixing involves a spatial-wise sampling function g_{sampling} to get $\mathcal{N}_{ij}$. Then $f_{\text{local}}(\cdot; i, j) : \mathcal{N}_{ij}(\cdot) \to \mathcal{N}_{ij}(\cdot)$ mixes the sampled token subset $\{x_{i',j'}\}_{(i',j') \in \mathcal{N}_{ij}}$ such that $\boldsymbol{o}_{i,j} = f_{\text{local}}(\{x_{i',j'}\}_{(i',j') \in \mathcal{N}_{ij}}; i, j)$, where f_{local} is usually a spatial arrangement operation like shifting or concatenation followed by a channel-wise projection (Yu et al., 2021; Lian et al., 2021).

The proposed regional mixing combines the previous mixing strategies. We regard the sampled tokens $\{x_{i',j'}\}_{(i',j') \in \mathcal{N}_{ij}}$ as the center tokens of regions with different sizes and deploy a mixing function in each region as f_{region}^r, where $f_{\text{region}}^r : \mathbb{R}^{H_r \times W_r} \to \mathbb{R}^{H_r \times W_r}$, to let the center token represent information in that region, with r denoting the size of the region. Below we describe this technique in detail.

3.2 Regional token-mixing via mixing and shifting

The regional token-mixing can be achieved with a composition of mixing and shifting, where they become dependent compared to conventional methods. Let's first define three terms to describe the proposed mixing-shifting clearly:

4

1) Shifting size: We denote S as the shifting size of the feature map. We equally split the input feature along the channel into S groups as $X = \boldsymbol{x}_{C1} \cup ... \cup \boldsymbol{x}_{CS}$ and $\forall n \neq m, \boldsymbol{x}_{Cn} \cap \boldsymbol{x}_{Cm} = \emptyset$. We assume the tokens in the first group $\boldsymbol{x}_{C1}$ as queries and shift all the other groups.

2) Relative distance: We denote d_n as the relative distance between the query group and each shifted groups X_{Cn}. We shift each feature group to reach a relative distance d_n with regards to the query group.

3) Mixing region size: We denote r_n as the mixing region size in X_{Cn}, where a larger r_n indicates a coarser granularity. Each group will take care of an $r_n \times r_n$ grid region.

Each time, like with local-mixing, we first target a center token $x_{i+d_n,j}$ or $x_{i,j+d_n}$ in each feature group X_{Cn} and mix the tokens in an $r_n \times r_n$ grid region around the center token. Then, we shift the channel groups horizontally or vertically to both align the center tokens and get the features before channel-wise mixing:

$$\begin{aligned}
\boldsymbol{y}_{i+d_n,j,C_n} &= f_{\text{region}}^{r_n}(\boldsymbol{x}_{i+d_n-r_n:i+d_n+r_n,j-r_n:j+r_n}), \\
\boldsymbol{o}_{i,j} &= f_{\text{local}}(\{\boldsymbol{y}_{i+d_n,j,C_n}\}_{n=1:S}),
\end{aligned} \tag{2}$$

where $f_{\text{region}}^{r_n}$ is instantiated with a depth-wise convolution with kernel size $r_n \times r_n$ and f_{local} is a shifting operation.

Fig. 2 shows the architecture of a mix-shifting block and an illustrative example of the proposed mix-shifting. In this example, we set $S = 5$, $d_n = n - 1$, and $r_n = 2d_n - 1$. In this horizontal mix-shifting, the feature map is divided into five groups, with group C2-C4 gradually increasing both the region size and the relative distance between the center token and the query (cobalt blue token in C1). After the global mixing in each region, each group is shifted back according to the relative distance and all center tokens are aligned for the channel mixing.

3.3 Complexity analysis

In this subsection, we compare the computational complexity of typical ways to interact within tokens spatially, including the multi-head self-attention (MSA) in ViTs (Dosovitskiy et al., 2020), window multi-head self-attention (W-MSA) in Swin Transformer (Liu et al., 2021b), focal multi-head self-attention (F-MSA) in Focal-Attention Transformer (Yang et al., 2021), global-mixing in MLP-Mixer (Tolstikhin et al., 2021), local-mixing in AS-MLP (Lian et al., 2021), and regional-mixing in our MS-MLP. We assume the channel-MLPs are the same, and that the shifting size and mixing region size in MS-MLP are the same as the focal level and focal region level in F-MSA. Denoting the input dimension as $H \times W \times C$ and the window size of W-MSA and F-MSA as M, the complexities of the above methods are shown as follows:

Table 1: Computation complexity comparison of different token interaction methods.

Method	MSA	W-MSA	F-MSA
Complexity	$\mathcal{O}(2(HW)^2C)$	$\mathcal{O}(2M^2HWC)$	$\mathcal{O}((S + \sum_{n=1}^{S}(r_n)^2)MHWC)$

Method	GM	AS	MS
Complexity	$\mathcal{O}((HW)^2C)$	$\mathcal{O}(S)$	$\mathcal{O}(\sum_{n=1}^{S}(r_n)^2)$

From Table 1, we can observe that compared with Transformers, MLPs largely reduce the computation complexity in dealing with token dependencies. In the comparison within MLPs, axial-shift and our proposed mix-shifting technique possess much lower complexity than global-mixing in MLP-Mixer. Note that here we make the shifting size S and mixing region size r_n the same as the focal level and focal region size in F-MSA, respectively, for better comparison, meaning each channel group has a different r_n. In practice, r_n is not necessarily dependent on S, *i.e.*, we can also set a focal level that is smaller than S and the complexity is still $\mathcal{O}(\sum_{n=1}^{S}(r_n)^2)$. If we set the focal level to 1 and fix $r_n = 1$, this special case of MS-MLP will reduce to an AS-MLP (Lian et al., 2021).

3.4 Model architecture overview

In this part we present an overview of the MS-MLP architecture. Following convention, we consider MS-MLP with Tiny, Small, and Base, corresponding to three different network

Table 2: MS-MLP model architectures with different configurations. Following the convention, we introduce three different configurations—Tiny, Small, and Base—for different model capacities.

	Input resolution	Layer Name	MS-MLP-Tiny	MS-MLP-Small	MS-MLP-Base
stage 1	56×56	Patch Embedding	$p_1 = 4; c_1 = 96$	$p_1 = 4; c_1 = 96$	$p_1 = 4; c_1 = 128$
		MS-block	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,1,3,5,7] \end{bmatrix} \times 3$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,1,3,5,7] \end{bmatrix} \times 3$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,1,3,5,7] \end{bmatrix} \times 3$
stage 2	28×28	Patch Embedding	$p_2 = 2; c_2 = 192$	$p_2 = 2; c_2 = 192$	$p_2 = 2; c_2 = 256$
		MS-block	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,3,3,5,7] \end{bmatrix} \times 3$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,3,3,5,7] \end{bmatrix} \times 3$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,3,3,5,7] \end{bmatrix} \times 3$
stage 3	14×14	Patch Embedding	$p_3 = 2; c_3 = 384$	$p_3 = 2; c_3 = 384$	$p_3 = 2; c_3 = 512$
		MS-block	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,5,5,5,7] \end{bmatrix} \times 9$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,5,5,5,7] \end{bmatrix} \times 27$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,5,5,5,7] \end{bmatrix} \times 27$
stage 4	7×7	Patch Embedding	$p_4 = 2; c_4 = 768$	$p_4 = 2; c_4 = 768$	$p_4 = 2; c_4 = 1024$
		MS-block	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,7,7,7,7] \end{bmatrix} \times 3$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,7,7,7,7] \end{bmatrix} \times 3$	$\begin{bmatrix} S = 5 \\ d_{1:S} = [0,1,2,3,4] \\ r_{1:S} = [1,7,7,7,7] \end{bmatrix} \times 3$

configurations with different model capacities. Here we follow previous works to adopt a pyramid-like architecture (Wang et al., 2021a; Wu et al., 2021a; Liu et al., 2021b; Yang et al., 2021; Guo et al., 2021) for our MS-MLP. Our model takes 224×224 pixel images as inputs and first splits the input image into patches (tokens) by a patch embedding. Then the token features go through a four-stage architecture. As the features go deeper to another stage, the number of tokens is reduced by a patch-embedding layer with ratio p_i and output channels are simultaneously increased by this ratio. The spatial reduction ratio p_i for these four stages is set as $[4, 2, 2, 2]$. An overview of three configurations is shown in Table 2. We keep $S = 5$ for all MS-blocks in the architecture for simplicity in presentation, though we believe there should exist a better configuration and we leave it to future exploration. Since the feature map resolution becomes smaller and smaller, we gradually decrease the fine-grained level region mixing. For example, at stage 4 we keep only the channel group with a 7×7 region mixing, since in the last stage the patch resolution is sufficiently small. We find the architecture in Table 2 performs the best; we also explored a simpler configuration in our ablations, where all MS-blocks keep the same configuration ($d_n = n - 1$ and $r_n = 2d_n - 1$) and the performance differed very slightly. Moreover, to match the number of parameters and floating-point operations per second (FLOPs) of most existing models, we increase the number of MS-blocks in each stage, but keep the ratio of 1:1:3:1.

3.5 Improving Transformers in low-level stages

How to combine the effectiveness of MLPs in computation with the flexibility of self-attention in Transformers is an interesting topic in computer vision research. In low-level stages, the model needs to process high-resolution inputs. Compared to W-MSA and local-mixing MLPs, MS-MLP has a much larger receptive field coverage; compared to MSA and global-mixing, MS-MLP covers just as many regions, but is more efficient with high-resolution inputs according to our analysis in Section 3.3. We empirically find MS-MLP can be combined with Transformers to boost performance, either by replacing the first-stage architecture or by being added as an additional stage to deal with finer-grained input (*e.g.*, input with patch size 2). In our experiments, we show corresponding improvements over both the Swin Transformer (Liu et al., 2021b) and Focal-Attention Transformer (Yang et al., 2021).

4 Experiments

In this section, we investigate the effectiveness of the MS-MLP architectures using experiments on multiple vision tasks. We first use image classification on ImageNet-1K (Deng et al., 2009) to compare MS-MLP with previous state-of-the-art MLPs. Next, we show the performance of combining MS-MLP with the Swin Transformer (Liu et al., 2021b) and with the Focal-Attention Transformer (Yang et al., 2021), with a comparison to all SoTA methods on this task. Furthermore, we compare MS-MLP with existing alternatives using object detection and semantic segmentation on COCO-2017 (Lin et al., 2014). Finally, we present ablation

Table 3: Comparison of the proposed MS-MLP architecture with existing vision MLP models on ImageNet. All models are trained and evaluated on 224×224 resolution, grouped according to the model size. Baseline results are quoted from the original papers.

Model	Mixing.	Params.	FLOPs	Throughput (image / s)	Top-1 acc. (%)
gMLP-Ti (Liu et al., 2021a)	global	6M	1.4G	-	72.3
ResMLP-S12 (Touvron et al., 2021)	global	15M	3.0G	1415.1	76.6
CycleMLP-B1 (Chen et al., 2021b)	local	15M	2.1G	1038.4	78.9
Wave-MLP-T (Tang et al., 2021b)	global	17M	2.4G	1208	**80.6**
Hire-MLP-T (Guo et al., 2021)	local	18M	2.1G	1561.7	79.7
gMLP-S (Liu et al., 2021a)	global	20M	4.5G	-	79.6
ViP-Small/7 (Hou et al., 2021)	local	25M	-	719.0	81.5
AS-MLP-T (Lian et al., 2021)	local	28M	4.4G	863.6	81.3
CycleMLP-B2 (Chen et al., 2021b)	local	27M	3.9G	640.6	81.6
Wave-MLP-S (Tang et al., 2021b)	global	30M	4.5G	720	**82.6**
Hire-MLP-S (Guo et al., 2021)	local	33M	4.2G	807.6	82.1
MS-MLP-T (ours)	regional	28M	4.9G	792.0	82.1
Mixer-B/16 (Tolstikhin et al., 2021)	global	59M	12.7G	-	76.4
S^2-MLP-deep (Yu et al., 2021)	local	51M	10.5G	-	80.7
ViP-Medium/7 (Hou et al., 2021)	local	55M	-	418.0	82.7
CycleMLP-B4 (Chen et al., 2021b)	local	52M	10.1G	320.8	83.0
AS-MLP-S (Lian et al., 2021)	local	50M	8.5G	478.4	83.1
Wave-MLP-M (Tang et al., 2021b)	global	44M	7.9G	413	**83.4**
Hire-MLP-B (Guo et al., 2021)	local	58M	8.1G	440.6	83.2
MS-MLP-S (ours)	regional	50M	9.0G	483.8	**83.4**
ResMLP-B24 (Touvron et al., 2021)	global	116M	23.0G	231.3	81.0
S^2-MLP-wide (Yu et al., 2021)	local	71M	14.0G	-	80.0
CycleMLP-B5 (Chen et al., 2021b)	local	76M	12.3G	246.9	83.2
gMLP-B (Liu et al., 2021a)	global	73M	15.8G	-	81.6
ViP-Large/7 (Hou et al., 2021)	local	88M	-	298.0	83.2
AS-MLP-B (Lian et al., 2021)	local	88M	15.2G	312.4	83.3
Wave-MLP-B (Tang et al., 2021b)	global	63M	10.2G	341	83.6
Hire-MLP-L (Guo et al., 2021)	local	96M	13.4G	290.1	**83.8**
MS-MLP-B (ours)	regional	88M	16.1G	366.5	**83.8**

studies on the effects of regional mixing. Appendix A provides detailed experimental settings.

4.1 Image classification on ImageNet-1K

On ImageNet-1K (Deng et al., 2009), for a fair comparison, we follow the commonly used training recipes in Dosovitskiy et al. (2020) and Wang et al. (2021a). All models are trained for 300 epochs with a batch size of 1,024. The initial learning rate is set to 10^{-3} with 20 epochs of linear warm-up starting from 10^{-5}. For optimization, we use AdamW (Loshchilov & Hutter, 2017) as the optimizer with a cosine learning rate scheduler. The weight decay is set to 0.05 and the maximal gradient norm is clipped to 5.0. We use the same set of data augmentation and regularization strategies as in Touvron et al. (2020), including Rand-Augment (Cubuk et al., 2020), MixUp (Zhang et al., 2017), CutMix (Yun et al., 2019), Label Smoothing (Szegedy et al., 2016), Random Erasing (Zhong et al., 2020), and DropPath (Huang et al., 2016). The stochastic depth drop rates are set to 0.2, 0.3, and 0.5 for our Tiny, Small, and Base models, respectively. During training, we crop images randomly to 224×224, while a center crop is used during evaluation on the validation set. All models are trained on a node with eight NVIDIA Tesla V100 GPUs, based on which we report the experimental results with top-1 accuracy, number of parameters, FLOPs, and throughput.

Main results comparing with MLPs: We compare the proposed MS-MLP with previous MLP-based models on ImageNet, as shown in Table 3. MS-MLP with region mixing consistently achieves competitive results with better computation efficiency. For example, compared with AS-MLP (Lian et al., 2021) and CycleMLP (Chen et al., 2021b), MS-MLP can perform significantly better with comparable parameter scales, FLOPs, and throughput. When compared with recently proposed Wave-MLP (Tang et al., 2021b) and Hire-MLP (Guo et al., 2021), MS-MLP obtains a better throughput and similar classification accuracy. In particular, scale up to Base configuration, MS-MLP achieves the best results (83.8%) with a

Table 4: Comparison of MS-MLP architecture with representative SoTA models on ImageNet-1K with a resolution of 224×224.

Model	Family	Params.	FLOPs	Throughput (images / s)	Top-1 acc. (%)
ResNet18 (He et al., 2016a)	CNN	12M	1.8G	-	69.8
ResNet50 (He et al., 2016a)	CNN	26M	4.1G	-	78.5
ResNet101 (He et al., 2016a)	CNN	45M	7.9G	-	79.8
ConvNeXt-T (Liu et al., 2022)	CNN	29M	4.5G	775	82.1
ConvNeXt-S (Liu et al., 2022)	CNN	50M	8.7G	447	83.1
ConvNeXt-B (Liu et al., 2022)	CNN	89M	15.4G	292	**83.8**
Swin-T (Liu et al., 2021b)	Trans	29M	4.5G	755	81.3
Swin-S (Liu et al., 2021b)	Trans	50M	8.7G	437	83.0
Swin-B (Liu et al., 2021b)	Trans	88M	15.4G	278	83.3
Focal-Attention-T (Yang et al., 2021)	Trans	29M	4.9G	319	82.2
Focal-Attention-S (Yang et al., 2021)	Trans	52M	9.4G	192	83.5
Focal-Attention-B (Yang et al., 2021)	Trans	90M	16.4G	138	**83.8**
MS-MLP-T (ours)	MLP	28M	4.9G	792	82.1
MS-MLP-S (ours)	MLP	50M	9.0G	484	83.4
MS-MLP-B (ours)	MLP	88M	16.1G	366	**83.8**

Table 5: Comparison of Swin and Focal-Attention transformer w/o MS-MLP on ImageNet-1K.

Model	Family	Params.	FLOPs	Throughput (images / s)	Top-1 acc. (%)
Swin-T (Liu et al., 2021b)	Trans	29M	4.5G	755	81.3
Swin-S (Liu et al., 2021b)	Trans	50M	8.7G	437	83.0
Swin-B (Liu et al., 2021b)	Trans	88M	15.4G	278	83.3
MS-MLP + Swin-T (ours)	MLP + T	29M	4.5G	779	**81.9**
MS-MLP + Swin-S (ours)	MLP + T	50M	8.7G	464	**83.5**
MS-MLP + Swin-B (ours)	MLP + T	88M	15.4G	279	**83.8**
Focal-Attention-T (Yang et al., 2021)	Trans	29M	4.9G	319	82.2
Focal-Attention-S (Yang et al., 2021)	Trans	52M	9.4G	192	83.5
Focal-Attention-B (Yang et al., 2021)	Trans	90M	16.4G	138	83.8
MS-MLP + Focal-Attention-T (ours)	MLP + T	29M	5.6G	451	**82.8**
MS-MLP + Focal-Attention-S (ours)	MLP + T	52M	10.1G	297	**83.9**
MS-MLP + Focal-Attention-B (ours)	MLP + T	90M	17.6G	207	**84.0**

throughput (366.5 images/sec) surpassing all the other models with a comparable number of parameters.

Comparing with SoTAs: Besides MLPs, we compare MS-MLP with representative CNN-based and Transformer-based SoTA models. The input image resolution is set as 224×224. As in Table 4, compared with SoTAs, MS-MLP achieves competitive performance with better efficiency. For example, MS-MLP-B achieves 83.8% top-1 accuracy, which is superior to Swin-B with 83.3% accuracy. The computational efficiency of MS-MLP is significantly better than Transformers, and slightly surpasses the CNN architectures like ConvNeXt (Liu et al., 2022).

Results with Transformers: To effectively combine the strengths of both MS-MLP and Transformers, we let MS-MLP and Transformers represent low-level and high-level stages, respectively. We found this design can largely boost model efficiency. In the experiments, we replace the first stage of the Swin Transformer (Liu et al., 2021b) and the Focal-Attention Transformer (Yang et al., 2021) with MS-blocks. Moreover, we add a stage zero that consists of two MS-blocks with $p_0 = 2, c_0 = c_1/2$ ahead of stage one. This novel configuration produces a modified model having similar parameter sizes, FLOPs, and throughputs as the original models, but a consistently better performance in terms of accuracy.

We compare the proposed MS-MLP+Transformer architecture with the original architectures on ImageNet-1K and summarize the results in Table 5. Compared with the Swin Transformer and Focal-Attention Transformer, the MS-MLP+Transformer architectures achieve both a higher accuracy and a higher throughput with a similar number of parameters and FLOPs. For example, with 88M parameters and 15.4G FLOPs, MS-MLP+Swin-B

Table 6: Object detection and instance segmentation results on COCO val2017. We compare MS-MLP with other backbones based on RetinaNet and Mask R-CNN frameworks. All models are trained on the "1x" schedule.

Backbone	RetinaNet 1×					Mask R-CNN 1×						
	Param / FLOPs	AP	AP_S	AP_M	AP_L	Param / FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
CycleMLP-B2 [Chen et al., 2021b]	36.5M / 230G	40.9	23.4	44.7	53.4	46.5M / 249G	41.7	63.6	45.8	38.2	60.4	41.0
Wave-MLP-S [Tang et al., 2021b]	37.1M / 231G	**43.4**	**26.6**	47.1	57.1	47.0M / 250G	44.0	65.8	48.2	40.0	**63.1**	42.9
Hire-MLP-S [Guo et al., 2021]	42.8M / 237G	41.7	25.3	45.4	54.6	52.7M / 256G	42.8	65.0	46.7	39.3	62.0	42.1
MS-MLP-T (ours)	39.6M / 265G	42.7	26.0	**47.3**	**59.7**	49.8M / 269G	**44.4**	**67.8**	**47.8**	**40.4**	61.3	**44.2**
CycleMLP-B3 [Chen et al., 2021b]	48.1M / 291G	42.5	25.2	45.5	56.2	58.0M / 309G	43.4	65.0	47.7	39.5	62.0	42.4
CycleMLP-B4 [Chen et al., 2021b]	61.5M / 356G	43.2	26.6	46.5	57.4	71.5M / 375G	44.1	65.7	48.1	40.2	62.7	43.5
Wave-MLP-M [Tang et al., 2021b]	49.4M / 291G	44.8	28.0	48.2	59.1	59.6M / 311G	45.3	67.0	49.5	41.0	64.1	44.1
Hire-MLP-B [Guo et al., 2021]	68.0M / 316G	44.3	28.0	48.4	58.0	77.8M / 334G	45.2	66.9	49.3	41.0	64.0	44.2
MS-MLP-S (ours)	61.2M / 360G	**45.0**	**28.2**	**48.9**	**60.4**	70.9M / 372G	**47.1**	**71.1**	**51.6**	**41.9**	64.1	**45.1**
CycleMLP-B5 [Chen et al., 2021b]	85.9M / 402G	42.7	24.1	46.3	57.4	95.3M / 421G	44.1	65.5	48.4	40.1	62.8	43.0
Wave-MLP-B [Tang et al., 2021b]	66.1M / 334G	44.2	27.1	47.8	58.9	75.1M / 353G	45.7	67.5	50.1	27.8	49.2	59.7
Hire-MLP-L [Guo et al., 2021]	105.8M / 424G	44.9	**28.9**	48.9	57.5	115.2M / 443G	45.9	67.2	50.4	41.7	**64.7**	45.3
MS-MLP-B (ours)	97.3M / 544G	**45.7**	27.8	**49.2**	**59.7**	107.5M / 557G	**46.4**	67.2	50.7	**42.4**	63.6	**46.4**

Table 7: Results of semantic segmentation on the ADE20K validation set. FLOPs are calculated with an input size of 2048×512.

Backbone	Param	FLOPs	SS mIoU	MS mIoU
Swin-T [Liu et al., 2021b]	60M	945G	44.5	46.1
Focal-T [Yang et al., 2021]	62M	998G	45.8	47.0
AS-MLP-T [Lian et al., 2021]	60M	937G	-	46.5
Hire-MLP-S [Guo et al., 2021]	63M	930G	**46.1**	**47.1**
MS-MLP-T (ours)	61M	939G	46.0	46.8
ResNet-101 [He et al., 2016b]	86M	1029G	43.8	44.9
Swin-S [Liu et al., 2021b]	81M	1038G	47.6	49.5
Focal-S [Yang et al., 2021]	85M	1130G	48.0	**50.0**
AS-MLP-S [Lian et al., 2021]	81M	1024G	-	49.2
Hire-MLP-B [Guo et al., 2021]	88M	1011G	48.3	49.6
MS-MLP-S (ours)	82M	1028G	**48.7**	49.6
Swin-B [Liu et al., 2021b]	121M	1188G	48.1	49.7
Focal-B [Yang et al., 2021]	126M	1354G	49.0	**50.5**
AS-MLP-B [Lian et al., 2021]	121M	1166G	-	49.5
Hire-MLP-L [Guo et al., 2021]	127M	1125G	48.8	49.9
MS-MLP-B (ours)	122M	1172G	**49.1**	49.9

achieves an 83.8 top-1 accuracy, surpassing Swin-B with 83.3% accuracy. With very similar 90M parameters and slightly higher FLOPs, MS-MLP+Focal-Attention improves both the top-1 classification accuracy and the throughputs, *e.g.* Focal-B is improved to 84.0%, which sets a new SoTA accuracy with a similar model size. The superiority of MS-MLP+Transformer clearly implies that the MS-MLP architecture has a better efficiency in the token aggregation process. This positive effect becomes more pronounced with larger token numbers, since the proposed mixing and shifting operations can exploit both global and local dependencies adequately.

4.2 Object detection and instance segmentation

Results on COCO-2017: We first conduct the object detection and instance segmentation experiments on COCO-2017 (Lin et al., 2014). Following previous works (Wang et al., 2021a; Liu et al., 2021b), we use the pretrained models as backbones and plug into RetinaNet (Lin et al., 2017), Mask R-CNN (He et al., 2017), and Cascade Mask R-CNN (Cai & Vasconcelos, 2018) in mmdetection (Chen et al., 2019). We adopt the single-scale and multi-scale training for the "1x" and "3x" schedules, respectively.

The results of object detection and instance segmentation with different frameworks and training schedules are reported in Table 6 and Table 11 (Appendix), respectively. As shown in Table 6, using MS-MLP as the backbone, RetinaNet and Mask R-CNN surpass most of the MLP-based baselines. For example, compared to recent MLPs, MS-MLP-B outperforms Hire-MLP-L and WaveMLP-B by 0.6% and 1.5%, respectively, on RetinaNet. In downstream tasks like object detection and segmentation, Hire-MLP-L and Wave-MLP-B usually require high-resolution input and both short- and long-range token interactions. Our regional mixing shows the effectiveness from this perspective, compared to the global-only or local-only token interactions.

Results on ADE20K: Besides the detection and instance segmentation tasks, we further evaluate our model on semantic segmentation, where we use UperNet (Xiao et al., 2018) with our pretrained models as the backbone. For all models, we use a standard recipe by

Table 8: Analogous comparison to Table 6 between using MS-MLP plugged-in Transformer architecture and their original architectures.

Backbone	RetinaNet 1×					Mask R-CNN 1×						
	Param / FLOPs	AP	AP_S	AP_M	AP_L	Param / FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
Swin-T (Liu et al., 2021b)	38.5M / 244G	41.5	**25.1**	44.9	55.5	47.8M / 264G	42.2	64.6	**46.2**	39.1	**61.6**	42.0
MS-MLP+Swin-T (ours)	38.5M / 262G	**42.0**	24.5	**45.4**	**57.4**	49.8M / 269G	**42.7**	64.8	45.9	**39.5**	59.8	**42.3**
Focal-Attention-T (Yang et al., 2021)	39.4M / 265G	43.7	**28.6**	47.4	56.9	44.8M / 291G	**44.8**	**67.7**	**49.2**	**41.0**	**64.7**	44.2
MS-MLP+Focal-Attention-T (ours)	39.4M / 284G	**44.3**	27.0	47.3	**58.9**	44.8M / 299G	44.6	65.7	48.7	40.6	62.6	**44.7**
Swin-S (Liu et al., 2021b)	59.8M / 334G	44.5	27.4	48.0	**59.9**	69.1M / 353G	44.8	66.6	48.9	40.9	63.4	44.2
MS-MLP+Swin-S (ours)	59.9M / 344G	**45.2**	**27.9**	**48.4**	59.5	69.1M / 379G	**45.3**	**67.2**	**49.6**	**41.0**	**64.1**	44.1
Focal-Attention-S (Yang et al., 2021)	61.7M / 367G	45.6	**29.5**	49.5	60.3	71.2M / 401G	47.4	**69.8**	51.9	42.8	66.6	46.1
MS-MLP+Focal-Attention-S (ours)	62.0M / 354G	**45.8**	27.8	**51.0**	**60.5**	72.7M / 446G	**47.5**	66.5	**52.3**	**43.0**	**67.3**	**46.4**
Swin-B (Liu et al., 2021b)	98.4M / 477G	45.0	**28.4**	49.1	**60.6**	107M / 496G	46.9	69.2	**51.6**	42.3	66.0	45.5
MS-MLP+Swin-B (ours)	98.7M / 502G	**45.3**	28.0	**49.4**	59.5	108.6M / 561G	**47.2**	**69.5**	51.3	**42.8**	66.0	**46.3**
Focal-Attention-B (Yang et al., 2021)	100.8M / 514G	46.3	**31.7**	50.4	60.8	110.0M / 533G	47.8	70.2	**52.5**	43.2	67.3	46.5
MS-MLP+Focal-Attention-B (ours)	104.2M / 609G	**46.6**	28.8	50.4	**62.2**	116.9M / 647G	**47.9**	**70.6**	52.4	**43.4**	**67.8**	**47.8**

Table 9: Configurations of different r_n and d_n of MS-MLP.

Configs	Local	Global	Isolated Regional	Regional
$r_{1:S}$	$[1, 1, 1, 1, 1]$	$[7, 7, 7, 7, 7]$	$[1, 1, 3, 5, 7]$	$[1, 1, 3, 5, 7]$
$d_{1:S}$	$[0, 1, 2, 3, 4]$	$[0, 1, 2, 3, 4]$	$[0, 2, 5, 10, 17]$	$[0, 1, 2, 3, 4]$

Table 10: Comparison of different patch size on ImageNet-1K.

Method	Patch size	Accuracy
Swin-T	2	81.4%
	4	81.3%
MS-MLP+Swin-T	2	81.9%
	4	81.6%

setting the input size as 512×512 and train the model for 160k iterations with a batch size of 16. The results are shown in Table 7. We can observe that MS-MLP achieves better single-scale mIoUs than the baselines and has competitive multi-scale mIoUs across different model capacities.

Results with Transformers: Similar to the classification tasks, we validate the effectiveness of combining MS-MLP with the Swin and Focal-Attention Transformers on COCO-2017 object detection and instance segmentation using their original settings. Table 8 shows an analogous comparison to Table 6 for this task. The MS-MLP+Transformer backbone improves their original Transformer backbones by 0.5-0.7% on both RetinaNet and Mask-RCNN.

4.3 Ablation studies

We conduct ablation studies using MS-MLP-Tiny by varying the region mixing size r_n and relative distance d_n to validate the effectiveness of regional mixing. We propose four different configurations as shown in Table 9: 1) all regional mixing sizes are restricted to 1 to achieve local mixing. This keeps only the shifting module and is close to AS-MLP (Lian et al., 2021). 2) All region sizes are set to 7 to cover as many tokens as possible. This setting aims to approximate the global mixing and is also related to W-MSA in Swin (Liu et al., 2021b). 3) We set the region to a different granularity but enlarge the shifting step size to let each region be isolated from each other. 4) The proposed regional mixing.

We show the classification results with inputs that have a patch size of 2 or 4 in Fig. 3. As we can see, only using local or global mixing has a lower performance than regional mixing. Especially, for global mixing without the self-attention mechanism, token-mixing becomes less flexible to capture important information. We observe the performance is even lower when the patch size is 2. For the isolated setting, without interactions within regions, information from different regions is hard to collect. This isolated mixing is even less effective than only using local information. When the patch size is 2, the number of input tokens becomes larger. We can see local and regional mixings both leverage finer-grained information improving classification accuracy, with significant improvements in regional mixing. This continues to validate the effectiveness of the regional mixing.

When combining MS-MLP with Transformers, we inject MS-MLP in both stages 0 and 1 of the Transformer to match the model parameters and throughputs. We study whether the Transformer itself can benefit from smaller input patches. We show a comparison between Swin-T and MS-MLP+Swin-T in Table 10, where we can see smaller patch inputs only improve Swin-T by 0.1%. However, the improvements from MS-MLP are more significant for both patch size 2 (81.4% vs. 81.9%) and 4 (81.3% vs. 81.6%).
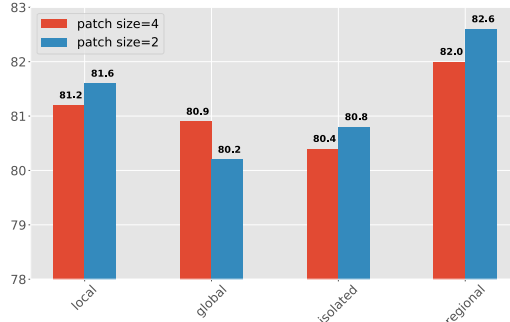
Figure 3: Image classification accuracy comparison with four configurations, as described in Table 9. Experiments are conducted on inputs of patch size 4 (red bins) and size 2 (blue bins).

5 Conclusion

In this paper, we present a regional mixing method via mixing and shifting operations to enable an efficient modeling of local-global dependency in MLPs, MS-MLP performs the token mixing at both fine-grain and coarse-grain levels, effectively handling both local and global dependencies with low computational cost. Extensive experiments show the effectiveness of MS-MLP over SoTA MLPs and other representative SoTA methods on both image classification and object detection and segmentation.

Acknowledgement

References

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Cai, Z. and Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

Chen, B., Li, P., Li, C., Li, B., Bai, L., Lin, C., Sun, M., Yan, J., and Ouyang, W. Glit: Neural architecture search for global and local image transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–21, 2021a.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Chen, S., Xie, E., Ge, C., Liang, D., and Luo, P. Cyclemlp: A mlp-like architecture for dense prediction. *arXiv preprint arXiv:2107.10224*, 2021b.

Contributors, M. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Guo, J., Tang, Y., Han, K., Chen, X., Wu, H., Xu, C., Xu, C., and Wang, Y. Hire-mlp: Vision mlp via hierarchical rearrangement. *arXiv preprint arXiv:2108.13341*, 2021.

Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., and Wang, Y. Transformer in transformer. *arXiv preprint arXiv:2103.00112*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016a.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016b.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2017.

Hou, Q., Jiang, Z., Yuan, L., Cheng, M.-M., Yan, S., and Feng, J. Vision permutator: A permutable mlp-like architecture for visual recognition. *arXiv preprint arXiv:2106.12368*, 2021.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In *European conference on computer vision*, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 2015.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, pp. 1097–1105, 2012.

LeCun, Y., Bengio, Y., et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Li, J., Hassani, A., Walton, S., and Shi, H. Convmlp: Hierarchical convolutional mlps for vision. *arXiv preprint arXiv:2109.04454*, 2021.

Lian, D., Yu, Z., Sun, X., and Gao, S. As-mlp: An axial shifted mlp architecture for vision. *arXiv preprint arXiv:2107.08391*, 2021.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision*, 2014.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2017.

Liu, H., Dai, Z., So, D. R., and Le, Q. V. Pay attention to mlps. *arXiv preprint arXiv:2105.08050*, 2021a.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021b.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. *arXiv preprint arXiv:2201.03545*, 2022.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Lou, Y., Xue, F., Zheng, Z., and You, Y. Sparse-mlp: A fully-mlp architecture with conditional computation. *arXiv preprint arXiv:2109.02008*, 2021.

Mao, X., Qi, G., Chen, Y., Li, X., Ye, S., He, Y., and Xue, H. Rethinking the design principles of robust vision transformer. *arXiv preprint arXiv:2105.07926*, 2021.

Melas-Kyriazi, L. Do you even need attention? a stack of feed-forward layers does surprisingly well on imagenet. *arXiv preprint arXiv:2105.02723*, 2021.

Polyak, B. T. and Juditsky, A. B. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollár, P. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10428–10436, 2020.

Rao, Y., Zhao, W., Zhu, Z., Lu, J., and Zhou, J. Global filter networks for image classification. *arXiv preprint arXiv:2107.00645*, 2021.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Tang, C., Zhao, Y., Wang, G., Luo, C., Xie, W., and Zeng, W. Sparse mlp for image recognition: Is self-attention really necessary? *arXiv preprint arXiv:2109.05422*, 2021a.

Tang, Y., Han, K., Guo, J., Xu, C., Li, Y., Xu, C., and Wang, Y. An image patch is a wave: Phase-aware vision mlp. *arXiv preprint arXiv:2111.12294*, 2021b.

Tatsunami, Y. and Taki, M. Raftmlp: Do mlp-based models dream of winning over computer vision? *arXiv preprint arXiv:2108.04384*, 2021.

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*, 2021.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Joulin, A., Synnaeve, G., Verbeek, J., and Jégou, H. Resmlp: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404*, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021a.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021b.

Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., and Zhang, L. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021a.

Wu, K., Peng, H., Chen, M., Fu, J., and Chao, H. Rethinking and improving relative position encoding for vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10033–10041, 2021b.

Xiao, T., Liu, Y., Zhou, B., Jiang, Y., and Sun, J. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

Yang, J., Li, C., Zhang, P., Dai, X., Xiao, B., Yuan, L., and Gao, J. Focal attention for long-range interactions in vision transformers. *Advances in Neural Information Processing Systems*, 34, 2021.

Yu, T., Li, X., Cai, Y., Sun, M., and Li, P. S^2-mlp: Spatial-shift mlp architecture for vision. *arXiv preprint arXiv:2106.07477*, 2021.

Yu, T., Li, X., Cai, Y., Sun, M., and Li, P. S^2-mlp: Spatial-shift mlp architecture for vision. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 297–306, 2022.

Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z., Tay, F. E., Feng, J., and Yan, S. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., and Torralba, A. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 2019.

A Experiment settings

A.1 Image classification on ImageNet-1K

Following the settings of the Swin transformer (Liu et al., 2021b), image classification is performed by applying an adaptive global average pooling layer on the output feature map of the last stage followed by a linear classifier. In evaluation, the top-1 accuracy using a single crop is reported.

The training settings mostly follow the Swin transformer (Liu et al., 2021b) and AS-MLP (Lian et al., 2021). For all model variants, we adopt a default input image resolution of 224×224. When training from scratch with an input of size 224×224, we employ an AdamW (Loshchilov & Hutter, 2017) optimizer for 300 epochs using a cosine decay learning rate scheduler with 20 epochs of linear warm-up. Keeping a batch size of 1,024, an initial learning rate of 0.001, a weight decay of 0.05, and a gradient clipping with a max norm of 1 are used. We include most of the augmentation and regularization strategies of (Touvron et al., 2020) in training, including Rand-Augment (Cubuk et al., 2020), MixUp (Zhang et al., 2017), CutMix (Yun et al., 2019), Label Smoothing (Szegedy et al., 2016), Random Erasing (Zhong et al., 2020), and DropPath (Huang et al., 2016). Different from the Swin and Focal-Attention Transformer, we empirically find the Exponential Moving Average (EMA) (Polyak & Juditsky, 1992) can enhance performance, but for a fair comparison we did not report the EMA results in the paper. An increasing degree of stochastic depth augmentation is employed for larger models, *i.e.*, 0.2, 0.3, 0.5 for MLP-T, MLP-S, and MLP-B, respectively.

A.2 Downstream tasks: Object detection on COCO and semantic segmentation on ADE20K

On COCO-2017 tasks, we consider four typical object detection frameworks: RetinaNet (Lin et al., 2017), Mask R-CNN (He et al., 2017), and Cascade Mask R-CNN (Cai & Vasconcelos, 2018) in mmdetection (Chen et al., 2019). We utilize the single-scale training and multi-scale training for the "1x" and "3x" (resizing the input such that the shorter side is between 480 and 800 while the longer side is at most 1,333) schedules, respectively. For these frameworks, we utilize the same settings where we adopt the AdamW optimizer (Loshchilov & Hutter, 2017) (initial learning rate of 0.0001, weight decay of 0.05, and a batch size of 16), and "3x" schedule (36 epochs with the learning rate decayed by $10\times$ at epochs 27 and 33).

On ADE20K (Zhou et al., 2019), which contains 20,210 training images and 2,000 validation images, we conduct semantic segmentation experiments following the settings in (Wang et al., 2021a; Chen et al., 2021b; Liu et al., 2021b). We use UperNet (Xiao et al., 2018) as the segmentation framework and a pretrained MS-MLP as the backbone. For all models, we use a standard recipe by setting the input size to 512×512 and train the model for 160k iterations with a batch size 16 in the training. We follow the training settings described in Liu et al. (2021b), where we employ the AdamW optimizer (Loshchilov & Hutter, 2017) with an initial learning rate of 6×10^{-5}, a weight decay of 0.01, a scheduler that uses linear learning rate decay, and a linear warmup of 1,500 iterations. For augmentations, we adopt the default setting in mmsegmentation (Contributors, 2020) of random horizontal flipping, random re-scaling within ratio range $[0.5, 2.0]$, and random photometric distortion. Stochastic depth with ratio of 0.2 is applied for all models.

B Additional Experiment Results.

B.1 Instance segmentation with 3x schedule

For the object detection and instance segmentation tasks on COCO-2017, we also train models with $3\times$ schedule and multi-scale training strategy as described in the main experiments. The results of Mask R-CNN (He et al., 2017) and Cascade Mask R-CNN (Cai & Vasconcelos, 2018) are shown in Table 11. Similar to the results in the "1x" schedule, the proposed MS-MLP achieves a higher performance.

Table 11: Instance segmentation results on COCO val2017. Mask R-CNN and Cascade Mask R-CNN are trained on the "3x" schedule.

Backbone	Mask R-CNN 3×							Cascade Mask R-CNN 3×						
	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet50 (He et al., 2016b)	260G	41.0	61.7	44.9	37.1	58.4	40.1	738G	46.3	64.3	50.5	40.1	61.7	43.4
AS-MLP-T (Lian et al., 2021)	260G	46.0	67.5	50.7	41.5	64.6	44.5	739G	50.1	68.8	54.3	43.5	66.3	46.9
Swin-T (Liu et al., 2021b)	264G	46.0	68.2	50.2	41.6	65.1	44.8	742G	50.5	69.3	54.9	43.7	66.6	47.1
Hire-MLP-Small (Guo et al., 2021)	256G	46.2	68.2	50.9	42.0	65.6	45.3	734G	50.7	69.4	55.1	44.2	66.9	48.1
MS-MLP-T (ours)	262G	46.2	67.8	50.8	41.7	65.2	45.0	744G	50.4	69.2	54.6	43.7	66.5	47.6
Swin-S (Liu et al., 2021b)	354G	48.5	70.2	53.5	43.3	67.3	46.6	832G	51.8	70.4	56.3	44.7	67.9	48.5
AS-MLP-S (Lian et al., 2021)	346G	47.8	68.9	52.5	42.9	66.4	46.3	823G	51.1	69.8	55.6	44.2	67.3	48.1
Hire-MLP-Base (Guo et al., 2021)	334G	48.1	69.6	52.7	43.1	66.8	46.7	813G	51.7	70.2	56.1	44.8	67.8	48.5
MS-MLP-S (ours)	424G	48.6	70.8	53.4	43.7	67.7	47.2	841G	51.9	70.8	56.6	43.7	66.5	47.3
Swin-B (Liu et al., 2021b)	496G	48.5	70.2	53.5	43.3	67.3	46.6	982G	51.9	71.8	57.5	45.8	67.4	49.7
MS-MLP-B (ours)	561G	49.0	70.0	52.6	43.7	65.4	46.7	986G	52.6	71.4	57.2	45.4	68.9	49.3

B.2 Image classification with different architecture

In the main paper we set the MS-MLP in an architecture with four-stages, where the number of blocks keeps a ratio of 1:1:3:1. To match the parameter size and throughputs, these stages contain 3-3-9-3 blocks for MS-MLP-T and 3-3-27-3 for MS-MLP-S and MS-MLP-B. We also match the architecture with Swin (Liu et al., 2021b) and AS-MLP (Lian et al., 2021), using the stage design of 2-2-6-2 for MS-MLP-T and 2-2-18-2 for MS-MLP-S and MS-MLP-B. The results are summarized in Table 12. We can see the results are comparable to the results shown for Swin (Liu et al., 2021b) and AS-MLP (Lian et al., 2021), while the parameter size and FLOPs are fewer.

For the regional mixing, we choose to use depth-wise convolution with different kernel size. We conduct experiments to see the effects of the kernel sizes and convolution type as an additional ablation. In Table 13, we can observe the full convolution has no special effects to the final results, while using full convolution increases the FLOPs and slows down the training. We also observe slightly changing the kernel size does not affect the results. However, the results show that decreasing the region size while the relative distance increases degrades model performance.

Table 12: Image classification results of different architecture MS-MLP.

Model	Blocks	#Parameters	FLOPs	Top-1 acc.
MS-MLP-T	2-2-6-2	24M	4.3G	81.4%
	2-2-2-6-2	24M	4.4G	81.3%
MS-MLP-S	2-2-18-2	42M	7.8G	82.8%
	2-2-2-18-2	42M	7.8G	83.0%
MS-MLP-B	2-2-18-2	74M	13.8G	83.3%
	2-2-2-18-2	74M	13.9G	83.2%

Table 13: Image classification results of different MS-block configurations.

Region size	Conv type	FLOPs	Top-1 acc.
1-1-3-5-7	DW	4.9G	82.1%
1-1-3-5-7	Full	7.7G	82.0%
1-3-5-7-9	DW	5.6G	81.8%
1-3-5-7-9	Full	9.1G	82.0%
1-7-5-3-1	DW	4.9G	81.1%
1-7-5-3-1	Full	7.7G	81.3%
1-5-3-3-1	DW	4.6G	81.2%
1-5-3-3-1	Full	6.8G	81.4%

C   More discussion and limitations

We demonstrate MS-MLP can perform as well as representative ViTs and CNNs on image classification, object detection, instance, and semantic segmentation tasks. While our goal is to offer a general way to handle global and local visual dependencies, MS-MLP still relies on a careful choice of the region size and the relative shifting distance, and we are not able to explore all possible configurations. As we realize computer vision applications are diverse, the current MS-MLP configuration may be suited for certain tasks, and we may need to explore a more general recipe for other tasks.

Table 14: Full comparison of MS-MLP architecture with SOTA models on ImageNet-1K.

Model	Family	Params.	FLOPs	Throughput (image / s)	Top-1 acc. (%)
ResNet18 (He et al., 2016a)	CNN	12M	1.8G	-	69.8
ResNet50 (He et al., 2016a)	CNN	26M	4.1G	-	78.5
ResNet101 (He et al., 2016a)	CNN	45M	7.9G	-	79.8
RegNetY-4G (Radosavovic et al., 2020)	CNN	21M	4.0G	1157	80.0
RegNetY-8G (Radosavovic et al., 2020)	CNN	39M	8.0G	592	81.7
RegNetY-16G (Radosavovic et al., 2020)	CNN	84M	16.0G	335	82.9
ConvNeXt-T (Liu et al., 2022)	CNN	29M	4.5G	774.7	82.1
ConvNeXt-S (Liu et al., 2022)	CNN	50M	8.7G	447.1	83.1
ConvNeXt-B (Liu et al., 2022)	CNN	89M	15.4G	292.1	83.8
GFNet-H-S (Rao et al., 2021)	FFT	32M	4.5G	-	81.5
GFNet-H-B (Rao et al., 2021)	FFT	54M	8.4G	-	82.9
DeiT-S (Touvron et al., 2020)	Trans	22M	4.6G	940	79.8
DeiT-B (Touvron et al., 2020)	Trans	86M	17.5G	292	81.8
PVT-Small (Wang et al., 2021b)	Trans	25M	3.8G	820	79.8
PVT-Medium (Wang et al., 2021b)	Trans	44M	6.7G	526	81.2
PVT-Large (Wang et al., 2021b)	Trans	61M	9.8G	367	81.7
T2T-ViT-14 (Yuan et al., 2021)	Trans	22M	5.2G	764	81.5
T2T-ViT-19 (Yuan et al., 2021)	Trans	39M	8.9G	464	81.9
T2T-ViT-24 (Yuan et al., 2021)	Trans	64M	14.1G	312	82.3
TNT-S (Han et al., 2021)	Trans	24M	5.2G	428	81.5
TNT-B (Han et al., 2021)	Trans	66M	14.1G	246	82.9
iRPE-K (Wu et al., 2021b)	Trans	87M	17.7G	-	82.4
iRPE-QKV (Wu et al., 2021b)	Trans	22M	4.9G	-	81.4
GLiT-Small (Chen et al., 2021a)	Trans	25M	4.4G	-	80.5
GLiT-Base (Chen et al., 2021a)	Trans	96M	17.0G	-	82.3
MS-MLP-T (ours)	MLP	28M	4.9G	792	82.1
MS-MLP-S (ours)	MLP	50M	9.0G	484	83.4
MS-MLP-B (ours)	MLP	88M	16.1G	366	**83.8**
Swin-T (Liu et al., 2021b)	Trans	29M	4.5G	755	81.3
MS-MLP + Swin-T (ours)	MLP + T	29M	4.5G	779	**81.9** (+0.6)
Swin-S (Liu et al., 2021b)	Trans	50M	8.7G	437	83.0
MS-MLP + Swin-S (ours)	MLP + T	50M	8.7G	464	**83.5** (+0.5)
Swin-B (Liu et al., 2021b)	Trans	88M	15.4G	278	83.3
MS-MLP + Swin-B (ours)	MLP + T	88M	15.4G	279	**83.8** (+0.5)
Focal-Attention-T (Yang et al., 2021)	Trans	29M	4.9G	319	82.2
MS-MLP + Focal-Attention-T (ours)	MLP + T	29M	5.6G	451	**82.8** (+0.6)
Focal-Attention-S (Yang et al., 2021)	Trans	52M	9.4G	192	83.5
MS-MLP + Focal-Attention-S (ours)	MLP + T	52M	10.1G	297	**83.9** (+0.4)
Focal-Attention-B (Yang et al., 2021)	Trans	90M	16.4G	138	83.8
MS-MLP + Focal-Attention-B (ours)	MLP + T	90M	17.6G	207	**84.0** (+0.2)