Strip-MLP: Efficient Token Interaction for Vision MLP

Guiping Cao[1,2]   Shengda Luo[1]   Wenjian Huang[1]   Xiangyuan Lan[2,*]
Dongmei Jiang[2]   Yaowei Wang[2]   Jianguo Zhang[1,2,*]

[1]Southern University of Science and Technology, Shenzhen, China
[2]Peng Cheng Laboratory, Shenzhen, China

12131099@mail.sustech.edu.cn, {luosd, huangwj, zhangjg}@sustech.edu.cn,
{lanxy, jiangdm, wangyw}@pcl.ac.cn

Abstract

*Token interaction operation is one of the core modules in MLP-based models to exchange and aggregate information between different spatial locations. However, the power of token interaction on the spatial dimension is highly dependent on the spatial resolution of the feature maps, which limits the model's expressive ability, especially in deep layers where the feature are down-sampled to a small spatial size. To address this issue, we present a novel method called **Strip-MLP** to enrich the token interaction power in three ways. Firstly, we introduce a new MLP paradigm called Strip MLP layer that allows the token to interact with other tokens in a cross-strip manner, enabling the tokens in a row (or column) to contribute to the information aggregations in adjacent but different strips of rows (or columns). Secondly, a **C**ascade **G**roup **S**trip **M**ixing **M**odule (CGSMM) is proposed to overcome the performance degradation caused by small spatial feature size. The module allows tokens to interact more effectively in the manners of within-patch and cross-patch, which is independent to the feature spatial size. Finally, based on the Strip MLP layer, we propose a novel **L**ocal **S**trip **M**ixing **M**odule (LSMM) to boost the token interaction power in the local region. Extensive experiments demonstrate that Strip-MLP significantly improves the performance of MLP-based models on small datasets and obtains comparable or even better results on ImageNet. In particular, Strip-MLP models achieve higher average Top-1 accuracy than existing MLP-based models by +2.44% on Caltech-101 and +2.16% on CIFAR-100. The source codes will be available at https://github.com/MedProcess/Strip_MLP.*

1. Introduction

In computer vision, Convolutional Neural Networks (CNNs) are one of the most popular network backbones, which have made a series of breakthroughs [8]. Inspired by
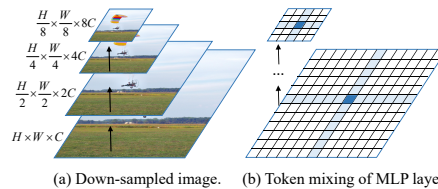
*Corresponding author.

Figure 1. Token interaction of the MLP layer on the down-sampled image feature. (a) The process of which the feature resolution will be down-sampled from $H \times W \times C$ into a small spatial size of $\frac{H}{8} \times \frac{W}{8} \times 8C$. (b) Token interaction on different feature resolutions.

the great success of self-attention-based architectures [32] in natural language processing (NLP), the Transformer models [5, 19, 34] are introduced into the field of computer vision, and have achieved comparable results with state-of-the-art (SOTA) CNNs. Although ViT [5] and its variants outperform traditional CNNs, the models introduce high computational complexity to construct attention maps. Recently, some studies [20, 26] in the vision community suggest that the attention mechanism is not necessary and simpler model architectures are proposed.

MLP-based models, like MLP-Mixer [28], gMLP [16] and ViP [10] process the data with the Multilayer Perceptrons (MLP), showing great potential to improve the performance of vision models [29]. As the first visual deep MLP network, MLP-Mixer [28] introduces two types of MLP layers: *Channel-Mixing MLPs* (CMM) and *Token-Mixing MLPs* (TMM). For CMM, the module mainly mixes the information between different channels of each token. For TMM, it allows each spatial token to interact with all other tokens (the whole image) in a single MLP layer. However, this design also introduces a much larger number of parameters and higher computational complexity prone to overfitting. To address this problem, Sparse MLP (SMLP) [26] and Vision Permutator (ViP) [10] propose a similar layer of parallel structure, which applies one dimension MLP along the axial directions, and parameters are shared among rows or columns, respectively. Therefore, it reduces the number

of model parameters and computational complexity, avoiding the common over-fitting problem.

Although SMLP and ViP alleviate some deficiencies of MLP-Mixer [28], both methods bring the challenge that the token interaction power is highly dependent on the feature spatial size when interacting tokens on spatial rows (or columns). As shown in Fig. 1, the spatial feature resolution is down-sampled to a small size but with more channels, which means the feature pattern of each token is mainly concentrated on the channel dimension rather than the spatial one. Interacting tokens along the spatial dimension by sharing the weights among all channels would seriously *ignore the feature pattern differences among different channels, which may degrade the token interaction power*, especially in deep layers with small spatial feature resolution. Here, we mark this problem as *the Token's interaction dilemma*. Taking SMLP as an example, we analyze the feature resolution and complexity of the model in different stages in detail (seen in Sec. 3.4). We find that as the spatial feature size decreases by down-sampling stage by stage, the token interaction layer also becomes smaller and smaller, which makes the token interaction power degraded rapidly.

To address these challenges, we propose a new efficient Strip MLP model, dubbed **Strip-MLP**, to enrich the power of the token interaction layer in three ways. For the level of a single MLP layer, inspired by the *cross-block* normalization schemes of HOG [3] and the sparse connections between the biological neurons, we design a Strip MLP layer to allow the token to interact with other tokens in a cross-strip manner, enabling each row or column of the tokens to contribute differently to other rows or columns. For the token interaction module level, we develop channel-wise group mixing of CGSMM, enabling the tokens in a row (or column) to contribute to the information aggregations in adjacent but different strips of rows (or columns). to tackle the problem that the token interaction power decreases in deep layers with the spatial feature size significantly reduced but with multiplying channels. Considering the existing methods [10, 26, 28] interact the tokens mainly in the long range of row (or column), which may not aggregate tokens well in the local region, we propose a new **L**ocal **S**trip **M**ixing **M**odule (LSMM) with a small Strip MLP unit to strengthen the token interaction power on local interactions.

The proposed Strip-MLP model significantly boosts the token interaction power, and the main contributions are:

- A new MLP paradigm for vision MLP: Strip MLP layer, which aggregates the adjacent tokens in a cross-strip manner and enables each row or column of the tokens to contribute differently to other rows or columns, interacting tokens more efficiently.
- Designing a Cascade Group Strip Mixing module and a Local Strip Mixing module, which effectively improves the model's token interaction power and boosts

the tokens aggregation in the local region, respectively;
- Extensive experiments show that Strip-MLP remarkably improves the performances of the MLP-based models. Strip-MLP achieves higher average Top-1 accuracy by +2.44% on Caltech-101 and +2.16% on CIFAR-100 over the existing MLP-based models. In addition, our models achieve comparable or even better performances on ImageNet-1K compared with traditional MLP-based models, and other popular CNNs and transformer-based models.

2. Related Work

Deep neural networks for vision recognition can be mainly divided into three categories: Convolutional Neural Networks (CNNs), Vision Transformers (ViTs), and Multi-Layer Perceptron-based models (MLPs) [10].

CNNs-Based Models. CNNs are the de-facto standard deep learning network models for vision tasks and have been intensely studied in the vision community. AlexNet [15] is a symbolically significant model that won the ILSVRC 2012 contest with far higher performance than others. Since then, CNNs-based models have attracted more attentions, and lots of deeper and more effective architectures [8, 11, 23–25, 33, 39, 40] have been proposed. With the convolution and pooling layer, CNNs aggregate the feature in a local region but not well in the long-term dependencies, which are optimized by the new vision model like Transformer [32] and MLP [28] models.

Transformer-Based Models. Transformer [32] is introduced for machine translation and becomes the reference model for all-natural language processing (NLP) tasks. Inspired by the great success of Transformer in NLP, ViTs [5] first applies a standard Transformer to images, which attains excellent results compared to the SOTA CNNs model. DeiT [30] introduces several training strategies and distillation methods to make ViTs more effective on the smaller ImageNet-1K dataset. By proposing a hierarchical Transformer with shifted windows, Swin Transformer [19] achieves SOTA accuracy on ImageNet-1K, bringing greater efficiency by self-attention in local window and cross-window connection. For these models, self-attention is the core module but with heavy computational burdens to obtain an attention map.

MLP-Based Models. Without the convolutions and self-attention mechanism, MLP-Mixer [28] builds the architecture that only uses the MLP layer and achieves competitive performance on image classification benchmarks. Since then, the researchers have developed many MLP-like variants [1, 7, 16, 26, 27, 29, 31, 35, 38] models. The work [17] gives a comprehensive survey about visual deep MLP models and compares the intrinsic connections and differences between convolution, self-attention mechanism, and Token-mixing MLP in detail. Sparse MLP [26] introduced a sparse
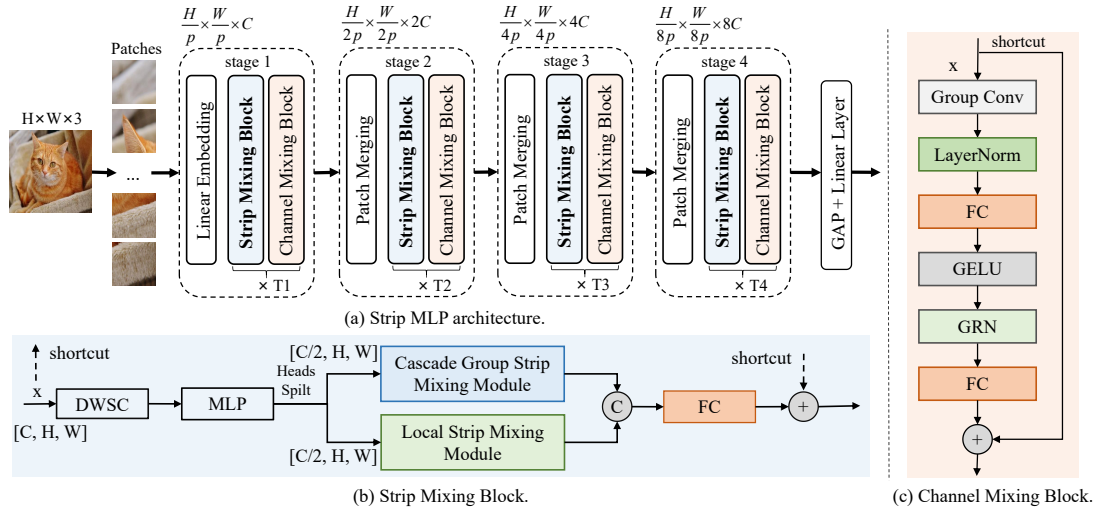
Figure 2. The overall and components architecture of Strip-MLP. (a) Strip-MLP has four stages, and $T1$ to $T4$ means the repeated times of the block in each stage. (b) The Strip Mixing Block splits the head (channel dimension) into two heads, and they are fed into the two parallel branches. (c) Channel Mixing Block architecture.

operation to separately aggregate information along axial directions, avoiding the quadratic computational complexity of conventional MLP. Hire-MLP [7] presents a novel variant of MLP-based architecture via hierarchically rearranging tokens to aggregate local and global spatial information. Wave-MLP [27] represents each token as a wave function with two parts of amplitude and phase, which has the ability to model varying contents from different input images. Above all, there is still the limitation of token interaction power degraded significantly especially when the spatial feature resolution becomes small, which has been overlooked by previous studies. In this paper, we aim to enrich the token interaction power on both the single MLP layer and token interaction module to advance the performance of MLP-based models.

3. Method

In this section, we first present the overall architecture of Strip-MLP. Then, we show the key proposed components: Strip MLP Layer, **C**ascade **G**roup **S**trip **M**ixing **M**odule (CGSMM) and **L**ocal **S**trip **M**ixing **M**odule (LSMM) of the model in detail, and present the comparison analysis between the Strip MLP and traditional MLP model in terms of parameters and complexity. Finally, we define four kinds of architecture variants to compare the performance of the models in different sizes.

3.1. Overall Architecture

An overview of the Strip-MLP model is depicted in Fig. 2 (a). We design the Strip-MLP as a hierarchical structure of a multi-stages model. Given the input image I, Strip-MLP model makes patch embedding and models the image feature in four stages.

Patch Embedding. Strip-MLP firstly splits the input

image $I \in \mathbb{R}^{H \times W \times 3}$ (H, W: image height and width) into a sequence of image patches (also referred as tokens) $I_p \in \mathbb{R}^{hw \times c}$ (c: the number of channels), where the patch size is $p \times p$ and patch number is hw ($h = \frac{H}{p}, w = \frac{W}{p}$), and then all patches are linearly projected into a desired higher dimension C of the feature ($X \in \mathbb{R}^{hw \times C}$).

Mixing Block. The purpose of mixing block is to boost the interactions between the features of diverse spatial locations and channels. To effectively aggregate spatial and channel information, we design two sub-blocks of the *Strip Mixing Block* and *Channel Mixing Block*. The Strip Mixing Block is comprised of CGSMM and LSMM, which blend and aggregate spatial information more efficiently at the global and local level, respectively. Patch merging module aims to merge the feature in which the spatial dimension will be reduced by 2×2, and the channel dimension increases by 2 times so that the model down-samples the feature from $\frac{H}{p} \times \frac{W}{p}$ into $\frac{H}{8p} \times \frac{W}{8p}$ stage by stage. To obtain multi-scale features, we apply a single convolutional layer to the output features of stages 1 and 2. Then, we add the resulting features to the input features of stages 3 and 4.

Head Layer. The features extracted by the multiple blocks are fed into a global average pooling (GAP) layer to reduce the feature dimension. Finally, the feature will be fed into a fully-connected head layer for classification.

3.2. Strip Mixing Block and Channel Mixing Block

Strip Mixing Block. To improve the token interaction power, we design the block to aggregate both long and short range interactions in a parallel manner. As illustrated in Fig. 2 (b), we split the feature in channel dimension, with one-half of the channel feature being fed into CGSMM to model the long-range interactions, and the remaining feature being fed into LSMM to aggregate the local interac-
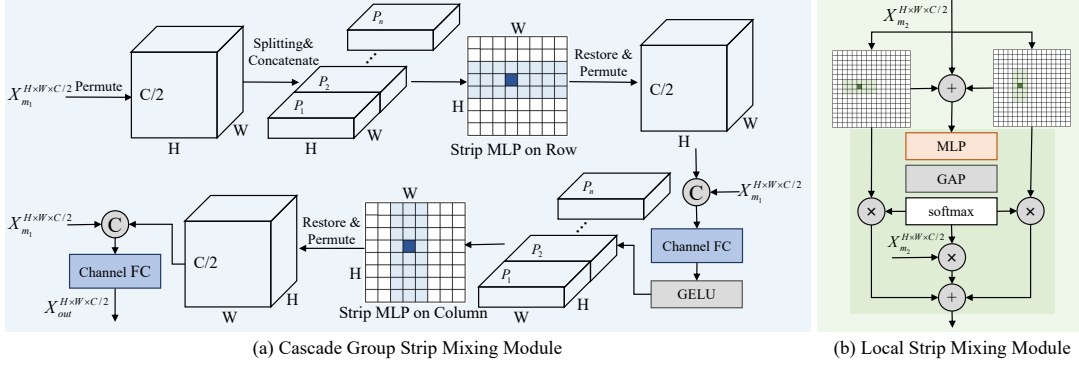
Figure 3. (a) The architecture of CGSMM. It contains the Strip MLP layer, the feature splitting, restoring, and permuting operation in a cascade mode. (b) The structure of the Local Strip Mixing module. The dark green block composes of a re-weight module.

tions. Given the input feature $X \in \mathbb{R}^{H \times W \times C}$, the block can be formulated as:

$$X_m = MLP((DWSC(X))) \qquad (1)$$

$$Y = FC(Cat(CGSMM(X_{m_1}^{\frac{C}{2}}), LSMM(X_{m_2}^{\frac{C}{2}}))) + X \qquad (2)$$

where X_m and Y are the intermediate and output feature of the block. X_m is split into X_{m_1} and X_{m_2} with half channel, respectively. DWSC means depth-wise convolution [2], and the kernel size is 3×3. MLP is a serial connection of fully-connection (FC), batch normalization [12] and GELU [9] activation layers. $Cat(*)$ denotes concatenation operation.

Channel Mixing Block. The block aims to aggregate the information between channels of the token, and the basic structure is shown in Fig. 2 (c). Following the inverted bottleneck in [20] and global response normalization (GRN) in [36], we design the Channel Mixing block to increase the contrast and selectivity of channels.

3.3. CGSMM and LSMM

We first introduce the Strip MLP Layer. Based on this layer, we design CGSMM using a simple but effective strategy that splits the feature into patches along channel dimension and interacts tokens more effectively in the manners of within-patch and cross-patch, which is independent to the feature spatial size. In addition, the design of existing methods [10, 26, 28] only allows tokens to interact in a long range of rows (or columns) with sharing weight; thus the resulting model may struggle with effectively aggregating both global and local information simultaneously. Therefore, we design LSMM to capture the local interactions more efficiently.

Strip MLP Layer. In MLP-based models, the majority of MLP layers treat each row and column of the data in isolation, as formulated in Eq. (3), which may lead to the inefficiency of token interaction. Inspired by the *cross-block* normalization schemes of HOG [3], which overlaps the blocks and enables each scalar cell response to contribute into different blocks, we propose the Strip MLP layer. The proposed layer applies MLP on the "strip" data of adjacent

rows or columns in the spatial direction in order to aggregate the feature in a cross-strip manner. Given the input X, we formulated the Strip MLP layer (setting strip width as 3 for example) in Eqs. (4) and (5):

$$X_{*,j}^h = W_1 X_{*,j}; \quad X_{i,*}^w = W_2 X_{i,*} \qquad (3)$$

$$X_{*,j}^h = W_3 Cat(X_{*,j-1}, X_{*,j}, X_{*,j+1}) \qquad (4)$$

$$X_{i,*}^w = W_4 Cat(X_{i-1,*}, X_{i,*}, X_{i+1,*}) \qquad (5)$$

where $W_1 \sim W_4$ are the weights of the MLP layer, i and j are the token index in row and column. The superiorities of the Strip MLP layer are mainly in two aspects: on one hand, Strip MLP layer enables the token to interact with other tokens in both short and long spatial ranges simultaneously. On the other hand, similar to the HOG cross-block normalization process, each row (or column) not only serves for the current row (or column) tokens aggregation but also makes contribution to the adjacent rows (or columns) feature aggregation. For example, in Eq. (4), $X_{*,j}$ makes different contributions for aggregating the processed feature of $X_{*,j-1}^h$, $X_{*,j}^h$ and $X_{*,j+1}^h$, making tokens interacting more efficiently in a cross-strip manner.

Cascade Group Strip Mixing Module. The architecture is shown in Fig. 3 (a). The module applies the Strip MLP layer in row and column direction in a cascade mode. As the operation of the Strip MLP layer on row is similar to that on column, we take one of them to show the method.

Patch Splitting of the Data along Channel Dimension. Given the input feature $X_{m_1} \in \mathbb{R}^{H \times W \times \frac{C}{2}}$, the module firstly permutes the feature into $X_{m_1}^{H \times \frac{C}{2} \times W}$, and splits the feature into P patches in the channel dimension and concatenated along the row ($X^{H \times \frac{C}{2P} \times PW}$) dimension.

Group Strip MLP Layer. To improve the token interaction power, we propose a *Group Strip MLP Layer* (GSML) to interact tokens in the manners of within-patch and cross-patch. In particular, we apply the unshared weights of Strip MLP on different patches, and the tokens within the same patch share the weights for interaction. Then, we restore the feature to original shape and concatenate it with the input

feature. To interact the tokens crossing patches, we apply a channel fully-connected (Channel FC) layer to interact tokens between different patches.

Local Strip Mixing Module. The module is illustrated in Fig. 3 (b). In order to better aggregate the local interactions on spatial dimension, we define a small Strip MLP unit, where the strip width and length are 3 and 7, respectively. Given the input feature $X_{m_2}^{H \times W \times \frac{C}{2}}$, we aggregate the local interactions in rows and columns direction simultaneously. Following [1, 10, 27], we sum all branches with a re-weight module.

3.4. Parameter and Complexity Analysis

In multi-stage processing architecture, *the Token's interaction dilemma* becomes more serious, as the feature spatial resolution will be down-sampled in deep layers, and the token interaction power will be weakened. In this section, we present the comparison analysis of parameters and complexity to show the effectiveness of the proposed CGSMM. Considering the majority MLP-Based models have the similar model structure of token interaction, we analyze the parameters and complexities with Sparse MLP [26], which is a popular model with good performance on various datasets.

Sparse MLP [26] firstly applies MLP on the columns and rows of X to map $\mathbb{R}^{CW \times H}$ to $\mathbb{R}^{CW \times H}$ separately. Then, the model concatenates the two processed features with the input feature X and uses a linear layer on the channel dimension to fuse the feature from $\mathbb{R}^{HW \times 3C}$ to $\mathbb{R}^{HW \times C}$. The number of parameters and FLOPs of the first step are $W^2 + H^2$ and $CHW(H + W)$; they are $3C^2$ and $3HWC^2$ for the fusion step. Comparatively, the number of parameters and FLOPs of Strip MLP layer for token interaction are $3P(H^2 + W^2)$ and $3CHW(H + W)$, respectively. The fusion step has $4C^2$ parameters and $4HWC^2$ FLOPs. In particular, for $H_1 = W_1 = 56, H_4 = W_4 = 7, C_1 = 112, C_4 = 112 \times 2^3 = 896$, and $P = \frac{C}{4}$, we calculate the number of parameters and FLOPs of both two layers. As shown in Tab. 1, for the Sparse MLP block, the number of parameters for token interaction in stage 4 (only $0.10k$) decreases by 62.70 times than in stage 1. In addition, most of the number of the block's parameters and the FLOPs concentrate in the fusion step. For example, only 0.01% parameters (0.52% FLOPs) are used for the token interaction step in stage 4. Based on the above analysis, to improve the token interaction power, it would be better to redesign the block to balance the number of parameters and FLOPs.

In CGSMM, we use a simple but effective strategy that splits the feature along channel dimension into patches and interacts the tokens in the manners of within-patch and cross-patch. No matter how the spatial resolution decreases, the module can still interact the tokens with channel-wise specificity in different patches. In Tab. 1, we improve the *Proportion* of the number of parameters in the token inter-

Items	Sparse MLP		Strip MLP	
	Params	FLOPs	Params	FLOPs
Stage 1	$6.27k$	39.34M	$526.85k$	118.01M
Stage 4	$0.10k$	0.62M	$65.86k$	1.84M
Changes	↓62.70	↓63.45	↓**8.00**	↓64.14
Stage 4 Fusion	2.41M	118.01M	3.21M	157.35M
Proportion	0.01%	0.52%	**2.01%**	**1.16%**

Table 1. The model parameters and complexity comparison between Sparse MLP and Strip MLP layer in different stages. *Stage 1* and *Stage 4* show the parameters (Params) and floating-point operations (FLOPs) for token interaction in one layer between different stages. *Stage 4 Fusion* indicates the Params and FLOPs in the feature fusion step of the token interaction module. *Proportion* means the ratio between the token interaction step and other steps in the token interaction module of stage 4.

action layer from 0.01% to 2.01% in stage 4. Although our GSML brings more parameters and computations when just considering one token interaction layer, the total number of parameters and overall computational complexity of the model are decreased as the token interaction power is improved and we set small model configurations of $T_1 \sim T_4$. Additionally, the experiments in Sec. 4 show that our models achieve better performance with fewer parameters and FLOPs than other SOTA models.

3.5. Architecture Variants

We developed four variants of our Strip-MLP network: Strip-MLP-T* (light tiny), Strip-MLP-T (tiny), Strip-MLP-S (small), Strip-MLP-B (base), which have similar or even smaller model sizes, compared with MLP-baed models [7, 26, 27] and Swin Transformer [18], respectively. The hyper-parameters of the four variants of the models are as follows:

- Strip-MLP-T*: $C = 80, \{T_1 \sim T_4\} = \{2, 2, 6, 2\}$;
- Strip-MLP-T: $C = 80, \{T_1 \sim T_4\} = \{2, 2, 12, 2\}$;
- Strip-MLP-S: $C = 96, \{T_1 \sim T_4\} = \{2, 2, 18, 2\}$;
- Strip-MLP-B: $C = 112, \{T_1 \sim T_4\} = \{2, 2, 18, 2\}$;

where C means the channel number of the hidden layers in the first stage, and $T_1 \sim T_4$ is the number of times repeated for Strip Mixing Block and Channel Mixing Block in each of the four stages, as shown in Fig. 2.

4. Experiments

We compare our Strip-MLP with three main categories of networks: CNNs-based, Transformer-based, and MLP-based models. We first show the experimental settings of the datasets and implementation details. Then, the proposed method is compared with other popular methods on three datasets. Finally, we ablate the critical design components of Strip-MLP network.

4.1. Experimental Settings

Datasets. Methods are evaluated in four datasets which vary significantly in the number of training images:

Method	Params	FLOPs	Top-1(%)
CNNs-Based			
ResNet50 [8]	23.71M	4.12G	89.61
ResNet101 [8]	42.71M	7.85G	88.56
ResNet152 [8]	58.35M	11.58G	89.04
Transformer-Based			
ViT-B/16 [5]	85.85M	16.86G	53.96
Swin-T [19]	27.60M	4.36G	80.40
Swin-S [19]	48.91M	8.52G	79.83
Swin-B [19]	86.85M	15.14G	78.42
MLP-Based			
Wave-MLP-T [27]	16.73M	2.48G	86.89
Hire-MLP-T [7]	17.58M	2.16G	87.85
Strip-MLP-T*(**ours**)	17.67M	2.53G	**90.11**
Wave-MLP-S [27]	30.25M	4.55G	87.43
Hire-MLP-S [7]	32.65M	4.26G	88.48
Sparse-MLP-T [26]	23.55M	5.02G	90.54
Strip-MLP-T(**ours**)	24.33M	**3.67G**	**90.93**
Wave-MLP-M [27]	43.59M	7.93G	88.45
Hire-MLP-B [7]	57.77M	8.17G	88.62
Sparse-MLP-S [26]	47.87M	10.36G	91.07
Strip-MLP-S(**ours**)	43.66M	**6.83G**	**92.09**
Wave-MLP-B [27]	62.90M	10.27G	88.81
Hire-MLP-L [7]	95.03M	13.50G	88.19
Sparse-MLP-B [26]	65.07M	14.04G	91.67
Strip-MLP-B(**ours**)	58.49M	**9.22G**	**92.26**

Table 2. Image classification results of our Strip-MLP and other models on Caltech-101. *Top-1* denotes the Top-1 accuracy. Here, the patch number of CGSMM is C/2.

- Caltech-101 [6]: it is a benchmark dataset that includes 101 classes, totally around $9k$ images. We randomly take 80% of each class of the data as the training set and use the remaining data as the testing set.
- CIFAR-10 [14]: it consists of $60k$ 32×32 images in 10 classes, and there are $50k$ images
- CIFAR-100 [14]: it has 100 classes containing 600 images of each class (totally $60k$), and there are 500 training images and 100 testing images per class.
- ImageNet-1K [4]: it is an image classification benchmark containing $1.28M$ training images and $50k$ testing images of 1000 classes.

Training Details. In all of our experiments, the input image size is 224×224, and the input patch size is 4×4. Based on the deep-learning framework of PyTorch [21], we train our models from scratch without any extra data for pre-training except for the transfer learning experiments. We employ an AdamW [13] optimizer with 300 epochs for all training datasets and apply a cosine decay learning rate scheduler and 30 epochs of linear warm-up. In addition, we adopt the same augmentation and regularization strategies as to Swin-Transformer [19].

4.2. Image Classification on Small Datasets

We conduct experiments on Small Datasets of Caltech-101, CIFAR-10 and CIFAR-100. The results consistently indicate that our Strip-MLP efficiently improves the token

interaction power, and advances the MLP-based models' performance on the small dataset than other models.

Results on Caltech-101. Tab. 2 presents the image classification resluts of three types models on Caltech-101 dataset. Compared to ResNet [8], all the four variants of our Strip-MLP models achieve higher Top-1 accuracy with fewer parameters and FLOPs, demonstrating an average increase of +2.69% over three widely-used ResNet models.

For the Transformer-based models, *e.g.* Swin Transformer [19], Strip-MLP models noticeably surpass all variants models: +10.53%/12.26%/13.84% over Swin-Transformer (T/S/B) models. ViT-B/16 [5] only gets an accuracy of 53.96%, indicating that transformer-based models depend more on the scale of the datasets.

When compared to the MLP-based models, like Sparse MLP [26], Wave-MLP [27] and Hire-MLP [7], our models still achieve the best accuracy with fewer model parameters and FLOPs. For example, with fewer FLOPs (-4.28G), Strip-MLP-B (92.26%) has an increase of +3.64% Top-1 accuracy than Hire-MLP-B (88.62%).

Results on CIFAR-10. Tab. 3 shows the classification results of performance comparisons. Compared with CNNs-based models, all four variants of our Strip-MLP models achieve higher Top-1 accuracy with fewer parameters and FLOPs, with an average increase of +1.14% (98.06% vs. 96.92%). When compared with transformer-based models, our Strip-MLP achieves better performance by +3.06% (98.06% vs. 95.00%) with fewer parameters (34.49M vs. 54.37M) and FLOPs (5.56G vs. 9.34G). Compared with MLP-based models, all of our Strip-MLP models obtain the best Top-1 accuracy with fewer parameters and FLOPs. For example, the average Top-1 accuracy of Strip-MLP is higher than Hire-MLP/Wave-MLP/Sparse-MLP by +1.45%/0.69%/0.19% (98.06% vs. 96.61%/97.37%/97.87%) with fewer average parameters (34.49M vs. 50.71M/38.32M/45.43M) and fewer average FLOPs (5.56G vs. 9.36G/6.31G/9.81G)

In particular, with only 16.76M parameters and 2.53G FLOPs, our Strip-MLP-T* noticeably surpasses all variants of ResNet [8] and Swin-Transformer [19] models, and Strip-MLP-T* outperforms the transfer learning models of ViT-S/16 [5]/TST-14 [37] by +0.69%/+0.29%, consistently demonstrating our Strip-MLP has significant superiorities in performance and complexity.

Results on CIFAR-100. Tab. 4 compares the image classification results on CIFAR-100 with three kinds of architecture models. Compared with CNNs-based models [8], Strip-MLP obtains average higher accuracy of +2.05%~+2.43% with a smaller number of parameters and FLOPs. When compared to the self-attention based model, *e.g.* Swin Transformer [19], our model shows great superiority that Strip-MLP-B achieves 86.40% Top-1 accuracy, +9.64% than the Swin-B model with fewer pa-

Method	Params	FLOPs	Top-1 (%)
CNNs-Based			
ResNet50 [8]	23.53M	4.12G	97.06
ResNet101 [8]	42.52M	7.85G	96.70
ResNet152 [8]	58.16M	11.58G	96.99
Transformer-Based			
Swin-T [19]	27.53M	4.36G	95.25
Swin-S [19]	48.84M	8.52G	94.87
Swin-B [19]	86.75M	15.14G	94.87
MLP-Based			
Wave-MLP-T [27]	16.69M	2.48G	96.91
Hire-MLP-T [7]	17.53M	2.16G	96.20
Strip-MLP-T*(**ours**)	16.76M	2.53G	**97.79**
Wave-MLP-S [27]	30.20M	4.55G	97.27
Hire-MLP-S [7]	32.61M	4.26G	96.70
Sparse-MLP-T [26]	23.49M	5.02G	97.67
Strip-MLP-T(**ours**)	**23.07M**	**3.67G**	**97.94**
Wave-MLP-M [27]	43.55M	7.93G	97.56
Hire-MLP-B [7]	57.72M	8.17G	96.88
Sparse-MLP-S [26]	47.80M	10.36G	97.99
Strip-MLP-S(**ours**)	**41.81M**	**6.83G**	**98.23**
Wave-MLP-B [27]	62.83M	10.27G	97.72
Hire-MLP-L [7]	94.96M	13.50G	96.64
Sparse-MLP-B [26]	64.99M	14.04G	97.95
Strip-MLP-B(**ours**)	**56.33M**	**9.22G**	**98.28**

Table 3. Image classification results of different models on CIFAR-10. The patch number of CGSMM is C/4. All models are trained from scratch without any extra data.

Method	Params	FLOPs	Top-1(%)
CNNs-Based			
ResNet50 [8]	23.71M	4.12G	83.03
ResNet101 [8]	42.71M	7.85G	83.75
ResNet152 [8]	58.35M	11.58G	84.35
Transformer-Based			
Swin-T [19]	27.60M	4.36G	78.56
Swin-S [19]	48.91M	8.52G	78.10
Swin-B [19]	86.85M	15.14G	76.76
MLP-Based			
Wave-MLP-T [27]	16.73M	2.48G	82.77
Hire-MLP-T [7]	17.58M	2.16G	82.44
Strip-MLP-T*(**ours**)	17.67M	2.53G	**85.10**
Wave-MLP-S [27]	30.25M	4.55G	83.34
Hire-MLP-S [7]	32.65M	4.26G	82.94
Sparse-MLP-T [26]	23.55M	5.02G	84.19
Strip-MLP-T(**ours**)	24.33M	**3.67G**	**85.23**
Wave-MLP-M [27]	43.59M	7.93G	83.95
Hire-MLP-B [7]	57.77M	8.17G	83.17
Sparse-MLP-S [26]	47.87M	10.36G	84.27
Strip-MLP-S(**ours**)	43.66M	**6.83G**	**86.18**
Wave-MLP-B [27]	62.89M	10.27G	84.23
Hire-MLP-L [7]	95.03M	13.50G	83.53
Sparse-MLP-B [26]	65.07M	14.04G	84.46
Strip-MLP-B(**ours**)	58.49M	**9.22G**	**86.40**

Table 4. Image classification results of different models on CIFAR-100. The patch number of CGSMM is C/2.

Model	Dataset	Params	Top-1(%)
ViT-S/16 [5]		49M	97.1
TST-14 [37]		22M	97.5
ViP-S/7 [10]	CIFAR-10	25M	98.0
DynaMixer-S [35]		26M	98.2
Strip-MLP-T*(**ours**)		**18M**	**98.8**
ViT-S/16 [5]		49M	87.1
TST-14 [37]		22M	88.4
ViP-S/7 [10]	CIFAR-100	25M	88.4
DynaMixer-S [35]		26M	88.6
Strip-MLP-T*(**ours**)		**18M**	**89.4**

Table 5. The transfer learning results of the model pre-trained on ImageNet-1K and fine-tuned to CIFAR-10 and CIFAR-100.

rameters (58.49M vs. 86.85M) and lower FLOPs (9.22G vs. 15.14G). In addition, our models keep the superiority over the other MLP-based models in that Strip-MLP achieves the best Top-1 accuracy (with a higher range of +1.04%~+3.01%) compared to the MLP-based models, *e.g.* Wave-MLP [27], Hire-MLP [7], and Sparse MLP [26], with fewer parameters and FLOPs.

4.3. Transfer Learning on Small Datasets

Following the transfer learning experiment of DynaMixer [35], we show the advantages of our models on the transfer learning tasks. With the pre-trained Strip-MLP-T* model on the ImageNet-1K, we transfer the model to downstream datasets of CIFAR-10 and CIFAR-100. Compared to the previous models, such as ViT [5], ViP [10] and DynaMixer [35] models, our Strip-MLP-T* models achieve the best accuracies of 98.8% and 89.4% on two datasets, as shown in Tab. 5, +1.7%/0.6% than ViT-S/16 and DynaMixer-S on CIFAR-10, +2.3%/0.8% over ViT-S/16 and DynaMixer-S on CIFAR-100. More importantly, the number of parameters of our model is only 18M, which is far less than other models, indicating that our model is more efficient and effective.

4.4. Image Classification on ImageNet-1K

We conduct experiments on ImageNet-1K to show the effectiveness of the method. Tab. 6 shows the performance together with the complexity (in terms of FLOPS and the number of parameters) of three types of models. Strip-MLP outperforms the CNNs-based model: +0.5% with half the number of parameters (25M vs. 39M) and FLOPs (3.7G vs. 8.0G) for Strip-MLP-T over RegNetY-8G [22]. Compared to Transformer-based models, *e.g.* Swin Transformer [19], Strip-MLP achieves comparable results but with obvious superiority in terms of the number of parameters and FLOPs, such as Strip-MLP-B achieves similar performance to Swin-B but with fewer parameters (57M vs. 88M) and FLOPs (9.2G vs. 15.4G). Compared to the MLP-based models, Strip-MLP still shows the same efficiency in the various models. The Strip-MLP-T* achieves average higher accuracy by +2.12% than other MLP-based models (81.2% vs. 79.08%). Both Strip-MLP-B and Wave-MLP-B [27] models get the same accuracy of 83.6%. However, Strip-MLP-B uses fewer parameters (57M vs. 63M) and FLOPs (9.2G vs. 10.2G), indicating that the token-interaction power of our model is more efficient than other

Method	Params	FLOPs	Throughput (image/s)	Top-1 (%)
CNNs-Based				
ResNet50 [8]	26M	4.1G	-	78.5
ResNet101 [8]	45M	7.9G	-	79.8
RegNetY-4G [22]	21M	4.0G	1157	80.0
RegNetY-8G [22]	39M	8.0G	592	81.7
RegNetY-16G [22]	84M	16.0G	335	82.9
Transformer-Based				
DeiT-S [30]	22M	4.6G	940	79.8
DeiT-B [30]	86M	17.5G	292	81.8
Swin-T [19]	29M	4.5G	755	81.3
Swin-S [19]	50M	8.7G	437	83.0
Swin-B [19]	88M	15.4G	278	83.5
MLP-Based				
ResMLP-S12 [29]	15M	3.0G	1415	76.6
gMLP-S [16]	20M	4.5G	-	79.6
CycleMLP-B1 [1]	15M	2.1G	1038	78.9
Hire-MLP-Tiny [7]	18M	2.1G	1562	79.7
Wave-MLP-T [27]	17M	2.4G	1208	80.6
Strip-MLP-T*(**ours**)	18M	2.5G	814	**81.2**
ResMLP-S24 [29]	30M	6.0G	715	79.4
Sparse-MLP-T [26]	24M	5.0G	639	81.9
CycleMLP-B2 [1]	27M	4.0G	641	81.6
Hire-MLP-Small [7]	33M	4.2G	808	82.1
Wave-MLP-S [27]	30M	4.5G	720	82.6
Strip-MLP-T(**ours**)	25M	**3.7G**	597	82.2
CycleMLP-B4 [1]	52M	10.1G	321	83.0
Sparse-MLP-S [26]	49M	10.3G	361	83.1
Wave-MLP-M [27]	44M	7.9G	413	83.4
Strip-MLP-S(**ours**)	43M	**6.8G**	381	83.3
ResMLP-B24 [29]	116M	23.0G	231	81.0
gMLP-B [16]	73M	15.8G	-	81.6
CycleMLP-B5 [1]	76M	12.3G	247	83.2
Sparse-MLP-B [26]	66M	14.0G	278	83.4
Hire-MLP-Base [7]	58M	8.1G	441	83.2
Wave-MLP-B [27]	63M	10.2G	341	83.6
Strip-MLP-B(**ours**)	57M	9.2G	300	**83.6**

Table 6. Classification results of three kinds of models on ImageNet-1K without extra data. The group number of CGSMM is C/4. Throughput is tested on a single V100 GPU following [19].

MLP-based models. In addition, we apply the Strip MLP layer into Wave-MLP-T (namely Wave-Strip-MLP-T) on ImageNet-1K, and we set the strip width as 3. The Wave-Strip-MLP-T gets better results than the original Wave-MLP-T model: 81.2% vs. 80.6%. The performance improvement by +0.6% shows that our Strip MLP makes token interaction more efficient and can easily be used in other MLP models.

In conclusion, as shown in Fig. 4, our Strip-MLP has significant superiority in performance and complexity over other state-of-the-art methods on different datasets, demonstrating our Strip-MLP is remarkably efficient and effective to enrich the token interaction power for vision MLP.

4.5. Ablation Studies

To better show the effectiveness of the proposed method, we ablate the key components of the model design. Due to the limited GPU resources, we do ablation studies on the

Strip Width	Dataset	Params	FLOPs	Top-1(%)
1		41.33M	6.71G	91.38
3	Caltech-101	43.66M	6.83G	92.09
5		45.99M	6.96G	91.72
7		48.33M	7.08G	91.86
1		40.71M	6.71G	86.09
3	CIFAR-100	41.88M	6.83G	86.50
5		43.04M	6.96G	86.39
7		44.21M	7.08G	86.35

Table 7. Ablation study on the strip width of Strip MLP. We randomly select the model of Strip-MLP-S for Caltech-101 and Strip-MLP-S for CIFAR-100.

Dataset	Patches	Params	FLOPs	Top-1(%)
	C/1	47.22M	6.83G	91.81
	C/2	43.66M	6.83G	92.09
Caltech-101	C/4	41.88M	6.83G	91.38
	C/8	40.99M	6.83G	91.33
	1	40.16M	6.83G	90.93
	C/1	47.22M	6.83G	86.20
	C/2	43.66M	6.83G	86.18
CIFAR-100	C/4	41.88M	6.83G	86.50
	C/8	40.99M	6.83G	85.98
	1	40.16M	6.83G	85.03

Table 8. Ablation study about the patch number of the CGSMM. The base model is Strip-MLP-S. $C/1$ means splitting the feature C into C patches in channel dimension, namely $P = C/1 = C$.

datasets of both Caltech-101 and CIFAR-100.

The effect of Strip Width in Strip MLP. The strip width affects the token interaction ranges and determines the field of each row or column's contribution to adjacent tokens. We further conduct experiments to verify its effect by varying the width from 1 to 7 with a step at 2. Tab. 7 shows the performances of our models on Caltech-101 [6] and CIFAR-100 [14], respectively. When the width becomes larger, the performance increases and tends to saturate, indicating that Strip MLP layer improves the token interaction power. In all of our comparison experiments in previous sections, we consistently set the strip width as 3 according to the ablation experiment results.

Effects of Patch Number in CGSMM. Different patch number brings varying degrees of token interaction power improvement and affects the model performance. In Tab. 8, we design five decreased patch numbers from C into 1 to show the effectiveness of CGSMM. Without the patch splitting ($P = 1$) and GSML operation, the model performance decreases by 1.16% on Caltech-101 (92.09% → 90.93%) and decreases by 1.47% on CIFAR-100 (86.50% → 85.03%), which consistently proves the effectiveness of CGSMM in improving the token interaction power. From the experimental results, we can observe that the optimal patch number is different on the datasets with scale differences, so the optimal patch number should be determined through validation experiments. We consistently set the patch number as C/4 in other ablation studies.

Cascade vs. Parallel Architecture of GSML. Applying the GSML in a cascade architecture enables the token interacts with other tokens across the entire 2D space in just one
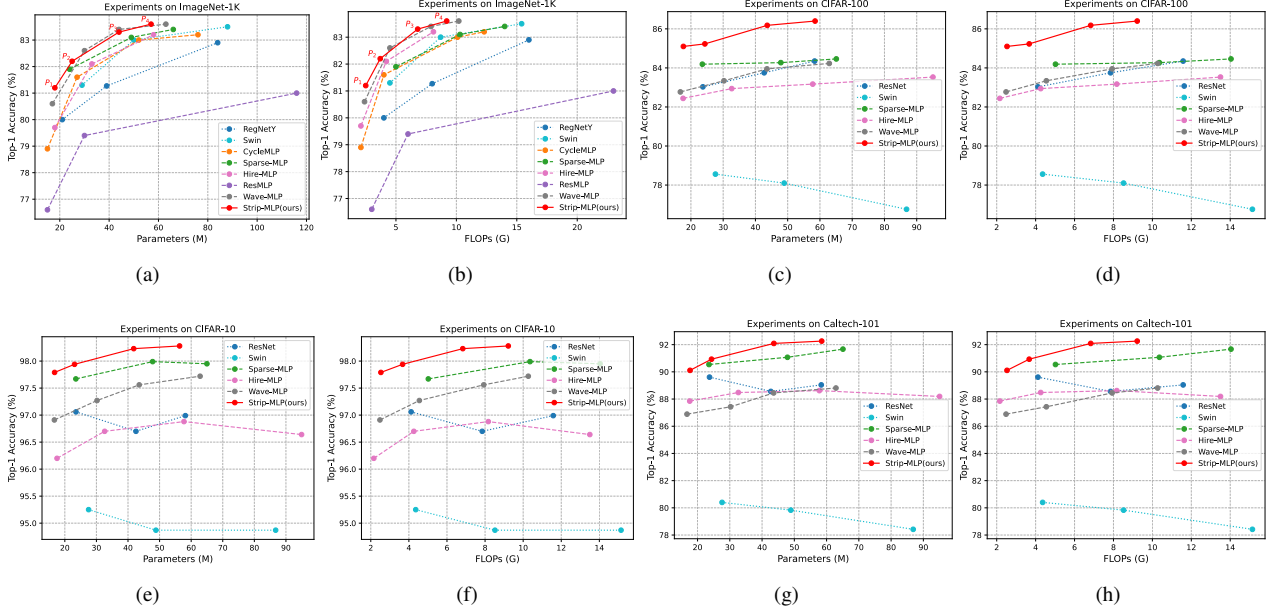
Figure 4. Performance comparison on four datasets of ImageNet-1K [4], CIFAR-100 [14], CIFAR-10 [14], and Caltech-101 [6]. The Top-1 accuracy, number of parameters and FLOPs are reported with networks of CNN-based, Transformer-based, and MLP-based models.

Method	Dataset	Params	FLOPs	Top-1(%)
Parallel	Caltech-101	42.67M	6.65G	91.61
Cascade		43.66M	6.83G	92.09
Parallel	CIFAR-100	40.90M	6.66G	85.89
Cascade		41.88M	6.83G	86.50

Table 9. Ablation study on the cascade and parallel structure of the Strip MLP in Group Strip MLP. *Parallel* means applying the Strip MLP on the row and column direction in parallel. *Cascade* means the proposed CGSMM method in this paper.

Method	Dataset	Params	FLOPs	Top-1(%)
CGSMM Only		43.40M	7.55G	91.10
LSMM Only	Caltech-101	43.96M	6.00G	90.31
CGSMM & LSMM		41.88M	6.83G	91.38
CGSMM Only		43.40M	7.55G	85.20
LSMM Only	CIFAR-100	43.96M	6.00G	86.13
CGSMM & LSMM		41.88M	6.83G	86.50

Table 10. Ablation study on the roles of two branches of CGSMM and LSMM. The base model is Strip-MLP-S. When any module branch has been removed, all input features would be fed into the remaining branch module.

module, which needs two modules for the parallel structure so that may decrease the efficiency of token interaction. In Tab. 9, we test the effects between the cascade structure of CGSMM and the parallel structure of **P**arallel **G**roup **S**trip **M**ixing **M**odule (PGSMM). Our experiments show that the cascade structure gets a higher accuracy compared to the parallel structure, with an increase of +0.48% and +0.61% on the Caltech-101 and CIFAR-100 datasets, respectively.

Roles of CGSMM & LSMM. To verify the significance of two complementary branches, we conduct experiments where we only keep one of the branches of CGSMM or LSMM. Tab. 10 displays the outcomes of our trial on Caltech-101 and CIFAR-100. Notably, we found that removing any module from either of the two branches significantly reduced the model's performance. These results highlight the essential role of both CGSMM and LSMM in enriching the tokens' interaction power.

5. Conclusion

This paper presents an efficient and effective token interaction MLP model of Strip-MLP for vision MLP. For *the*

Token's interaction dilemma problem, we design the Strip Mixing Block to enrich the token interaction power by three strategies. Our experimental analysis reveals that the Strip-MLP remarkably improves the performances of the MLP-based models on small datasets and keeps comparable or even better results on ImageNet with great superiorities on the number of parameters and FLOPs. These observations clearly demonstrate that our models make token interaction more efficient among MLP-based models in image classification tasks. It is expected that Strip MLP layer has the potential to be a standard layer in the future and have promising performance on diverse vision tasks.

References

[1] Shoufa Chen, Enze Xie, Chongjian Ge, Ding Liang, and Ping Luo. Cyclemlp: A mlp-like architecture for dense prediction. In *International Conference on Learning Representation (ICLR), Oral*, 2022.

[2] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

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

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

[6] Li Fei-Fei, Robert Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

[7] Jianyuan Guo, Yehui Tang, Kai Han, Xinghao Chen, Han Wu, Chao Xu, Chang Xu, and Yunhe Wang. Hire-mlp: Vision mlp via hierarchical rearrangement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 826–836, 2022.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[10] Qibin Hou, Zihang Jiang, Li Yuan, Ming-Ming Cheng, Shuicheng Yan, and Jiashi Feng. Vision permutator: A permutable mlp-like architecture for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[11] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[16] Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. *Advances in Neural Information Processing Systems*, 34:9204–9215, 2021.

[17] Ruiyang Liu, Yinghui Li, Linmi Tao, Dun Liang, and Hai-Tao Zheng. Are we ready for a new paradigm shift? a survey on visual deep mlp. *Patterns*, 3(7):100520, 2022.

[18] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12009–12019, 2022.

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[20] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[22] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10428–10436, 2020.

[23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[24] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[25] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[26] Chuanxin Tang, Yucheng Zhao, Guangting Wang, Chong Luo, Wenxuan Xie, and Wenjun Zeng. Sparse mlp for image recognition: is self-attention really necessary? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2344–2351, 2022.

[27] Yehui Tang, Kai Han, Jianyuan Guo, Chang Xu, Yanxi Li, Chao Xu, and Yunhe Wang. An image patch is a wave: Phase-aware vision mlp. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10935–10944, 2022.

[28] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34:24261–24272, 2021.

[29] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[30] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[31] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxim: Multi-axis mlp for image processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5769–5780, 2022.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[33] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020.

[34] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 568–578, 2021.

[35] Ziyu Wang, Wenhao Jiang, Yiming M Zhu, Li Yuan, Yibing Song, and Wei Liu. Dynamixer: a vision mlp architecture with dynamic mixing. In *International Conference on Machine Learning*, pages 22691–22701. PMLR, 2022.

[36] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. *arXiv preprint arXiv:2301.00808*, 2023.

[37] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 558–567, 2021.

[38] David Junhao Zhang, Kunchang Li, Yunpeng Chen, Yali Wang, Shashwat Chandra, Yu Qiao, Luoqi Liu, and Mike Zheng Shou. Morphmlp: A self-attention free, mlp-like backbone for image and video. *arXiv preprint arXiv:2111.12527*, 2021.

[39] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.

[40] Yipeng Zhang, Zahra M Aghajan, Matias Ison, Qiujing Lu, Hanlin Tang, Guldamla Kalender, Tonmoy Monsoor, Jie Zheng, Gabriel Kreiman, Vwani Roychowdhury, et al. Decoding of human identity by computer vision and neuronal vision. *Scientific reports*, 13(1):651, 2023.