Rethinking Token-Mixing MLP for MLP-based Vision Backbone

Tan Yu, Xu Li, Yunfeng Cai,
Mingming Sun, Ping Li
Cognitive Computing Lab,
Baidu Research
{tanyu01,lixu13,caiyunfeng,sunmingming01,liping11}@baidu.com

Abstract

By introducing the inductive bias from the image processing, convolution neural network (CNN) has achieved excellent performance in numerous computer vision tasks and has been established as *de facto* backbone. In recent years, inspired by the great success achieved by Transformer in NLP tasks, vision Transformer models emerge. Using less inductive bias, they have achieved promising performance compared with their CNN counterparts. More recently, researchers investigate in using the pure-MLP vision backbone to further reduce the inductive bias, achieving good performance. The pure-MLP backbone is built upon channel-mixing MLPs to fuse the channels and token-mixing MLPs for communications between patches. In this paper, we re-think the design of the token-mixing MLP. We discover that token-mixing MLP in existing MLP-based backbones is spatial-specific, and thus it is sensitive to spatial translation. Meanwhile, the channel-agnostic property of the existing token-mixing MLPs limits their capability in mixing tokens. To overcome these limitations, we propose a Circulant Channel-Specific (CCS) token-mixing MLP, which is spatial-invariant and channel-specific. It takes fewer parameters but achieves higher classification accuracy on ImageNet1K benchmark.

1 Introduction

Convolution neural network (CNN) [14, 23] has been *de facto* backbone for computer vision in the past years. Recently, inspired by the accomplishment achieved by Transformer [37] in NLP, several vision Transformer methods emerge [9, 35]. Compared with CNN methods, vision Transformer methods do not need hand-crafted convolution kernels and simply stack a few Transformer blocks. Albeit vision Transformer methods take much less inductive bias, they have achieved comparable or even better recognition accuracy than CNN models. More recently, researchers take a step forward. They propose MLP-based models [28, 33, 34] with only MLP layers taking even less inductive bias.

MLP-mixer [33] is the pioneering work of pure-MLP vision backbone. It is built on two types of MLP layers: channel-mixing MLP layer and token-mixing MLP layer. Channel-mixing MLP is for communications between the channels. In parallel, token-mixing MLP conducts communications between patches. Compared with vision Transformer adaptively assigning attention based on the relations between patches, token-mixing MLP assigns fixed attention to patches based on their spatial locations. In fact, the channel-mixing MLP is

Property	Reception	Spatial	Channel
Depthwise	local	agnostic	specific
Token-mixing	global	specific	agnostic
CCS	global	agnostic	group-specific

Table 1: The properties of depthwise convolution, token-mixing MLP in MLP-mixer [33] and our channel-specific circulant (CSC) token-mixing MLP.

closely related with depthwise convolution [4, 18, 20]. The differences between token-mixing MLP and depthwise convolution are three-fold. Firstly, the token-mixing MLP has a global reception field but the depthwise convolution has only a local reception field. The global reception field enables the token-mixer MLP to have access to the whole visual content in the image. Secondly, the depthwise convolution is spatial-invariant, whereas token-mixing MLP no longer possesses the spatial-invariant property and thus its output is sensitive to spatial translation. Lastly, for a specific position, the token-mixing MLP assigns an identical weight to elements in different channels. In contrast, the depthwise convolution applies different convolution kernels on different channels for encoding richer visual patterns.

Observing the strength and weakness of the vanilla token-mixing MLP in existing MLP-based backbones, we propose an improved structure, Circulant channel-specific (CCS) token-mixing MLP which preserves the strength of existing token-mixing MLP and overcomes its weakness. Similar to the vanilla token-mixing MLP, our CCS token-mixing MLP has a global reception field. Meanwhile, we adopt a circulant structure in the weight matrix to achieve the shift-invariant property. Moreover, we adopt a channel-specific design to exploit richer manners in mixing tokens. In Table 1, we compare the properties of our CCS with depthwise convolution and vanilla token-mixing MLP. Moreover, benefited from circulant structure, our CCS needs considerably fewer parameters than token-mixing MLP. To be specific, CCS only needs GN parameters where N is the number of patches and G is the number of groups. In contrast, token-mixing MLP normally takes N^2 parameters, which are significantly more than CCS since $G \ll N$. By replacing token-mixing MLP with the proposed CCS in two existing pure-MLP vision backbones, we achieve consistently higher recognition accuracy on ImageNet1K with fewer parameters.

2 Related Work

2.1 Vision Transformer

Vision Transformer (ViT) [9] is the pioneering work adopting the architecture solely with Transformer layers for computer vision tasks. It crops an image into non-overlap patches and feeds these patches through a stack of Transformer layers for attaining communications between patches. Using less hand-crafted design, ViT achieves competitive recognition accuracy compared with its CNN counterparts. Nevertheless, it requires a huge scale of images for pre-training. DeiT [35] adopts a more advanced data augmentation method as well as a stronger optimizer, achieving excellent performance through training on a medium-scale dataset. PVT [38] introduces a progressive shrinking pyramid into ViT, improving the recognition accuracy. PiT [16] integrates depthwise convolution between Transformer blocks and also devises a shrinking pyramid structure. In parallel, Tokens-to-Tokens ViT [42] effectively models the local structure through recursively aggregating neighboring tokens. It achieves higher recognition accuracy with fewer FLOPs. Transformer-iN-Transformer (TNT) [13]

also focuses on modeling the local structure. It devises an additional Transformer to model the intrinsic structure information inside each patch. Recently, the focus of vision Transformer is on improving efficiency through exploiting locality and sparsity. For instance, Swin [26] develops a hierarchical backbone with local-window Transformer layers with variable window sizes. Exploiting local windows considerably improves the efficiency, and the variable window scopes achieve the global reception field. Twins [5] alternately stacks a local-dense Transformer and a global-sparse Transformer, also achieving a global reception field in an efficient manner. Shuffle Transformer [19] also adopts the local-window Transformer and achieves the cross-window connections through spatially shuffling. S^2ViTE [3] explores on integrating sparsity in vision Transformer and improves the efficiency from both model and data perspectives. Recently, some works [11, 39] investigate combining convolution and Transformer to build a hybrid vision backbone. Multi-scale vision Transformer [10] builds a Transformer-base backbone for video recognition. Multi-view vision Transformer [2] utilizes Transformer for 3D object recognition.

2.2 MLP-based Backbone

Recently, MLP-mixer [33] proposes a pure-MLP backbone with less inductive bias than vision Transformer and CNNs, achieving excellent performance in image recognition. It is built upon two types of MLP layers: channel-mixing MLP and token-mixing MLP. Channel-mixing MLP is equivalent to 1×1 convolution layer. It achieves the communications between channels. The token-mixing MLP achieves cross-patch communications. It is similar to the self-attention block in Transformer. But the attention in Transformer is dependent on the input patches, whereas the attention in token-mixing MLP is agnostic to the input. Feed-forward (FF) [28] adopts a similar architecture as MLP-mixer, also achieving excellent performance. ResMLP [34] simplifies the token-mixing MLP in MLP-mixer and adopts a deep architecture stacking a huge number of layers. Meanwhile, to stabilize the training, ResMLP proposes an affine transform layer to replace the layer normalization in MLP-mixer. Benefited from exploiting a deeper architecture, ResMLP achieves a better performance than MLP-mixer. Meanwhile, by smartly trading off the hidden size and depth, ResMLP takes fewer FLOPs and fewer parameters than MLP-mixer. Recently, gMLP [25] exploits a gating operating to enhance the effectiveness of the token-mixing MLP, attaining a higher recognition accuracy than MLP-mixer. In parallel, External Attention [12] replaces the self-attention operation with the attention on external memory implemented by fully-connected layers, achieving comparable performance with vision Transformers. Spatial-shift MLP [40, 41] utilizes the spatial-shift operations for communications between patches.

3 Preliminary

In this section, we briefly review MLP-Mixer [33].

Input. For an input image of the size $W \times H \times 3$, we crop it into N non-overlap patches of $p \times p \times 3$ size. For each patch, it is unfolded into a vector $\mathbf{p} \in \mathbb{R}^{3p^2}$. In total, we obtain a set of patch feature vectors $\mathcal{P} = \{\mathbf{p}_0, \cdots, \mathbf{p}_{N-1}\}$, which are the input of the MLP-Mixer.

Per-patch fully-connected layer maps each patch feature $\mathbf{p}_i \in \mathcal{P}$ into a vector by

$$\mathbf{x}_i \leftarrow \mathbf{W}_0 \mathbf{p}_i + \mathbf{b}_0, \tag{1}$$

where $\mathbf{W}_0 \in \mathbb{R}^{3p^2 \times C}$ and $\mathbf{b}_0 \in \mathbb{R}^C$ are the weights of the per-patch fully-connected layer.

Mixer layers. MLP-mixer stacks L mixer layers of the same size, and each layer contains two MLP blocks: the token-mixing MLP block and the channel-mixing MLP block. Let us denote the patch features in the input of each mixer layer as $\mathbf{X} = [\mathbf{x}_0, \cdots, \mathbf{x}_{N-1}] \in \mathbb{R}^{C\times N}$ where C is the number of channels and N is the number of patches. Channel-mixing block projects patch features $\mathbf{X}$ along the channel dimension by

$$\mathbf{U} = \mathbf{U} + \mathbf{W}_2\sigma[\mathbf{W}_1\text{LayerNorm}(\mathbf{X})]. \tag{2}$$

$\mathbf{W}_1 \in \mathbb{R}^{rC\times C}$ represents weights of a fully-connected layer increasing the feature dimension from C to rC where $r > 1$ is the expansion ratio. $\mathbf{W}_2 \in \mathbb{R}^{C\times rC}$ denotes weights of a fully-connected layer decreasing the feature dimension from rC back to C. LayerNorm$(\cdot)$ denotes the layer normalization [1] widely used in Transformer-based models and $\sigma(\cdot)$ denotes the activation function implemented by GELU [15]. Then the output of the channel-mixing block $\mathbf{U}$ is fed into the token-mixing block for communications between patches:

$$\mathbf{Y} = \mathbf{U} + \sigma[\text{LayerNorm}(\mathbf{U})\mathbf{W}_3]\mathbf{W}_4, \tag{3}$$

where $\mathbf{W}_3 \in \mathbb{R}^{N\times M}$ and $\mathbf{W}_4 \in \mathbb{R}^{M\times N}$ denote the weights of fully-connected layers projecting patch features $\mathbf{U}$ along the token dimension. ResMLP [34] adopts a simplified block:

$$\mathbf{Y} = \mathbf{U} + \text{LayerNorm}(\mathbf{U})\mathbf{W}_3, \tag{4}$$

where $\mathbf{W}_3 \in \mathbb{R}^{N\times N}$ is a square matrix. ResMLP [34] shows that the simplified block achieves comparable performance as the original one defined in Eqn. (3). Unless otherwise specified, the token-mixing we mention below is the simplified form defined in Eqn. (4).

Classification head. N patches features from the last mixer layer are aggregated into a global vector through average pooling, which is fed into a fully-connected layer for classification.

4 Circulant Channel-Specific Token-mixing MLP

In this section, we introduce our circulant channel-specific (CCS) token-mixing MLP. As we mention in the introduction, the vanilla token-mixing MLPs in MLP-mixer and ResMLP are spatial-specific and channel-agnostic. The spatial-specific property makes the token-mixing MLP sensitive to spatial translation. Meanwhile, the channel-agnostic configuration limits its capability in mixing tokens. The motivation of designing CCS token-mixing MLP is to achieve the spatial-agnostic property and meanwhile attain a channel-specific configuration for mixing tokens in a more effective way. To this end, we make two modifications to the vanilla token-mixing MLP, as illustrated in details below.

4.1 Circulant structure

We devise the weight matrix $\mathbf{W}_3$ in Eqn. (4) in a circulant structure. To be specific, we set the weight matrix $\mathbf{W}_3$ in the below form:

$$\mathbf{W}_3 = \begin{bmatrix} w_0 & w_{N-1} & w_{N-2} & \cdots & w_2 & w_1 \\ w_1 & w_0 & w_{N-1} & \cdots & w_3 & w_2 \\ w_2 & w_1 & w_0 & \cdots & w_4 & w_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ w_{N-2} & w_{N-3} & w_{N-4} & \cdots & w_0 & w_{N-1} \\ w_{N-1} & w_{N-2} & w_{N-3} & \cdots & w_1 & w_0 \end{bmatrix}. \tag{5}$$

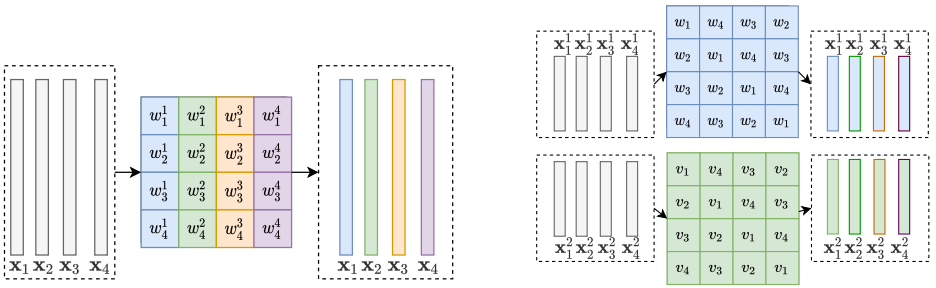
It is fully specified by one vector, $\mathbf{w} = [w_0, \cdots, w_{N-1}]$, which appears as the first column of $\mathbf{W}_3$. The circulant structure is naturally spatial-agnostic for mixing tokens. Below we show it in detail. Let us denote LayerNorm($\mathbf{U}$) in Eqn. (4) by $\hat{\mathbf{U}} = [\hat{\mathbf{u}}_0, \cdots, \hat{\mathbf{u}}_{N-1}]$, where each item $\hat{\mathbf{u}}_i$ ($i \in [0, N-1]$) is the patch feature after layer normalization. We rewrite Eqn. (4) into

$$\mathbf{Y} = \mathbf{U} + \hat{\mathbf{U}}\mathbf{W}_3. \tag{6}$$

Let us denote the i-th column of $\hat{\mathbf{U}}$ by $\hat{\mathbf{u}}_i$ and the i-th column of $\hat{\mathbf{U}}\mathbf{W}_3$ by $\bar{\mathbf{u}}_i$. Then it is straightforward to obtain that

$$\bar{\mathbf{u}}_i = \sum_{j=0}^{N-1} w_j \hat{\mathbf{u}}_{(i+j)\%N}, \forall i \in [0, N-1], \tag{7}$$

where % denotes the modulo operation. As shown in Eqn. (7), the mixing operation for ob-



(a) Vanilla token-mixing MLP. (b) Our CCS.

Figure 1: Comparisons between vanilla token-mixing MLP in existing MLP-based backbones and the proposed Circulant Channel-Specific (CCS) token-mixing MLP. The proposed CCS imposes a circulant-structure constraint on the weights of the token-mixing MLP. Meanwhile, our CCS splits the input $\mathbf{X}$ into multiple groups along the channel dimension. It uses different weight matrices on different groups for mixing tokens in a more flexible way.

taining each patch feature $\bar{\mathbf{u}}_i$ is invariant to the location i. Thus, the token-mixing MLP with the circulant-structure $\mathbf{W}_3$ is spatial-agnostic, which is not sensitive to spatial translation. Meanwhile, the circulant structure reduces the number of parameters from N^2 to N. At the same time, the multiplications between a N-dimension vector and an $N \times N$ circulant matrix only take $\mathcal{O}(N\log N)$ complexity using FFT, which is more efficient than the vanilla vector-matrix multiplication with $\mathcal{O}(N^2)$ computational complexity. Specifically, given an vector $\mathbf{x} \in \mathbb{R}^N$ and a circulant matrix $\mathbf{W}$ with the first column $\mathbf{w}$, the matrix-vector multiplication $\mathbf{W}\mathbf{x}$ can be computed efficiently through

$$\mathbf{x}\mathbf{W} = \text{FFT}[\text{IFFT}(\mathbf{x}) \odot \text{FFT}(\mathbf{w})], \tag{8}$$

where FFT denotes fast Fourier transform, IFFT denotes inverse fast Fourier transform, and $\odot$ denotes the element-wise product. Since both FFT and IFFT take only $\mathcal{O}(N\log N)$ computational complexity, the total complexity is only $\mathcal{O}(N\log N)$. In contrast, the vanilla token-mixing MLP in existing methods takes $\mathcal{O}(N^2)$ computational complexity. But when the number of patches, N, is not large, FFT with $\mathcal{O}(N\log N)$ complexity can not demonstrate its efficiency advantage over vanilla token-mixing MLP with $\mathcal{O}(N^2)$ complexity.

Algorithm 1 Pseudocode of our Circulant Channel-Specific (CCS) token-mixing MLP.

```
class CCS(nn.Module):
    def __init__(self, groups, patches):
        super().__init__()
        self.groups = groups
        self.w = nn.Linear(patches, self.groups).weight
    def forward(self, x):
        B, N, C = x.shape
        x = x.permute(0,2,1).contiguous()
        x = ifft(x)
        w = fft(self.w).unsqueeze(0).unsqueeze(1).expand(B,C//self
            .groups,self.groups,N).reshape(B,C,N)
        x = x.mul(w)
        x = fft(x).real
        x = x.permute(0,2,1).contiguous()
        return x
```

4.2 Channel-Specific settings

The token-mixing MLP in MLP-mixer [33] shares the same weight among different channels. That is, the same token-mixing MLP is applied to each of C channels in $\hat{\mathbf{U}} \in \mathbb{R}^{C\times N}$. A straightforward extension is to devise C separable MLPs, namely, $\{\text{MLP}_c\}_{c=1}^{C}$. For each slide of $\hat{\mathbf{U}}$ along the channel, $\mathbf{U}[c,:] \in \mathbf{R}^N$, it is processed by a specific MLP_c. This straightforward extension increases the number of parameters in vanilla token-mixing MLP of MLP-mixer from N^2 to N^2C. But the number of parameters, N^2C, is extremely huge, making the network prone to over-fitting. Thus, it does not bring noticeable improvements in the recognition accuracy, as claimed in the supplementary material of MLP-mixer [33].

In contrast, in our circulant-structure settings, even if we adopt C separable MLPs to process different channels of $\hat{\mathbf{U}}$, the number of parameters only increases from N to NC. The number of parameters is still on a medium scale. Thus, as shown in our experiments, using C separable circulant-structure MLPs can achieve better performance than that with a single circulant-structure MLP. To further reduce the number of parameters, we split C channels into G groups and each group contains C/G channels. For each group, we use a specific MLP. Thus, in this configuration, the number of parameters of our circulant-structure token-mixing MLP increases to GN, which is still significantly fewer than N^2 parameters in vanilla MLP-mixing MLP since $G \ll N$. It is worth noting that, the computational complexity of our CCS token-mixing MLP is irrelevant to the number of groups, G. Thus, using G MLPs in token-mixing does not bring more computational cost than using a single MLP. We visualize the architecture of the vanilla token-mixing MLP and our CCS token-mixing MLP in Figure 1. We compare our CCS with the vanilla token-mixing MLP in Table 2. As shown in the table, our CCS token-mixing MLP takes fewer parameters than the vanilla

	# of parameters	computational complexity
token-mixing	N^2	N^2C
CCS (ours)	GN	$N\log NC$

Table 2: Comparisons between our CCS and vanilla token-mixing MLP on the number of parameters and computational complexity. N is the number of patches, C is the number of channels, and G denotes the number of groups, where $G \ll N$.

token-mixing. We give the pseudocode of our CCS token-mixing MLP in Algorithm 1.

5 Experiments

Datasets. The main results are on ImageNet1K dataset [8]. It contains 1.2 million images from one thousand categories for training and 50 thousand images for validation. Note that MLP-mixer and ResMLP also exploit larger-scale datasets including ImageNet21K [8] and JFT-300M [30] datasets. Nevertheless, due to limited computing resources, it is impractical for us to train our models on these two datasets.

default settings	N	L	C	r	p	G	parameters
CCS-Mixer-B/16 [33]	196	12	768	4	16	8	57M
CCS-ResMLP-36 [34]	196	36	384	4	16	8	43M

Table 3: N is the number of tokens, L is the number of blocks, C is the hidden size, r is the expansion ratio, p is the patch size, and G denotes the number of groups in Table 2.

Settings. Our training follows the settings in DeiT [35]. To be specific, we adopt AdamW [27] as the optimizer with weight decay 0.05. A linear warm-up process is conducted at the beginning of training. The initial learning rate is 1e-3 and gradually drops to 1e-5 in a cosine-decay manner. We utilize multiple data augmentation approaches including random crop, random clip, Rand-Augment [7], Mixup [44] and CutMix [43]. Besides, we use replace dropout with DropPath [24] and conduct label smoothing [31] and repeated augmentation [17]. The whole training process takes 300 epochs. We integrate the proposed CCS token-mixing MLP in two types of mainstream MLP-based backbones including Mixer-B/16 [33] and ResMLP-36 [34]. Specifically, we only replace the vanilla token-mixing MLP with the proposed CCS token-mixing MLP and keep others unchanged. In Table 3, we give the detailed settings. Our CCS token-mixing MLP is implemented in PaddlePaddle platform developed by Baidu.

5.1 Main experimental results

The main experimental results are presented in Table 4. We compare ours with existing CNN-based models, Transformer-based models and MLP-based models. The first part of Table 4 contains the CNN-based models, including the classic ResNet-50 [14] and the recent state-of-the-art methods such as RegNetY-16GF [29], EfficientNet-B3 [32] and EfficientNet-B5 [32]. Note that, the architectures of RegNetY-16GF, EfficientNet-B3 and EfficientNet-B5 are all attained based on network architecture search (NAS). The main strength of EfficientNet is its excellent recognition accuracy with the extremely cheap computational cost.

The second group of methods is based on vision Transformer. ViT [9] is the pioneering work with architecture solely based on Transformer. We show its performance with extra regularization reported in MLP-mixer [33], which is better than the performance in the original ViT paper. DeiT [35] adopts more advanced optimizer and data augmentation approaches, achieving considerably better performance than ViT. The following works including TNT [13] and T2T [42] enhance the effectiveness of modeling local structure, achieving better performance. In parallel, PVT [38] and PiT [16] adopt a pyramid structure following CNNs, also attaining higher accuracy than DeiT and ViT. CPVT [6] improves the positional encoding. CaiT [36] investigates in deeper structure. Swin [26] exploits the locality, achieving higher efficiency. Container [11] proposes a hybrid architecture that exploits

Model	Resolution	Top-1 (%)	Top5 (%)	Params (M)	FLOPs (B)
CNN-based methods					
ResNet50 [14]	224×224	76.2	92.9	25.6	4.1
RegNetY-16GF [29]	224×224	80.4	–	83.6	15.9
EfficientNet-B3 [32]	300×300	81.6	95.7	12	1.8
EfficientNet-B5 [32]	456×456	84.0	96.8	30	9.9
Transformer-based methods					
ViT-B/16* [9, 33]	224×224	79.7	–	86.4	17.6
DeiT-B/16 [33]	224×224	81.8	–	86.4	17.6
PVT-L [38]	224×224	82.3	–	61.4	9.8
TNT-B [13]	224×224	82.8	96.3	65.6	14.1
T2T-24 [42]	224×224	82.6	–	65.1	15.0
CPVT-B [6]	224×224	82.3	–	88	17.6
PiT-B/16 [16]	224×224	82.0	–	73.8	12.5
CaiT-S32 [36]	224×224	83.3	–	68	13.9
Swin-B [26]	224×224	83.3	–	88	15.4
Nest-B [45]	224×224	83.8	–	68	17.9
Container [11]	224×224	82.7	–	22.1	8.1
MLP-based methods					
FF [28]	224×224	74.9	–	59	12
Mixer-B/16 [33]	224×224	76.4	–	59	12
Mixer-B/16*	224×224	77.2	92.9	59	12
S^2-MLP-wide [41]	224×224	80.0	94.8	71	14
CCS-Mixer-B/16*	224×224	79.8	94.6	57	11
ResMLP-36 [34]	224×224	79.7	–	44	8.9
ResMLP-36*	224×224	79.8	94.7	44	8.9
S^2-MLP-deep [41]	224×224	80.7	95.4	53	11
CCS-ResMLP-36*	224×224	80.6	95.3	43	8.9

Table 4: Comparisons with other models on ImageNet-1K benchmark without extra data. ViT-B/16* denotes the result of ViT-B/16 model reported in MLP-Mixer [33] with extra regularization. Mixer-B/16* and ResMLP-36* denotes the results of our implementation of Mixer-B/16 [33] and ResMLP-36 [34] based on settings in DeiT [33]. "+CCS" denotes replacing the vanilla token-mixing MLP with the proposed CCS token-mixing MLP.

both convolution and Transformer, achieving excellent accuracy and efficiency. Compared with the CNN-based methods, methods based on vision Transformer have achieved comparable or even better performance. Besides, Transformer-based methods need less inductive bias, which is more friendly to network architecture search (NAS). Compared with vision Transformer counterparts, our CCS-MLP network achieves comparable recognition accuracy using simpler architecture without the self-attention.

At last, we compare with MLP-based backbones including FF [28], MLP-mixer [33] and ResMLP [34] and S^2-MLP [41]. To make a fair comparison, we re-implement both MLP-mixer [33] and ResMLP [34] through the same settings as ours and DeiT. As shown in Table 4, using the same settings as ours and DeiT, Mixer-B/16* and ResMLP-36* achieve

better performance than that in the original papers, Mixer-B/16 [33] and ResMLP-36 [34]. By replacing vanilla token-mixing MLP with the proposed CCS token-mixing MLP, both MLP-mixer and ResMLP achieve higher recognition accuracy with fewer parameters. To be specific, using vanilla token-mixing MLP, Mixer-B/16* only achieves a 77.2% top-1 accuracy with 59M parameters. In contrast, using our CCS token-mixing MLP, we achieve a 79.8% top-1 accuracy with only 57M parameters. Besides, ResMLP-36* achieves a 79.8% top-1 accuracy with 44M parameters using vanilla token-mixing MLP. After using our CCS token-mixing MLP, the top-1 accuracy increases to 80.6% with only 43M parameters. Meanwhile, since the number of patches, $S = 196$, is not large. Using FFT for achieving the multiplication between a vector and a circulant matrix does not bring a considerable reduction in computation cost compared with the vanilla token-mixing MLP. Moreover, our CCS-MLP networks achieve a comparable accuracy as S^2-MLP with fewer parameters and FLOPS.

5.2 Ablation study

The influence of G. The core hyper-parameter in our CCS token-mixing MLP layer is G, the number of groups. We show the influence of G on ResMLP-36+CCS in Table 5. As shown in Table 5, when $G = 1$, it achieves a 80.0, which has outperformed the original ResMLP-36* [34], validating the effectiveness of using circulant structure only. Meanwhile, when G increases from 1 to 8, the recognition on ImageNet1K consistently improves. This validates the effectiveness of conducting different token-mixing operations on different groups of channels. Besides, when G further increases from 8 to 384, the recognition accuracy saturates but the number of parameters increases from 43M to 46M. Considering both effectiveness and efficiency, we set the number of groups, $G = 8$, by default.

G	1	4	8	384	ResMLP-36* [34]
top-1	80.0	80.4	**80.6**	**80.6**	79.8
top-5	95.0	95.1	**95.3**	**95.3**	94.7
parameters	43M	43M	43M	46M	44M

Table 5: The influence of the number of groups, G, on the recognition accuracy and the number of parameters. The experiments are conducted on ImageNet1K dataset.

Compared with convolution. To further demonstrate the effectiveness of the proposed CCS token-mixing MLP, we compare it with its CNN counterparts. To make a fair comparison, we replace the CCS token-mixing MLP with convolution layers and keep the other layers as the same. We first compare with the methods replacing the token-mixing MLP with a single depthwise convolution layer. Specifically, we compare with the depthwise convolution with different reception fields including 3×3 and 14×14. 3×3 depthwise convolution is widely used in efficient CNN neural networks and 14×14 depthwise convolution has a global reception field as ours. To make the number of patches after convolution identical to that before convolution, we conduct zero-padding for the depthwise convolution layers. To be specific, we add 2 rows/columns zero-padding for 3×3 convolution and 7 rows/columns zero-padding for 14×14 convolution. As shown in Table 6, our CCS-MLP outperforms these depthwise convolution layers. We further compare the methods replacing the token-mixing MLP with a stack of two pointwise convolution layers and a depthwise convolution layer and the spatial-shift MLP [41]. As shown in Table 6, our CCS-MLP achieves a comparable accuracy with considerably fewer parameters and FLOPs.

Settings	Top-1 Accuracy	Para.	FLOPs
Depthwise (3×3)	79.7	43M	8.4G
Depthwise (14×14)	80.2	45M	8.9G
Depthwise (3×3) + 2×Pointwise	80.5	53M	10.5G
Spatial-Shift + 2×Pointwise [41]	80.7	53M	10.5G
CCS-MLP	80.6	43M	8.9G

Table 6: Comparisons with CNN counterparts.

Transfer learning. We evaluate the transfer learning performance of the proposed CCS-MLP network. We fine-tune the model pre-trained on the large-scale ImageNet-1K dataset on target smaller-scale datasets including CIFAR10 [22], CIFAR100 [22] and Stanford Car [21] datasets. To be specific, both CIFAR10 and CIFAR100 contain $50,000$ images for training and Stanford Car contains only $8,144$ training images. We fine-tune the pre-trained model for 200 epochs on CIFAR10 and CIFAR100 datasets and $1,000$ epochs on Stanford Car dataset. We compare the proposed CCS-ResMLP-36 with DeiT-B [35], ResMLP-S24 [34] and ResMLP-36* [34]. Since ResMLP [34] does not report the transfer learning performance of ResMLP-36, we re-implement it using the same settings as ours. As shown in Table 7, without self-attention, our CCS-ResMLP-36 achieves comparable performance with Deit-B with fewer parameters and FLOPs. Meanwhile, using fewer parameters, our CCS-ResMLP-36 considerably outperforms ResMLP-36* on CIFAR100 and Stanford Car datasets.

method	CIFAR10	CIFAR100	Car	Para.	FLOPs
DeiT-B [35]	99.1	90.8	92.1	86.4M	17.6G
ResMLP-S24 [34]	98.7	89.5	89.5	29.5M	6.0G
ResMLP-36* [34]	98.7	88.5	91.0	44M	8.9G
CCS-ResMLP-36	98.9	90.0	92.5	43M	8.9G

Table 7: The performance of transfer learning on CIFAR10, CIFAR100 and Car. ResMLP-36* denotes the performance of fine-tuned ResMLP-36 with the same settings as ours.

6 Conclusion

The token-mixing MLP in the existing MLP-based vision backbone is spatial-specific and channel-agnostic. The spatial-specific configuration makes it sensitive to spatial translation. Meanwhile, the channel-agnostic property limits its capability in mixing tokens. To overcome these limitations, we propose a Circulant Channel-specific (CCS) token-mixing MLP, which is spatial-agnostic and channel-specific. The spatial-agnostic property makes our CCS token-mixing MLP more robust to spatial translation and the channel-specific property enables it to encode richer visual patterns. Meanwhile, by exploiting the circulant structure, the number of parameters in the token-mixing MLP layer is significantly reduced. Experiments on ImageNet1K dataset show that by replacing the existing token-mixing MLPs with our CCS token-mixing MLPs, we achieve higher recognition accuracy with fewer parameters.

References

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

[2] Shuo Chen, Tan Yu, and Ping Li. MVT: Multi-view vision transformer for 3d object recognition. In *Proceedings of the 32nd British Machine Vision Conference (BMVC)*, Virtual Event, UK, 2021.

[3] Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. *arXiv preprint arXiv:2106.04533*, 2021.

[4] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017*, pages 1800–1807, Honolulu, HI, 2017.

[5] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *arXiv preprint arXiv:2104.13840*, 2021.

[6] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.

[7] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, Miami, FL, 2009.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, Virtual Event, 2021.

[10] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. *arXiv preprint arXiv:2104.11227*, 2021.

[11] Peng Gao, Jiasen Lu, Hongsheng Li, Roozbeh Mottaghi, and Aniruddha Kembhavi. Container: Context aggregation network. *arXiv preprint arXiv:2106.01401*, 2021.

[12] Meng-Hao Guo, Zheng-Ning Liu, Tai-Jiang Mu, and Shi-Min Hu. Beyond self-attention: External attention using two linear layers for visual tasks. *arXiv preprint arXiv:2105.02358*, 2021.

[13] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *arXiv preprint arXiv:2103.00112*, 2021.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, 2016.

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

[16] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. *arXiv: 2103.16302*, 2021.

[17] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8126–8135, Seattle, WA, 2020.

[18] 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.

[19] Zilong Huang, Youcheng Ben, Guozhong Luo, Pei Cheng, Gang Yu, and Bin Fu. Shuffle transformer: Rethinking spatial shuffle for vision transformer. *arXiv preprint arXiv:2106.03650*, 2021.

[20] Lukasz Kaiser, Aidan N. Gomez, and François Chollet. Depthwise separable convolutions for neural machine translation. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[21] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[22] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1106–1114, Lake Tahoe, NV, 2012.

[24] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[25] Hanxiao Liu, Zihang Dai, David R So, and Quoc V Le. Pay attention to mlps. *arXiv preprint arXiv:2105.08050*, 2021.

[26] 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. *arXiv preprint arXiv:2103.14030*, 2021.

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, New Orleans, LA, 2019.

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

[29] Ilija Radosavovic, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10425–10433, Seattle, WA, 2020.

[30] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 843–852, Venice, Italy, 2017.

[31] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, Las Vegas, NV, 2016.

[32] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 6105–6114, Long Beach, CA, 2019.

[33] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. MLP-Mixer: An all-MLP architecture for vision. *arXiv preprint arXiv:2105.01601*, 2021.

[34] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, and Hervé Jégou. ResMLP: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404*, 2021.

[35] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 10347–10357, Virtual Event, 2021.

[36] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, Long Beach, CA, 2017.

[38] 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. *arXiv preprint arXiv:2102.12122*, 2021.

[39] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. CvT: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.

[40] Tan Yu, Xu Li, Yunfeng Cai, Mingming Sun, and Ping Li. S^2-MLPv2: Improved spatial-shift mlp architecture for vision. *arXiv preprint arXiv:2108.01072*, 2021.

[41] Tan Yu, Xu Li, Yunfeng Cai, Mingming Sun, and Ping Li. S^2-MLP: Spatial-shift mlp architecture for vision. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, HI, 2022.

[42] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token ViT: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

[43] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. CutMix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031, Seoul, Korea, 2019. IEEE.

[44] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[45] Zizhao Zhang, Han Zhang, Long Zhao, Ting Chen, and Tomas Pfister. Aggregating nested transformers. *arXiv preprint arXiv:2105.12723*, 2021.