ARE WE READY FOR A NEW PARADIGM SHIFT? A SURVEY ON VISUAL DEEP MLP

Ruiyang Liu*[1], Yinghui Li*[2], Linmi Tao§[1], Dun Liang*[1], Shi-Min Hu§[1], and Hai-Tao Zheng§[2]

[1]Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
[2]Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China

ABSTRACT

Multilayer perceptron (MLP), as the first neural network structure to appear, was a big hit. But constrained by the hardware computing power and the size of the datasets, it once sank for tens of years. During this period, we have witnessed a paradigm shift from manual feature extraction to the CNN with local receptive field, and further to the Transformer with global receptive field based on self-attention mechanism. And this year (2021), with the introduction of MLP-Mixer, MLP has re-entered the limelight and has attracted extensive research from the computer vision community. Compare to the conventional MLP, it gets deeper but changes the input from full flattening to patch flattening. Given its high performance and less need for vision-specific inductive bias, the community can't help but wonder, *Will deep MLP, the simplest structure with global receptive field but no attention, become a new computer vision paradigm*? To answer this question, this survey aims to provide a comprehensive overview of the recent development of deep MLP models in vision. Specifically, we review these MLPs in detail, from the subtle sub-module design to the global network structure. We compare the receptive field, computational complexity, and other properties of different network designs in order to understand the development path of MLPs clearly. The investigation shows that MLPs' resolution-sensitivity and computational densities remain unresolved, and pure MLPs are gradually evolving towards CNN-like. We suggest that the current data volume and computational power are not ready to embrace pure MLPs, and artificial visual guidance remains important. Finally, we provide our viewpoint about open research directions and potential future works. We hope this effort will ignite further interest in the community and encourage better visual tailored design for the neural network in the future.

***K*eywords** Computer Vision · Neural Network · Deep MLP · Paradigm Shift

1 Introduction

In computer vision, deep learning aims to imitate how the brain perceives and understands visual information using computational models of multiple processing layers. Multilayer perceptron (MLP) or fully connected (FC) networks are the first proposed and most classical type of neural networks, which are composed of multiple linear layers and nonlinear activations stacked together [1, 2]. To better explain the MLP, the researchers analyzed it through mathematical theory and concluded that **Multilayer feedforward networks are universal approximators** [3, 4]. Several results all show that multilayer perceptrons of different forms and complexity can approximate arbitrarily well a continuous function, provided that an arbitrarily large number of units is available [5, 6, 7, 8, 9]. Although the theory is beautiful, the practice is quite hard. Multilayer perceptron was among the first classifiers tested on MNIST [10, 11, 12], which is the most widely used computer vision benchmark during that period. Classification is done by flattening the 28×28 image into a one-dimensional vector by rows, which is considered as the initial nodes, and then passing through a stack of fully connected layers. Most had few layers or few artificial neurons per layer, but apparently back then they were the biggest feasible MLPs, trained when CPU cores were at least hundreds or thousands of times slower than today, not to mention

*{lry20, liyinghu20, liangd16}@mails.tsinghua.edu.cn
§{linmi, shimin}@tsinghua.edu.cn, zheng.haitao@sz.tsinghua.edu.cn

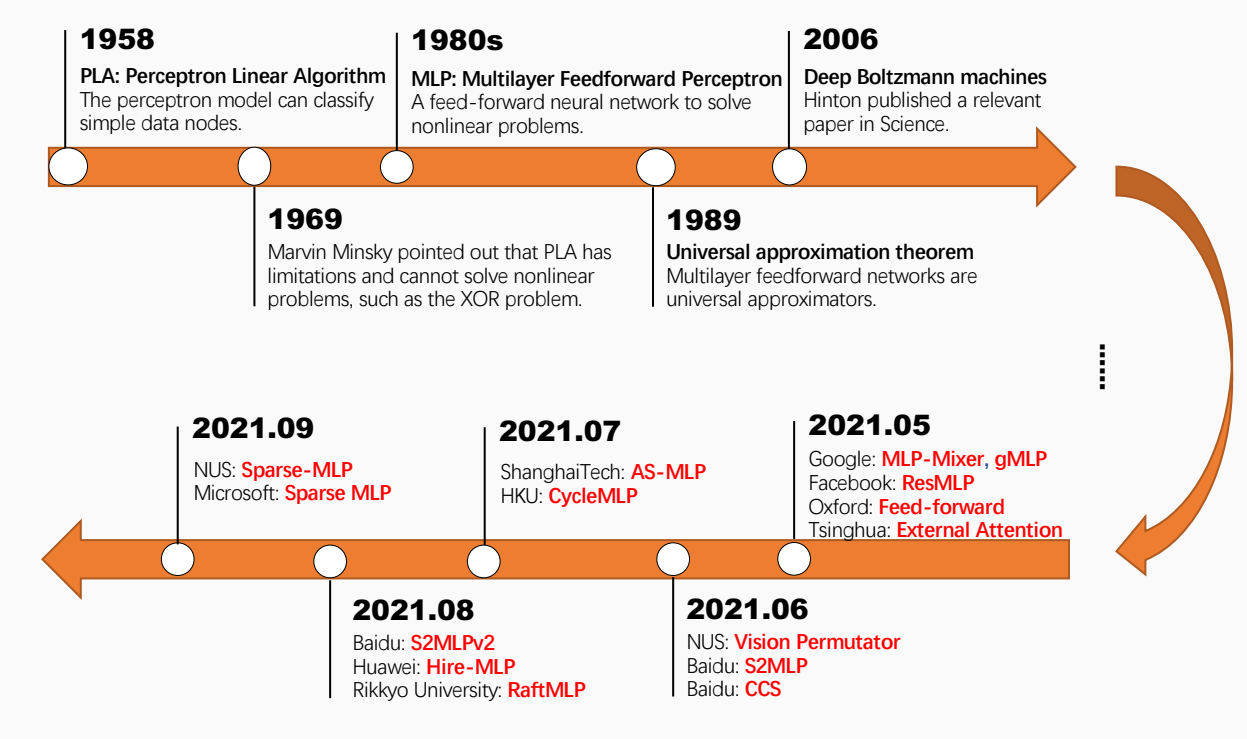


Figure 1: Key milestones in the development of MLPs. The new vision MLP models are marked in red.

GPU and TPU. Through practice, it had been found that MLP is computationally intensive and prone to overfitting when the amount of data is insufficient. Hinton proposed pretraining and fine-tuning to create a deep autoencoder to solve the problems [13]. Nevertheless, input flattening is still a pain for MLPs and limits the input resolution. Still limited by the hardware and datasets available at the time, MLP did not see a spring in its step.

Just a few years later, the availability of larger datasets (e.g., ImageNet [14]) coupled with increased computational capacity leads to the first **paradigm shift**, and the convolutional neural networks (CNNs) [15, 16, 17, 18, 19, 20, 21, 22, 23, 24] take the stage. Convolutional neural networks replace the hand-designed choices from hard-wired features with flexible and trainable architectures. The inductive biases inherent to CNNs, such as translation invariance and local connectivity, help a lot in image feature extraction. Undeniably, the local connectivity is still a compromise for computing capability. While CNNs have been the de-facto standard for computer vision, the introduction of Vision Transformer [25] (ViT) broke this legend in 2020, and a new round of **paradigm shift** ushered in. One stone stirs up a thousand waves! Based on self-attention layers [26], ViT [25] and its variants [27, 28, 29, 30, 31, 32, 33, 34, 35] introduce global receptive field to computer vision, and their strong performance on image classification benchmarks has prompted significant interest from the vision community. Feature extraction is no longer artificially designed to be performed on a local scale, but is considered globally. This allows the model to encode object-to-object interrelationships. Within these paradigm shifts, there is a gradual reduction in human intervention. But there is no such thing as a free lunch. Compared with CNNs, Vision Transformer has a larger computational cost and requires more data for training.

In the first week of May 2021, when the Transformer craze is not over yet, MLP makes a comeback, with more hidden layers and compromise input flattening. In particular, researchers from four different institutions: Google [36], Oxford University [37], Tsinghua University [38, 39], and Facebook [40], all question at almost the same time: *is the convolution layer or attention layer even necessary*? Or, *is the current ready for a new paradigm shift*? By simply stacking a series of fully connected layers applied over the patch and feature dimensions in an alternating fashion, researchers obtain a performance that is only a few points weaker than CNNs and ViT on ImageNet [36, 37, 40]. These pure deep MLP-based architectures retain the global receptive field, introduce a few inductive biases[1], and relies further

[1]The inductive bias of a learning algorithm is the set of assumptions that the learner uses to predict outputs of given inputs that it has not encountered [41]. Such as the translation invariance and local connectivity in CNNs. Vision Transformers removes these several hard decisions, but using the similarity of key and query as attention weights is still an inductive bias. Deep MLP further removes these assumptions and allows the network to learn the weights autonomously from the raw data.

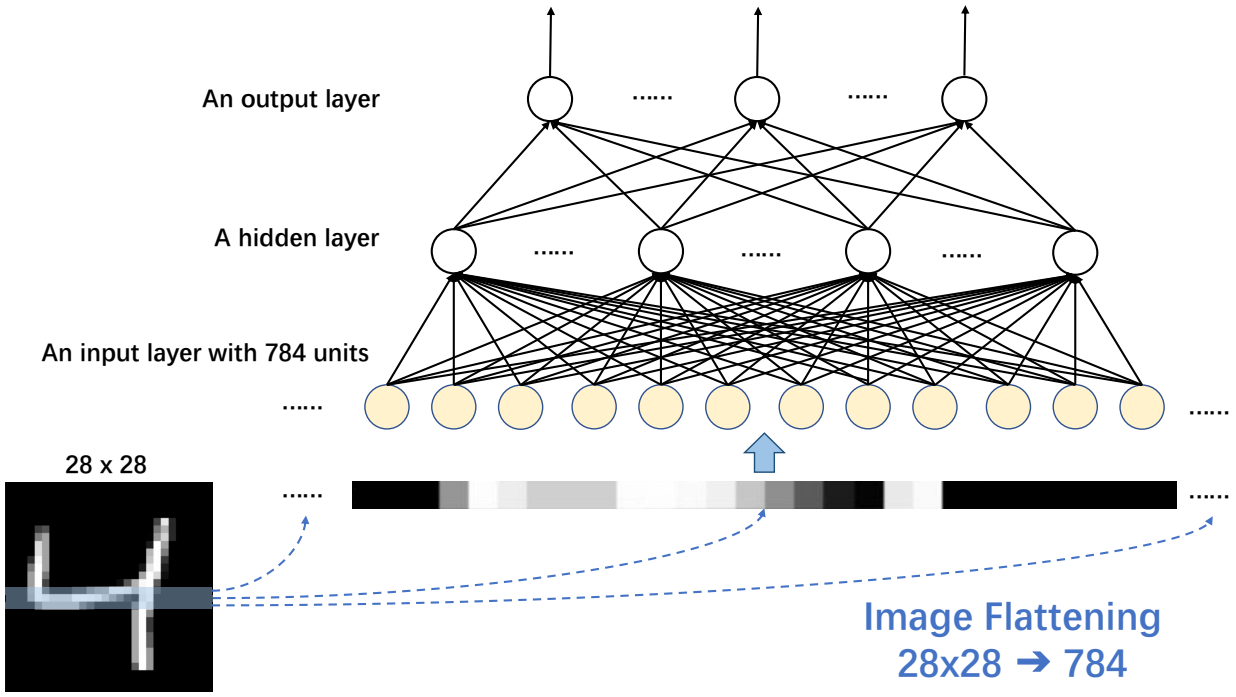


Figure 2: Structure of conventional MLP. For better visualization, only artificial neurons are shown.

on learning from raw data. Figure 1 shows the development timeline of MLPs. The rapid increase in the number of visual deep MLP models seems to suggest that a new **paradigm shift** has arrived. Is this true?

To answer this question, this survey focuses on providing a comprehensive overview of the latest visual deep MLP models and clarifying the recent development of deep MLPs. Specifically, we first review conventional MLPs and provide a brief overview of paradigm shifts in the last decade, which we hope be helpful for understanding the latest network designs (Section 2). Following closely, we review the design of the latest pioneering models, and describe the differences and connections between MLP, convolution and self-attention mechanisms, and present the bottlenecks and challenges faced by the pure deep MLP architecture (Section 3). Then we carefully investigate and compare the block designs of various deep MLP variants, analyzing their similarities and differences in terms of design implementation, computational complexity, and receptive field (Section 4). From the inside out, we analyze the development of block stacking from a macroscopic perspective and divide it into the single-stage, two-stage, and pyramid types. And the performance of different MLP-based, CNN-based, and Transformer-based models is also compared and discussed (Section 5). With a thorough understanding of the latest developments in deep MLP, we find that MLPs' resolution-sensitivity and computational densities remain unresolved, and pure deep MLPs are gradually evolving towards CNN-like. *We are not ready for deep MLPs as a new paradigm shift yet.* This promotes us to revisit the importance of artificial visual tailored design for current neural networks and suggests a way forward for future development (Section 6). The specific details of CNNs and Vision Transformer will not be developed in this article, but you can learn more from the CNN survey [42, 43] and Vision Transformer survey [44, 45, 46].

2 Preliminary

Histories make men wise. We begin with a brief introduction to conventional MLP networks, and then review the history of paradigm shifts during the decade of MLP silence. We hope this brief review will provide a useful insight into the deep MLP in subsequent sections.

2.1 Conventional Multilayer Perceptron

Multilayer perceptron mainly originated in the 1980s. As shown by Cybenko's theorem [6], MLPs are universal function approximators and can be used to create mathematical models by regression analysis. Since classification is a particular case of regression when the response variable is categorical, MLPs make good classifier algorithms. A conventional

MLP consists of three or more layers: an input layer, one or multiple hidden layers, and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. The bottom layer that takes input from datasets is called the input layer (or visible layer), and it is the exposed part of the network. The required tasks, such as prediction and classification, are performed by the output layer. An arbitrary number of hidden layers placed between the input and output layers is the true computational engine of the MLP. Since the input of a conventional MLP is a one-dimensional vector, a two-dimensional input image is usually flattened by row or by column, as shown in Figure 2.

As the name implies, the main feature of the fully connected layer lies in the full connectivity, which is different from the local connectivity of the convolution layer. The most notable problem that follows is sensitivity to input resolution. Image flattening makes the input unit fixed, and does not meet the requirement for resolution insensitivity in downstream tasks. Although in the past this was not important, today it is well worth considering. Another significant problem with full connectivity is the huge parameter number and computational cost, which grows quadratically with the image resolution. This is the biggest obstacle preventing MLPs from moving forward and towards practical applications. The third problem is that the network cannot be too deep, otherwise the gradient will disappear. All these issues have temporarily sunk the MLP.

2.2 Paradigm Shifts in The Last Decade

Deep learning never stops with MLP stagnation. In 2009, ImageNet [14], a large image dataset, was proposed to provide raw materials for the network to learn image features autonomously. Three years later, AlexNet [15] won the ImageNet competition, which is a symbolically significant event of the first paradigm shift. CNN-based architectures have gradually been utilized to automatically extract image features instead of hand-crafted features. In traditional computer vision algorithms, features such as gradient, texture, and color are extracted locally. Hence, the inductive biases inherent to CNNs, such as translation invariance and local connectivity, help a lot in image feature extraction. And the computational cost due to the local connectivity of CNN is within the tolerance range in computing at that time. On the one hand, the computing power keeps increasing, on the other hand, network regularization techniques [47, 48] and residual structures [18] have been proposed to significantly alleviate the gradient disappearance problem, both fueling the network with more and more layers. The development of self-supervision [49, 50, 51], as well as training strategies [52, 53, 54, 55, 56], has fueled the continuous improvement of deep CNN performance. In addition to classification, these deep CNNs outperform traditional algorithms for almost all computer vision tasks such as object detection [57, 58], segmentation [59, 60], demosaicing [61], super-resolution [62, 63, 64], deblurring [65], and so on. *CNN is the de-facto standard for computer vision* has become the consensus of the vision community.

Keeping pace with Moore's law [66], the computer capability has increased steadily with each new generation of chips. This encourages researchers to try designs with greater computational cost, and the natural language processing (NLP) community pioneers the Transformer. In detail, Transformer introduces the concepts of key, query, and value, uses the similarity matrix of key and query as the weight of the global receptive field, which is so-called the attention mechanism, and finally weights summation of value to obtain the new encoded features. In 2020, visual researchers note the application and success of the Transformer [26] in NLP and are surprised to realize that it is time to move beyond the limits of local connectivity and towards global connectivity. ViT [25] is the first one to promote the research on Transformer in the field of vision. It uses a pure Transformer framework to extract visual features, where an image is divided into 16×16 patches, and the convolution layer is completely abandoned. It shows that the Transformer-based architecture can perform well in large-scale datasets (e.g., JFT-300M [67]). Since then, the new paradigm is widely accepted in the vision community, and numerous works have improved on Transformer-based architectures, from block designs to self-learning methods [68]. Not only for tasks such as objection detection [27] and segmentation [27, 69, 70] Transformer-based achieves better performance than CNN-based, but also achieves state-of-the-art performance on denoising [71], deraining [71], super-resolution [71, 72, 73, 74], etc. Furthermore, there is a line of works [75, 76, 77, 78] showing that the Transformer-based architecture has better robustness than CNN-based. All the developments over the year indicate that the paradigm of computer vision is shifting from CNN to Transformer.

The success of Vision Transformer marks the paradigm shift to the era of the global receptive field in computer vision. Can we further abandon the artificially designed attention mechanism and let the model learn the weights autonomously from the raw data? This motivation reminds the researchers of that long-dusted simplest structure, the MLP. After a long period of slumber, MLP finally wakes up in May 2021, when the first deep MLP, called MLP-mixer [36], is launched.

3 Pioneering Model and New Paradigm

In this section, we detailedly review the structure of the latest so-called pioneering MLP model, MLP-Mixer [36], and followed by a brief review of the contemporaneous ResMLP [40] as well as the Feed-forward [37]. After that, we

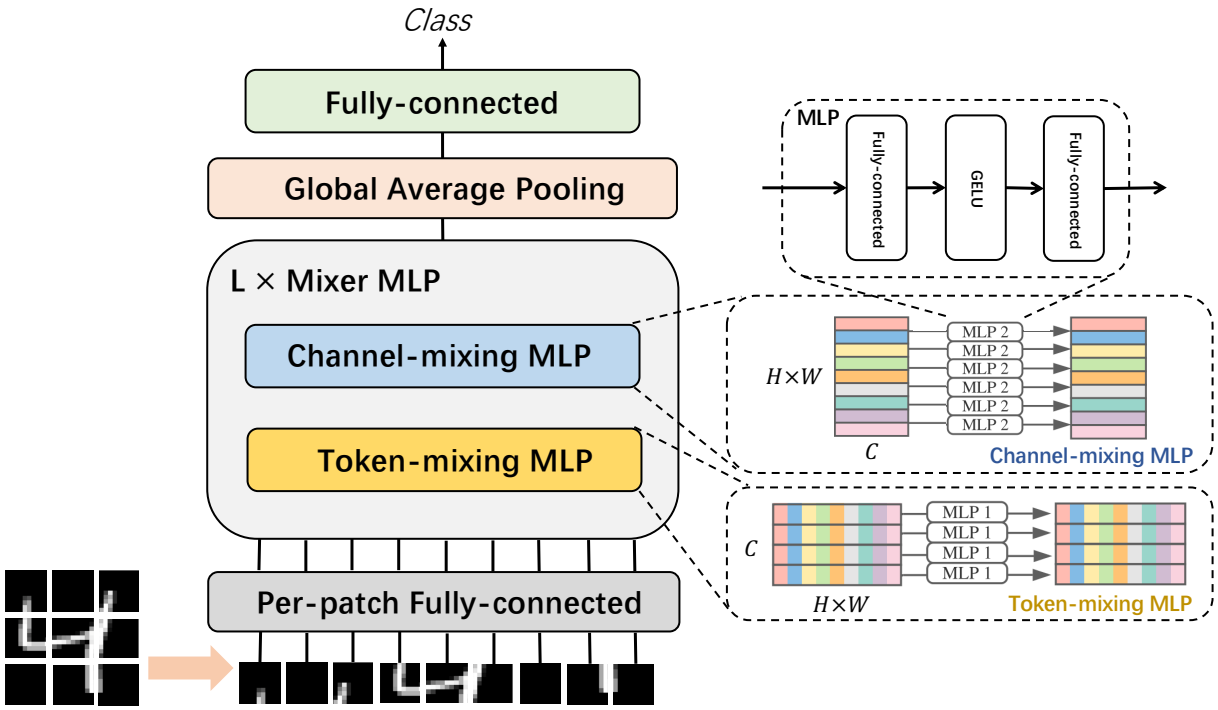


Figure 3: MLP-Mixer structure. To simplify the representation, we omit the Layer Normalization and the residual path.

strip the new paradigm, MLP, from the network and elaborate its differences and connections with convolution and self-attentive mechanisms. Finally, we explore the bottlenecks of MLP and lay the foundation for the introduction of subsequent variants.

3.1 Structure of Pioneering Model

MLP-Mixer [36] is the first proposed visual deep MLP network, also known as a pioneering MLP model by the vision community. Compare to the conventional MLP, it gets deeper but changes the input from full flattening to patch flattening. In detail, the MLP-Mixer consists of three modules, including a per-patch fully-connected layer, a stack of N Mixer layers, and a fully-connected layer for classification, as shown in Figure 3.

Patch Embedding and Classification Header. The per-patch fully-connected layer and classifier of MLP-Mixer are exactly the same as ViT [25]. For an input image of the size $W \times H \times 3$, we crop it into S non-overlap patches of $p \times p \times 3$ size. The patch is also called as token. 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 $\mathcal{P} = \{\mathbf{p}_1, \cdots, \mathbf{p}_S\}$, which are the input of the MLP-Mixer. For each patch feature $\mathbf{p}_i \in \mathcal{P}$, the per-patch fully-connected layer maps it into a C-dimensional embedding vector. Intuitively, such an embedding layer is exactly equivalent to a convolution layer with kernel size equal to patch size and stride equal to patch size. As for classification, the final feature vectors are aggregated into a global vector through an average pooling. Then the global vector is fed into a fully-connected layer for results.

Mixer layers. MLP-mixer stacks L mixer layers of the same size, and each layer contains two MLP blocks: the Token-mixing MLP and the Channel-mixing MLP, as shown in Figure 3. Let us denote the patch features in the input of each mixer layer as $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_S] \in \mathbb{R}^{C \times S}$, where C and S are the number of channels and patches. The Token-mixing MLP acts on columns of $\mathbf{X}$ (i.e. it is applied to a transposed input table $\mathbf{X}^\top$), maps $\mathbb{R}^S \mapsto \mathbb{R}^S$, and is shared across all columns. The Channel-mixing MLP acts on rows of $\mathbf{X}$, maps $\mathbb{R}^C \mapsto \mathbb{R}^C$, and is shared across all rows. And each MLP consists of two fully-connected layers and a GELU [79] activation function placed between them (the upper right corner of Figure 3). Thus mixer layer can be written as follows (omitting layer indices):

$$\begin{aligned} \mathbf{U}_{*,i} &= \mathbf{X}_{*,i} + \mathbf{W}_2 \sigma\left(\mathbf{W}_1 \operatorname{LayerNorm}(\mathbf{X})_{*,i}\right), \quad \text{for } i = 1 \ldots C, \\ \mathbf{Y}_{j,*} &= \mathbf{U}_{j,*} + \mathbf{W}_4 \sigma\left(\mathbf{W}_3 \operatorname{LayerNorm}(\mathbf{U})_{j,*}\right), \quad \text{for } j = 1 \ldots S. \end{aligned} \tag{1}$$

Here σ is an element-wise nonlinearity (GELU [79]), and $\operatorname{LayerNorm}(\cdot)$ denotes the layer normalization [48] widely used in Transformer-based models. $\mathbf{W}_3 \in \mathbb{R}^{rC \times C}$ represents weights of a fully-connected layer increasing the feature

Table 1: Comparion between separable convolution, self-attention, and Token-mixing MLP.

Operation	Weights	Receptive Field	Scale Variable	Spatial	Channel	Params	FLOPs
Convolution	learned	local	True	agnostic	specific	k^2C^2	$\mathcal{O}(HWC^2)$
Separable Convolution	learned	local	True	agnostic	specific	k^2C	$\mathcal{O}(HWC)$
Self-Attention	data-driven	global	True	agnostic	specific	$3C^2$	$\mathcal{O}(H^2W^2C)$
Token-mixing MLP	learned	global	False	specific	agnostic	H^2W^2	$\mathcal{O}(H^2W^2C)$

dimension from C to rC where $r > 1$ is the expansion ratio. And $\mathbf{W}_4 \in \mathbb{R}^{C\times rC}$ denotes weights of a fully-connected layer decreasing the feature dimension from rC back to C. Usually r is taken as 4. It is worth mentioning that, each mixer layer takes an input of the same size, which is most similar to Transformers, or deep RNNs in other domains. However, it unlikes most CNNs, which have a *pyramidal* structure: deeper layers have a lower resolution input, but more channels. In addition, MLP-Mixer does not use position embeddings because the token-mixing MLPs are sensitive to the order of the input tokens.

Compared to MLP-Mixer, Feed-forward (FF) [37] and ResMLP [40] are put on arXiv[2] a few days later. Feed-forward [37] adopts essentially the same structure as the MLP-Mixer, just swaps the order of Channel-mixing MLP and Token-mixing MLP, and is not repeated here. As another contemporaneous work, ResMLP [40] simplifies the Token-mixing MLP in MLP-Mixer from two layers to one. Meanwhile, ResMLP proposes an affine transform layer to replace the layer normalization in MLP-Mixer to stabilize the training.

Experimentally, these three pure MLP models are found to achieve comparable performance to CNN and ViT for image classification (Table 4). These empirical results significantly break past perceptions, challenge the necessity for convolution layer as well as attention layer, and prompt the community to rethink is the convolution layer or attention layer necessary. This further induces us to explore whether a pure MLP stack, i.e., the mixer layers, will become the new paradigm for computer vision.

3.2 New Paradigm: From Convolution and Self-Attention to MLP?

In order to find out whether pure MLP stacked into a mixer layer will become a new paradigm, it is necessary to first delineate it from convolution and self-attentive mechanisms.

There is an indisputable fact that the Channel-mixing MLP in MLP-Mixer, as well as the MLP in ViT, is just a 1×1 convolution commonly used in CNNs, which allows communication between different channels. The core points we need to compare come naturally to Token-mixing MLP, self-attention, and convolution, which allow communication between different spatial locations. A detailed comparison between the three is shown in Table 1, and we will describe them in detail later on.

In fact, the feedforward layer over the patch dimension (Token-mixing MLP) can be viewed as an unusual type of convolution with a full receptive field and a single channel, and it shares the same kernel (of full receptive field) for all of the channels. This is very close to the separable convolutions [80] used in CNN, which apply convolutions to each channel independently of the other channels. However, in separable convolutions, the convolution kernels are usually small and a different convolutional kernel is applied to each channel. What's more, the global receptive field makes Token-mixing MLP spatially sensitive, while the local receptive field makes the separable convolution translation invariant. In some sense, the self-attentive mechanism can also be seen as a separable convolution operation with a global receptive field, although the convolution kernel is similarity matrices rather than learnable parameters. For this reason, the self-attentive layer is also insensitive to space. Thus we know that, the global receptive field weights in the MLP are location-dependent, while the global receptive field weights in the attention layer are input-dependent. For a fair comparison, we perform spatial flattening of the CNN feature maps. And the comparison of the three at the convolution kernel level is shown in Figure 4 [81].

What does the new paradigm bring to the learning weights? Figure 5 displays the visualization of the linear interaction layers in MLP-Mixer and ResMLP trained on ImageNet or JFT-300M. As seen from the results of MLP-Mixer, as the amount of data increases, the percentage of low-frequency filters increases, and the weights become more sparse. ResMLP's shallow filters also show local connectivity properties. These fully connected layers actually still do local computation, which can be replaced by convolution layers. Thus, we conclude that **The patterns in the shallow linear**

[2]arXiv is an open-access repository of electronic preprints and post-prints approved for posting after moderation, but not peer review. The standard access route is through the `arXiv.org` website.

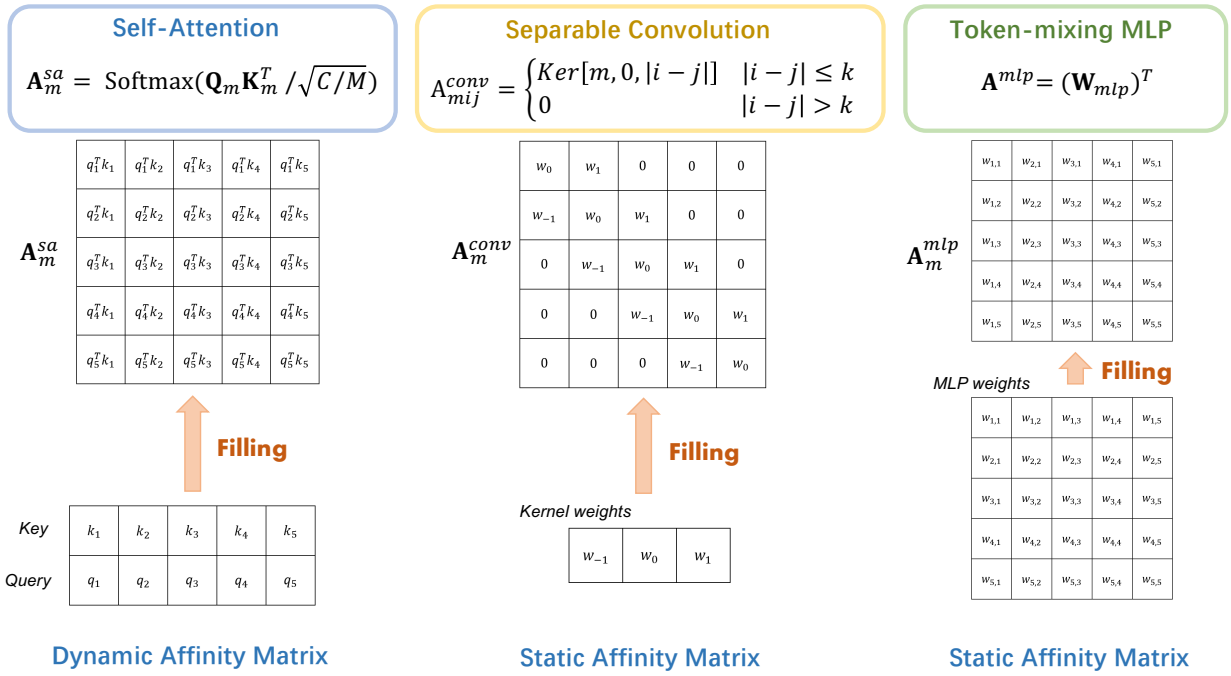


Figure 4: Comparion between self-attention, separable convolution, and Token-mixing MLP.

layers share similarities with convolutions. As the number of layers deepens, the effective range of receptive field increases and the weights become disorganized. It is unclear if this is due to a lack of training data or if it should be so. Notably, while MLP-Mixer and ResMLP share a highly similar structure (except for the normalization layer), their learning weights are vastly different. This puts a question mark on whether MLP is learning generic visual features. The interpretability of MLPs is far behind.

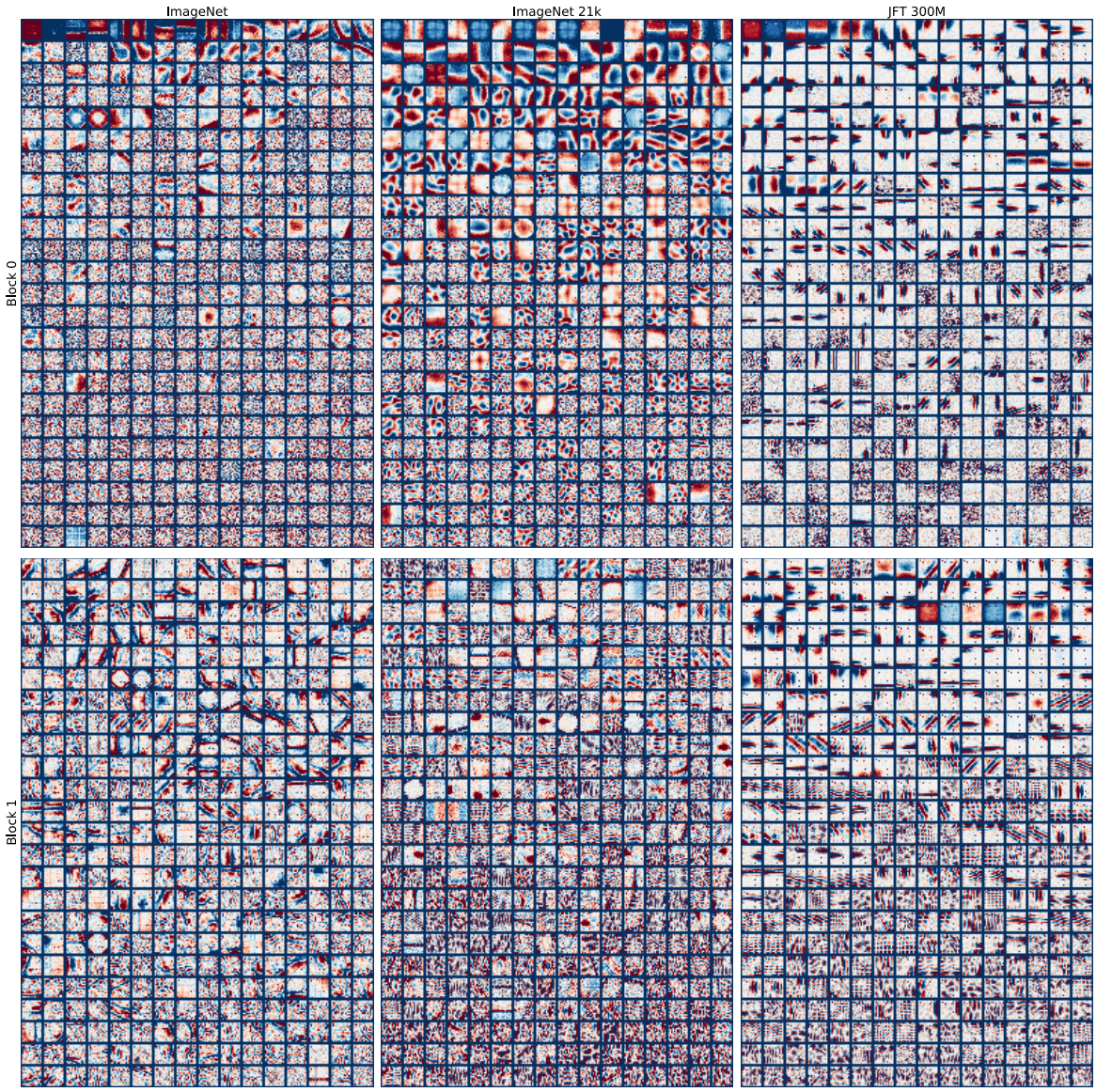
We now move to another important metric, computational complexity. Without loss of generality, assume that the current feature map size is $HW \times C$ (or $H \times W \times C$ in CNNs), where H and W are the spatial resolutions of the feature map and C is the number of channels. For simplicity, we only consider the single-layer Token-Mixer MLP, which does not affect the results of the complexity analysis. For separable convolution, the convolution kernel size k is much smaller than H and W, thus its complexity is $\mathcal{O}(HWC)$ (The dense convolution is $\mathcal{O}(HWC^2)$). For the self-attentive layer, it needs to compute the similarity matrix of size $HW \times HW$, for which the computational complexity is $\mathcal{O}(H^2W^2C)$. For Token-mixing MLP, it is clear that its computational complexity is still $\mathcal{O}(H^2W^2C)$. But it is less computationally intensive than the attention layer due to the lack of calculations such as the similarity matrix.

As for the number of parameters. The minimum separable convolution, for k^2C (The dense convolution is k^2C^2). The mapping needs to be done along the channel direction to calculate the key, query, and value of each token, and the number of parameters of self-attention is $3C^2$. However, for Token-mixing MLP, the number of parameters is H^2W^2, which is strongly correlated with the image resolution. So once the network is fixed, the corresponding image resolution is also fixed. It is worth noting that, ViT is sensitive to image resolution, but this is due to the position embedding rather than the attention layer.

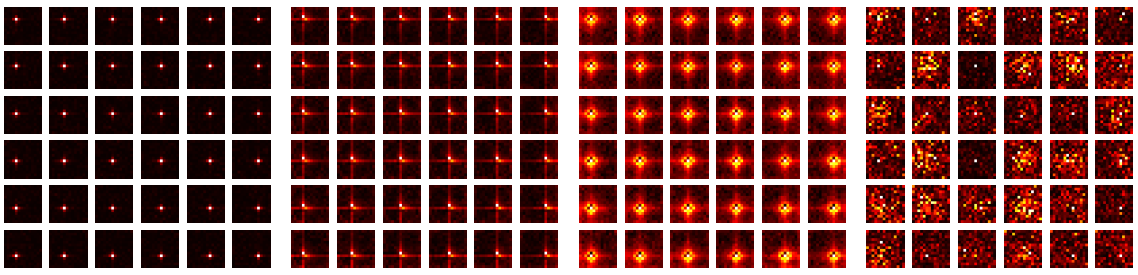
Based on the above analysis and comparison, it is clear that MLP, a seemingly new paradigm, still faces a number of bottlenecks:

(1) Without the inductive biases of the local connectivity and the self-attention, the Token-mixing MLP is more flexible and has a stronger fitting capability. But the freedom is accompanied by the risk of over-fitting than the convolution and self-attention layers. Thus, the ultra large-scale dataset, JFT-300M [67], is needed to shorten the classification accuracy gap between MLP-Mixer and ViT as well as CNN.

(2) Although avoiding the computation of key, query, and attention matrix, the complexity of the Token-mixing MLP is still quadratic to image size (mapping $\mathbb{R}^S \mapsto \mathbb{R}^S$), which makes it intractable for existing MLP-like models on high-resolution images, with the current computing capability.



(a) MLP-Mixer



(b) ResMLP

Figure 5: Visualisation of the linear layers in MLP-Mixer and ResMLP. We show the results of filters for both networks. Each filter matrix is resized to 14×14 pixel images. The results are from [36] and [40], respectively.

(3) The Token-mixing MLP is connected to all spatial tokens, for spatial information communication. Mapping $\mathbb{R}^S \mapsto \mathbb{R}^S$, the structure of Token-mixing MLP is configured by $H \times W$, i.e., the input spatial scale. Once the nodes number in the fully connected layer is set, the network can not deal with flexible input scales. This requirement can be easily satisfied for ImageNet [14] classification as the input resolution of 224×224 is adopted for both training and evaluation as a popular setting. However, some tasks adopt a multi-scale training strategy [82] and have different input resolutions between training and evaluation stages [83, 84]. Furthermore, some downstream tasks usually demand larger input resolution, e.g., 512×512 for semantic segmentation on ADE20K [84], making MLP models non-transfer and impractical.

(4) Such models are composed of blocks with the same architectures, resulting in features with a single-scale in low resolution. Thus, the non-hierarchical architectures make the model infeasible to provide pyramid feature representations, which have been proved to be very important in many downstream tasks, such as object detection and instance segmentation, and semantic segmentation.

To address these challenges, there has been a lot of exploration and practice in the vision community, and many models have sprung up. Their main contributions are modifications to Token-mixing MLP, including reducing computational effort, removing resolution sensitivity, and introducing local receptive field. A few other works have designed the whole network as a pyramidal shape and achieved excellent performance in downstream tasks. We will describe them in detail in the subsequent sections.

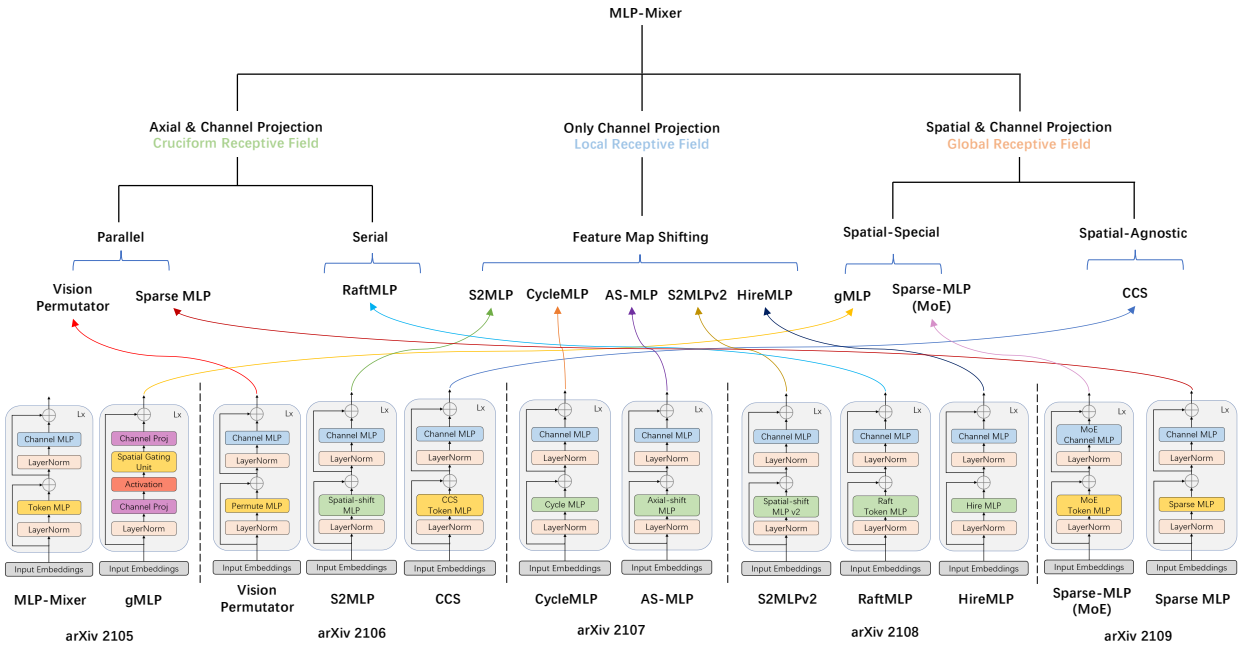


Figure 6: Block comparison of different MLP-Mixer variants. Sorted from left to right by model release month (from May to September). *Yellow* Token-mixing modules indicate sensitivity to image resolution, while *Green* Token-mixing modules indicate insensitivity to image resolution. And "arXiv 2105" denotes "May, 2021". We reproduce most of variants in Jittor [85, 86] and Pytorch [87] respectively.

4 Interior: Development of Token-mixing MLP

In order to overcome the challenges faced by MLP-Mixer, many attempts have been made by the vision community, and a large number of variant models have been proposed in a short period of time. The lower part of Figure 6 compares the block design comparisons of the latest MLP models released in the last five months, and it can be seen that they are basically modified for Token-mixing MLP. Except for the gMLP [88], all other models retain the tandem spatial MLP and channel MLP. And the vast majority of improvements make the spatial MLP no longer sensitive to image resolution (Green Blocks). In this section, we first detail the corresponding modules of the latest MLP variants, then compare their properties and receptive field, and finally discuss and conclude.

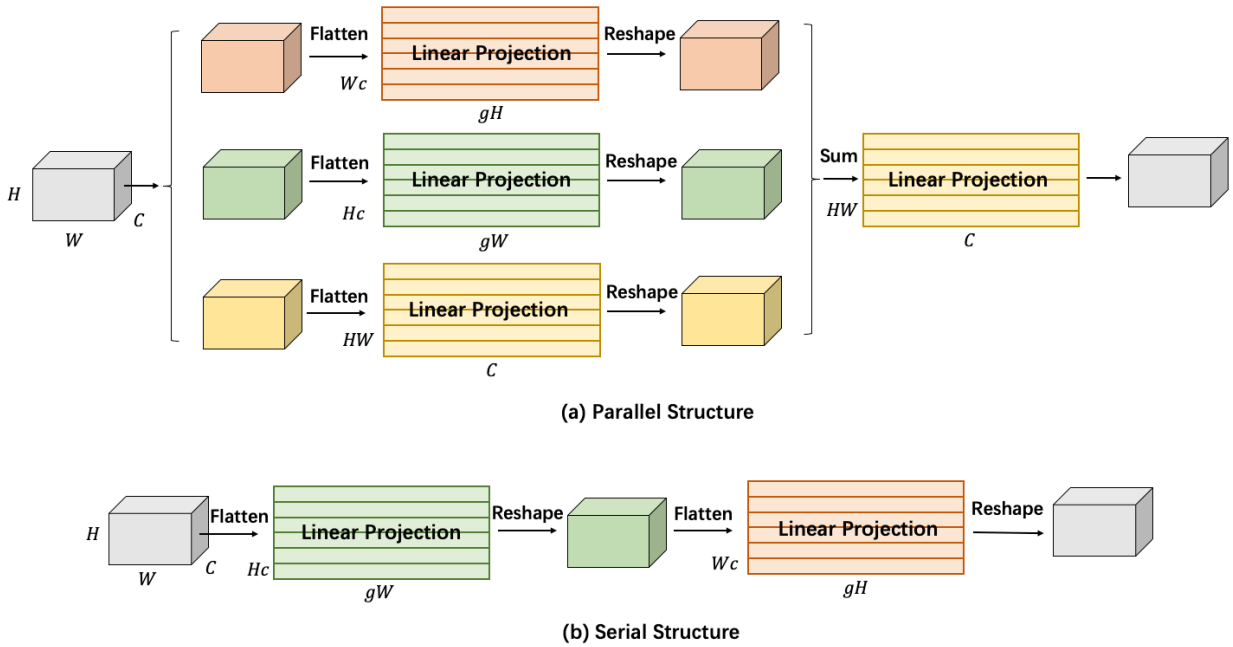


Figure 7: Parallel and serial structures of axial projection blocks. ViP and Sparse MLP adopt parallel strategy, while RaftMLP employs the serial structure. g indicates the number of groups, usually taken as 1, 2, or 3, and we have $C = gc$. The projection of channel dimensions is usually not grouped anymore.

4.1 MLP Block Variants

We divide variants of MLP blocks into three categories: (1) mappings with both whole spatial and channel dimensions, (2) mappings with both the axial direction and channel dimensions, and (3) mappings with only channel dimensions. The network variants are categorized as shown in the upper part of Figure 6. We first review the variants with axial and channel projection, who perform orthogonal decomposition of spatial projections, substantially reducing computational effort and alleviating the overfitting problem on medium-sized datasets. Second, we review the channel-only projection variants, who address the sensitivity to image resolution, and cleverly introduce the concept of local receptive field by shifting the feature map. And we last review the variants that contain both spatial and channel projection, since they only improve performance doing little to alleviate the challenges encountered by Token-mixing MLP. Since the Channel-mixing MLP are basically the same (except for the gMLP), we focus on reviewing and describing the changes to the Token-mixing MLP.

4.1.1 Axial and channel projection blocks

The initial Token-mixing MLP is prone to overfitting on small and medium-sized datasets due to the large number of parameters (H^2W^2) and the global receptive field. How can we maintain the advantage of long-range dependencies while no longer encoding spatial information along the flattened spatial dimensions? The most straightforward approach is to decompose the spatial projection orthogonally.

Hou et al. present Vision Permutator (ViP) [89], which separately encodes the feature representations along the height and width dimensions with linear projections. This design allows ViP to capture long-range dependencies along one spatial direction and meanwhile preserve precise positional information along the other direction. As shown in Figure 7(a), Permute-MLP consists of three branches, each of which is in charge of encoding information along the either height, or width, or channel dimension. It is worth mentioning that, Permute-MLP is not a simple implementation of the $\mathbb{R}^H \mapsto \mathbb{R}^H$ mapping along the height dimension. It first splits the feature map into g segments along the channel dimension, where $g = C/H$, and the segments are then concatenated along the height dimension. Thus, it actually maps $\mathbb{R}^{gH} \mapsto \mathbb{R}^{gH}$, and is shared across width and part channels. After mapping, the feature map is next recover to the original shape. The branch treatment in the width direction is the same as in the height direction. As for the third path is a simple mapping along the channel dimension, which can also be regarded as a 1×1 convolution. Finally, the outputs from all three branches is fused by exploiting the split attention [90].

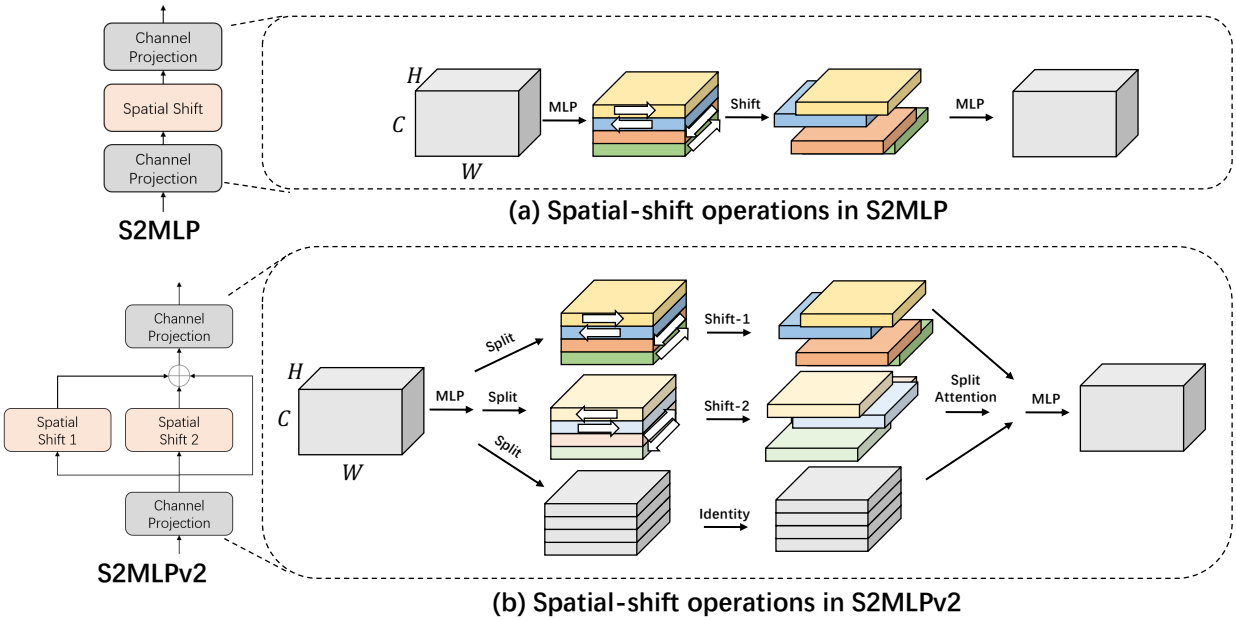


Figure 8: Basic structure of spatial-shift operations in S2MLP and S2MLPv2.

Tang et al. adopt a strategy consistent with ViP for spatial information extraction, and build an attention-free network called Sparse MLP [91]. As illustrated in Figure 7(a), the sMLP block also contains three parallel branches. The difference from ViP is that, it no longer splits the feature map along the channel dimension, but directly maps $\mathbb{R}^H \mapsto \mathbb{R}^H$ and $\mathbb{R}^W \mapsto \mathbb{R}^W$ along the height and width dimension. Without split attention, Sparse MLP's fusion strategy is to concatenate the three tributary outputs by channel and then pass them through a fully connected layer for dimensionality reduction. There is also a slight improvement where Sparse MLP places a depthwise convolution in front of the sMLP block.

Different from the parallel branches of ViP and Sparse MLP, RaftMLP [92] directly concatenates the mappings in the height and width dimensions to form a Raft-Token-mixing block (Figure 7(b)). In the specific implementation, the Raft-Token-mixing block also splits the feature map along the channel dimension, which is consistent with ViP.

Encoding spatial information along the axial direction instead of the whole plane preserves the long-range dependence to a certain extent, and reduces the number of parameters and computational cost from $\mathcal{O}(H^2W^2)$ to $\mathcal{O}(H^2 + W^2)$. Experimental results also show that such a change can bring performance improvement for MLP-Mixer on medium-sized dataset (e.g. ImageNet1k). But such design still does not solve the problem that *the network is sensitive to the image resolution*.

4.1.2 Channel-only projection blocks

To achieve resolution-insensitive, the easiest way to think of is to replace all the spatial fully connected layers with channel projection, i.e., a 1×1 convolution. However, it makes the patches no longer interact with each other, and the concept of the receptive field disappears. To reintroduce receptive field, many works align features at different spatial locations to the same channel by shifting (or moving) the feature maps, and then interact with spatial information through channel projection. Such an operation enables us to utilize an MLP architecture to achieve the same local receptive field as CNN-like architecture.

Yu et al. from Baidu Research first propose a spatial-shift MLP architecture for vision, called S2MLP [93]. The actual practice is quite simple. As shown in Figure 8(a), The proposed spatial-shift module groups C channels into several groups, and it shifts different groups of channels in different directions. The feature map after shift aligns different token features to the same channel, and then the interaction of spatial information can be realized after channel projection. In view of the simplicity of this approach, Yu et al. then borrow the idea of ViP to extend the spatial-shift module into three parallel branches and then fuse the branch features by split attention to further improve the performance of the network (Figure 8(b)). This newly proposed network is called S2MLPv2 [94].

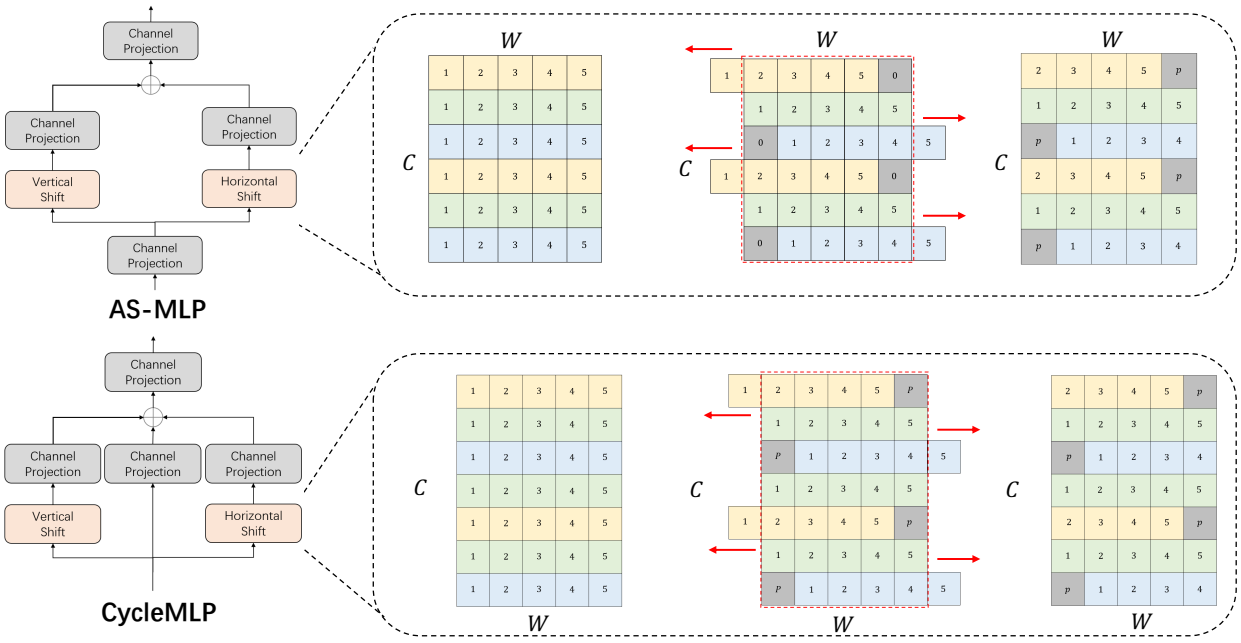


Figure 9: Basic structure of axial-shift operations in AS-MLP and CycleMLP, where the red arrow indicates the shifting direction and the gray P indicates padding value.

Unlike grouping and then performing the same shift operation for each group, the Axial Shifted MLP (AS-MLP) [95] performs different operations within each group which contains a few layers, like 3 or 5. For example, every three consecutive feature maps in the channel direction are, respectively, left-shifted, no-shifted, right-shifted, and so on. In addition, AS-MLP uses two parallel branches for horizontal and vertical shifting. The outputs of the two branches are element-added and fed into the full connection in the channel dimension for information integration. It is worth mentioning that AS-MLP is also extended with different shifting strategies, allowing the receptive field to be similar to the dilated convolution (atrous convolution) [96, 97].

Chen et al. publish CycleMLP [98] on arXiv three days after AS-MLP is proposed. Although CycleMLP does not directly shift feature maps, it integrates features at different spatial locations along the channel direction by deformable convolution [99], which is an equivalent approach to shifting the feature map. A slight difference is that, CycleMLP has three branches (horizontal, constant, and vertical), while AS-MLP has only two branches (horizontal and vertical). In addition, CycleMLP also relies on split attention to fuse the branched features. The comparison of AS-MLP and CycleMLP is shown in Figure 9.

In contrast to the aforementioned models, HireMLP [100] adopts a completely new idea. More specifically, HireMLP presents inner-region rearrangement and cross-region rearrangement, as shown in Figure 10. In the inner-region rearrangement, the feature maps are first grouped along the height or width dimension, and then the tokens within the group are concatenate along the channel direction. Without loss of generality, let us consider the vertical direction. The feature map $X \in \mathbb{R}^{H\times W\times C}$ is divided into g groups along the height dimension, and we get $\{X_1, x_2, \cdots, X_g\}$. For feature X_i with shape $h \times W \times C$ in the i-th region along the height direction, where $h = H/g$, we get the rearranged feature X_i^c with shape $W \times hC$ with inner-region rearrangement operation. Usually h is small, e.g. 2 to 5. Then the rearranged feature X_i^c is sent to a MLP module $\mathcal{F}$ to mix information along the last dimension and get output feature $X_i^O \in \mathbb{R}^{W\times hC}$. For efficiency, the MLP $\mathcal{F}$ is implemented by two linear projections with bottleneck, i.e., the feature dimension is firstly reduced to $W \times C/2$ and then restored to $W \times hC$. With inner-region rearrangement, the output tokens only involve input tokens in a local region, producing a local receptive field whose size depends on the size of each region. To exchange information across different regions, we need cross-region rearrangement, that is, recurrently shifting all the tokens along a specific direction with a given step size s. After shifting, the tokens contained by a local region will change. At a macro level, the HireMLP block still consists of three parallel branches, and then the output feature is obtained by summing up these branched features (Figure 10).

There is another special variant that uses only channel projection, called ConvMLP [101]. Its authors call it a hierarchical Convolutional MLP which is a light-weight, stage-wise, co-design of convolution layers. For spatial information

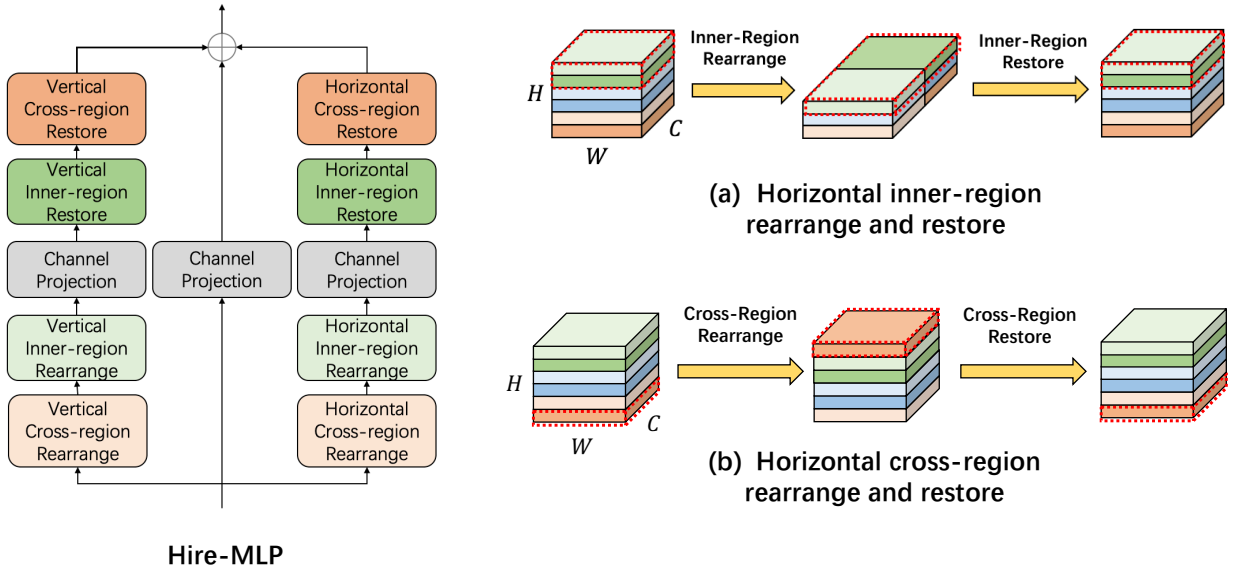


Figure 10: Basic structure of HireMLP Block.

fusion, ConvMLP uses depthwise convolution instead of shifting feature maps. In fact, it just replaces part of the 1×1 convolution in the CNN with a Channel-mixing MLP. The only difference is that one 1×1 convolution layer is replaced by two layers and the GELU activation function is inserted in the middle, as well as LayerNorm uses before and after. Therefore, we consider ConvMLP as a pure CNN model rather than an MLP-like model. In addition, UniNet [102] jointly searches the optimal combination of convolution, Transformer, and MLP for building a series of all-operator network architectures with high performances on visual tasks. It is beyond the scope of Vision MLP. For the completeness of the survey, we briefly present them here, but the two are not in our main analysis.

4.1.3 Spatial and channel projection blocks

There are also some variants that still retain full space and channel projection. Their module design is not short of sparkle and enhances the performance of the MLP-like model to some extent.

gMLP [88] is the first proposed variant of MLP-mixer, proposed by Liu et al., who have experimented with several design choices for token-mixing architecture and find spatial projections work well when they are linear and paired with multiplicative gating. In detail, the spatial gating unit first does a linear projection for the input X, $f_{W,b} = WX + b$. The spatial projection matrix $W \in \mathbb{R}^{HW \times HW}$ here is independent from the input representations, where HW is the number of token. Then the output of the spatial gating unit is $X \odot f_{W,b}(X)$, where $\odot$ denotes the element-wise product. For training stability, gMLP initializes W as near-zero values and b as ones, meaning that $f_{W,b}(X) \approx 1$ and therefore $X \odot f_{W,b}(X) \approx X$ at the beginning of training. This initialization ensures each gMLP block behaves like a regular FFN at the early stage of training. They further find it effective to split X into two independent parts (X_1, X_2) along the channel dimension for the gating function and for the multiplicative bypass: $X_{out} = X_1 \odot f_{W,b}(X_2)$. Notice that in the gMLP block, there is a channel projection before and after the spatial gating unit. But there is no channel-mixing MLP anymore. Pleasingly, gMLP achieves good performance in both computer vision and natural language processing.

The motivation of Lou et al. [103] is different from all the aforementioned work. They consider how to scaling the network to more parameters with comparable computation cost, making the model more computation-efficient and better performing. Specifically, they introduce Mixture-of-Experts (MoE) [104] into the MLP-Mixer, and propose Sparse-MLP, which renamed with the work of Tang et al [91]. In fact, Carlos et al. [105] had already applied MoE on the FeedForward of Transformer block (i.e., Channel-mixing MLP in MLP-Mixer) three months earlier (in June 2021) than Lou et al. did. Thus, Lou et al. are the ones who expanded MoE from Channel-mixing MLP to Token-mixing MLP and achieved some performance gains compared to the most primitive MLP-Mixer. MoE only adds parallel fully connected layers as well as some auxiliary losses. It is not a modification of the network design, so we do not describe it in detail here. More details about MoE can be found in [104, 105].

Finally, and the most ingeniously, Yu et al. [106]. cleverly solve the problem of Token-mixing MLP sensitivity to spatial location while preserving the global receptive field. Please note that is not a solution to the problem that the

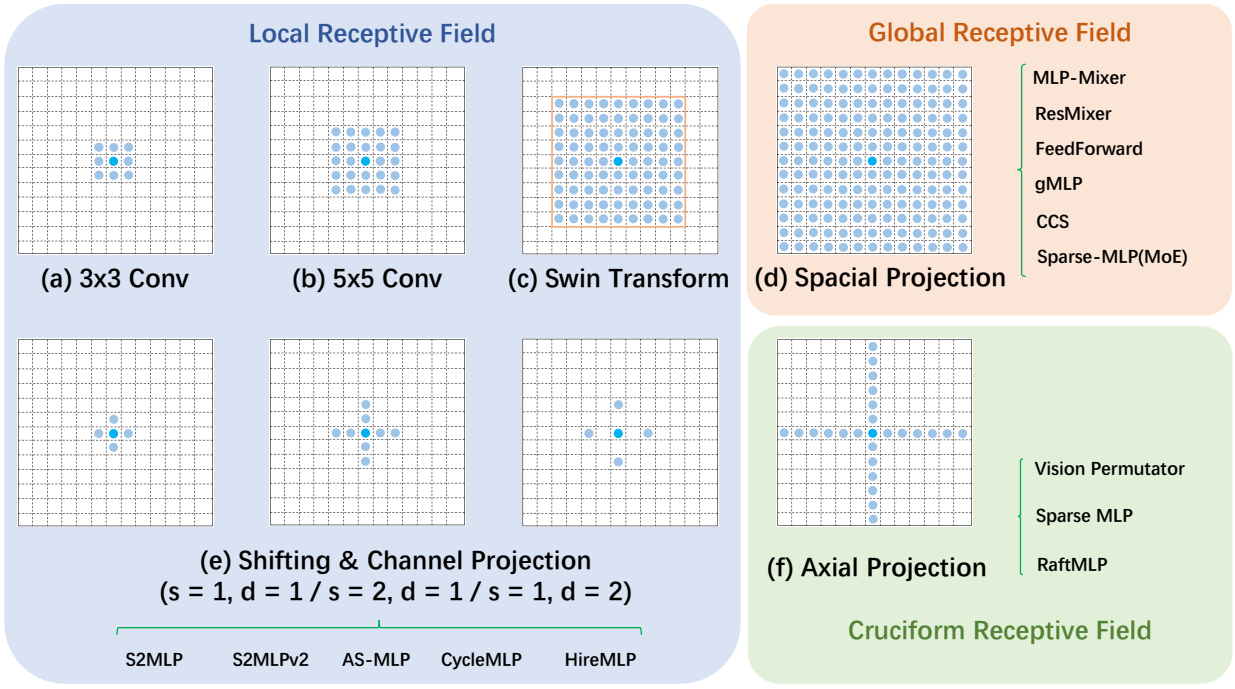


Figure 11: The schematic diagram of different receptive field, including local, global and cruciform. The channel dimension is omitted here. The bright blue dot indicates the position of the encoded token. The orange rectangle is the window of Swin Transformer. (s = 1, d = 1) shows the shifting operation when the shift size is 1 and dilation is 1.

model is sensitive to the input resolution. The spatial-specific property makes the Token-mixing MLP sensitive to spatial translation. In comparison, both the convolution layer and attention layer are spatial-agnostic. By imposing a circulant-structure constraint on the weights of the Token-mixing MLP, spatial projection is no longer sensitive to spatial translation. However, sharing such weights in all channels limits its capability in mixing tokens. For this purpose the Yu groups the feature maps in the channel dimension, sharing weights within groups and varying weights between groups. To this end, the improved structure is termed as Circulant Channel-Specific (CCS) token-mixing MLP, which is spatial invariant and channel-specific. CCS reduces the number of parameters from $\mathcal{O}(H^2W^2)$ to $\mathcal{O}(HW)$, but does not reduce the computation cost. To be computationally efficient, Yu further uses fast Fourier transform (FFT)[3], making the multiplications between a N-dimension vector and an $N \times N$ circulant matrix only take $\mathcal{O}(N\sqrt{N})$. In other words, it reduces the FLOPs from $\mathcal{O}(H^2W^2C)$ to $\mathcal{O}(HW\sqrt{HW}C)$. Experimental results show that it takes fewer parameters but achieves higher classification accuracy on ImageNet1K benchmark.

4.2 Receptive Field and Complexity Analysis

The main novelty of MLP is allowing the model to autonomously learn the global receptive field from the raw data. But do these so-called MLP variants still hold true to the original intent? Immediately following the module design, we compare and analyze the receptive field of these blocks, in order to provide a more in-depth presentation for those MLP variants. Following that, we compare and analyze the complexity of different modules themselves, and the comparison is provided in Table 2. A comparison of the spatial sensitivity, channel sensitivity, and resolution sensitivity of the block is also provided. These properties are mentioned in the block design, and we do not describe them in detail.

4.2.1 Receptive field

Receptive field, region in the sensory periphery within which stimuli can influence the electrical activity of sensory cells, is a physiology concept. The term *receptive field* was first used by Sherrington in 1906 to describe the area of

[3]The matrix-vector multiplication $\mathbf{Wx}$ can be computed efficiently through fast Fourier transform:

$$\mathbf{xW} = \mathrm{FFT}[\mathrm{IFFT}(\mathbf{x}) \odot \mathrm{FFT}(\mathbf{W})], \tag{2}$$

where FFT denotes fast Fourier transform, IFFT denotes inverse fast Fourier transform, and $\odot$ denotes the element-wise product.

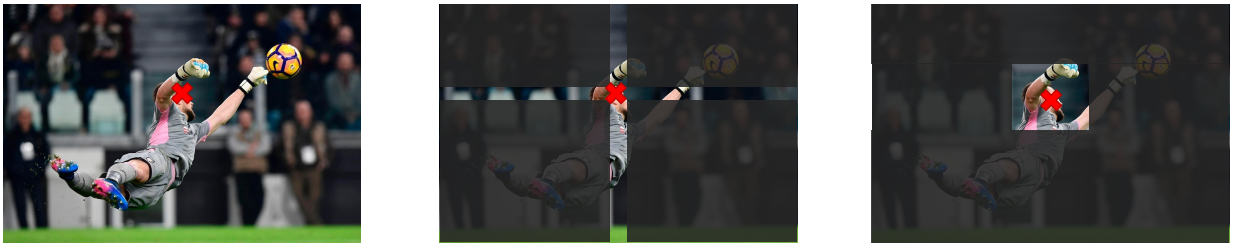


Figure 12: Schematic diagram receptive field response, where a red cross indicating the location of concern (the encoded token), with the activated part bright and the inactive part dark.

skin from which a scratch reflex could be elicited in a dog [107]. Nowadays, the term receptive field is also used in the context of artificial neural networks, which is defined as the size of the region in the input that produces the feature. Basically, it is a measure of the association of an output feature (of any layer) to the input. Figure 11 displays the schematic diagram of different receptive field and the corresponding MLP-like models.

Obviously, the full spacial projection in gMLP, Sparse-MLP, and CCS still retains the global receptive field (Figure 11(d)), as the same with MLP-Mixer, that is, the encoded features of each token are the weighted sum of all token features. Without using similarity as weights in self-attention layer or local connectivity in convolution layer, the Token-mixing MLP requires more data for training. It must be acknowledged that this global receptive field is at the patch-level, not at the pixel-level in the traditional sense. In other words, the globalness is approximated by patch partition. And this is the same in Transformer-based. The size of the patch affects the final result as well as the computational complexity of the network. Notably, the patch partition is actually a strong artificial assumption, that is always ignored.

As a choice for balancing long-distance dependence and computational cost, axial projection decomposes the full spacial projection orthogonally, i.e., along with horizontal and vertical directions. Projection of the two axis directions is done in serial (RaftMLP) or parallel (ViP and Sparse MLP). Thus, the token to be encoded only interacts with horizontal or vertical tokens in a single projection. The resulting cruciform receptive field is shown in Figure 11(f). In contrast to global receptive field, the cruciform receptive field retains horizontal and vertical long-range dependence. However, if the token of interest and the current token are not at the same horizontal or vertical area, the two tokens cannot interact with each other. As shown in Figure 12, if there is a soccer ball in the upper right corner of the goalkeeper, such a cruciform receptive field may not interact well with the soccer and the goalkeeper.

A number of MLP variants released centrally in the summer of 2021 shift feature maps and use channel projection to interact with information from different tokens. The feature map shifting size is usually small, i.e. 1 or 2, and a local receptive field is formed for this reason. In Figure 11, we compare the convolution, Swin Transformer [27], and MLP variant models based on shifting feature maps. The receptive field of convolution is usually smaller, and the window of Swin Transformer will be slightly larger. And both they are square. However, since the shifting operation of the MLP variant is usually divided into horizontal and vertical directions, the resulting receptive field is cruciform. By varying the shift size and dilation, different sizes of receptive field can be formed and even contain voids, just like a dilated convolution. In fact, all these MLP-like models that contain only channel projection cannot perceive how large the specific resolution is. In other words, they give up the long-range dependence (global receptive field) in order to be insensitive to the image resolution. This defeats the original purpose of the pure MLP model.

Virtually, the cruciform and local receptive field is a special case of the global receptive field, e.g., the weight value is approximately 0 except for a specific area. Therefore, these two types of receptive field are equivalent to learning only a small part of the weights of the global receptive field and letting the other weights be constant to 0. This is also an artificial inductive bias, just like the locality introduced by the convolution layer.

4.2.2 Complexity analysis

Since the channel projection contains in all aforementioned network blocks, where the number of parameters is $\mathcal{O}(C^2)$ and the FLOPs is $\mathcal{O}(HWC^2)$, we ignore this item in the later analysis and focus mainly on the module used for spatial information fusion, i.e., Token-mixing MLP and its variants. Table 2 lists the comparison of the attribute and complexity analysis of different spatial information fusion modules, naming each module by the network name for ease of understanding.

Table 2: Comparison of different variants of spatial information fusion modules.

Spacial Operation	Block	Spatial	Channel	Scale variable	param	FLOPs
Spacial Projection	MLP-Mixer	specific	agnostic	False	$\mathcal{O}(H^2W^2)$	$\mathcal{O}(H^2W^2C)$
	ResMLP	specific	agnostic	False	$\mathcal{O}(H^2W^2)$	$\mathcal{O}(H^2W^2C)$
	FeedForward	specific	agnostic	False	$\mathcal{O}(H^2W^2)$	$\mathcal{O}(H^2W^2C)$
	gMLP	specific	agnostic	False	$\mathcal{O}(H^2W^2)$	$\mathcal{O}(H^2W^2C)$
	Sparse-MLP	specific	agnostic	False	$\mathcal{O}(H^2W^2)$	$\mathcal{O}(H^2W^2C)$
	CCS	agnostic	group-specific	False	$\mathcal{O}(HW)$	$\mathcal{O}(HW\sqrt{HWC})$
Axial Projection	RaftMLP	specific	agnostic	False	$\mathcal{O}(H^2+W^2)$	$\mathcal{O}(HWC(H+W))$
	ViP	specific	specific	False	$\mathcal{O}(H^2+W^2+C^2)$	$\mathcal{O}(HWC(H+W+C))$
	Sparse MLP	specific	specific	False	$\mathcal{O}(H^2+W^2+C^2)$	$\mathcal{O}(HWC(H+W+C))$
Shifting & Channel Projection	S2MLP	agnostic	specific	True	$\mathcal{O}(C^2)$	$\mathcal{O}(HWC^2)$
	S2MLPv2	agnostic	specific	True	$\mathcal{O}(C^2)$	$\mathcal{O}(HWC^2)$
	AS-MLP	agnostic	specific	True	$\mathcal{O}(C^2)$	$\mathcal{O}(HWC^2)$
	CycleMLP	agnostic	specific	True	$\mathcal{O}(C^2)$	$\mathcal{O}(HWC^2)$
	HireMLP	agnostic	specific	True	$\mathcal{O}(C^2)$	$\mathcal{O}(HWC^2)$

The full spacial projection in MLP-Mixer, gMLP, Sparse-MLP contains the number of parameters $\mathcal{O}(H^2W^2)$, and its FLOSs is $\mathcal{O}(H^2W^2C)$, both quadratic with the image resolution. Thus, it is difficult to apply the network to large resolution images, with the current computational power. If a compromise is made to get computationally efficient, the size of each patch will be set to be very large (e.g. 14×14 or 16×16), and the information extracted is too coarse and not friendly to small objects, etc. The only difference is that CCS constructs an $N \times N$ weight matrix using weight vectors of length N and uses a fast Fourier transform to reduce the computational cost.

In contrast, the axial projection, which is the orthogonal decomposition of the full spacial projection, reduces the number of parameters from $\mathcal{O}(H^2W^2)$ to $\mathcal{O}(H^2+W^2)$, and reduces the FLOPs from $\mathcal{O}(H^2W^2C)$ to $\mathcal{O}(HWC(H+W))$, just as RaftMLP does. If three parallel branches are used, the number of parameters and the FLOPs are $\mathcal{O}(H^2+W^2+C^2)$ and $\mathcal{O}(HWC(H+W+C))$, respectively. Notice that the fusion of information from multiple branches does not actually increase the computational complexity. Notice that the fusion of information from multiple branches does not actually increase the computational complexity, such as splitting attention, dimensionality reduction after channel concatenation, etc.

The approach based on shifting the feature map and channel projection further reduces the computational complexity, i.e., the number of parameters is $\mathcal{O}(C^2)$ with FLOPs $\mathcal{O}(HWC^2)$. The number of weights is decoupled from the image resolution, so that the network is no longer constrained by the image resolution.

It is worth raising that, the reduction in complexity does not mean that the proposed network has fewer parameters. Conversely, the various networks retain a comparable number of parameters. This allows networks of lower complexity to have a smaller patch size (e.g. 4×4) and a larger number of channels to obtain a finer and larger number of features.

4.3 Discussions

The preceding methods take different approaches in how they attempt to reduce model computational complexity, mitigate overfitting, and resolve image resolution sensitivity. Specifically, decompose the full spacial projection orthogonally, or use only the channel projection. These careful and clever designs show that researchers have noticed that the current amount of data and computational power is not enough for pure MLPs. Note that the former does not address the issue of image resolution sensitivity, the latter does. However, the channel projection makes it impossible to interact between different tokens. To solve this problem, researchers shift feature maps so as to interact with information within the local receptive field. The seemingly clever design actually deviates from the original purpose of the MLP model, that is, the long-range dependence is no longer retained. In this sense, the design is already out of the realm of pure MLP models. In addition, Chen et al. note that the shifting feature map and channel projection can be equivalently implemented by deformable convolution[4]. This further illustrates the point that the new paradigm is instead moving towards the old paradigm. To put it more bluntly, the development of MLP back to the way of CNNs. To this end, we still need to make effort on how to balance long-distance dependence and image resolution sensitivity.

[4]Deformable convolution is used in the official CycleMLP implementation: `https://github.com/ShoufaChen/CycleMLP/blob/main/cycle_mlp.py`

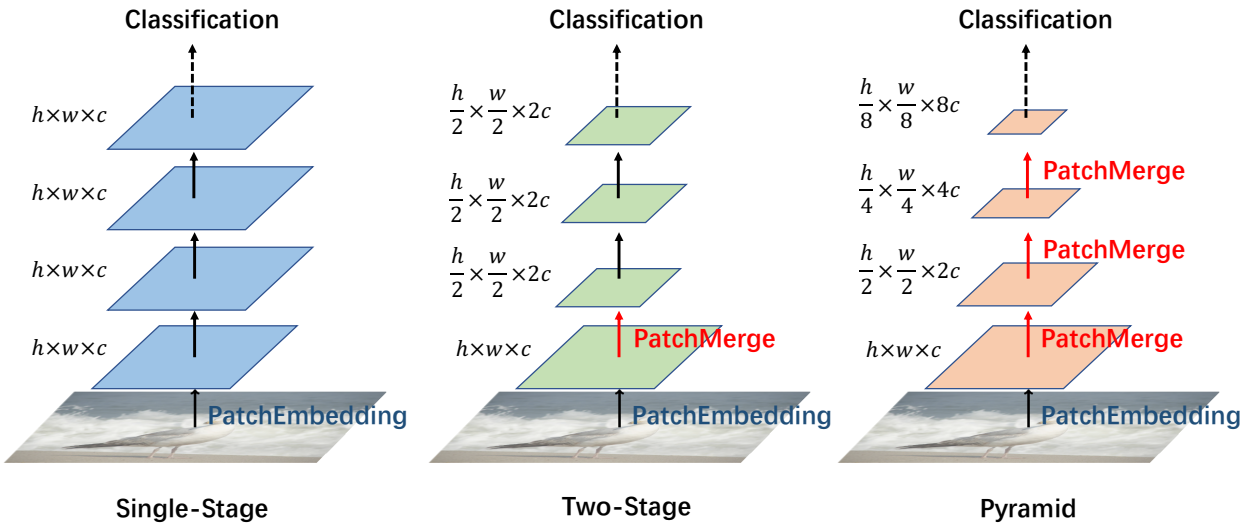


Figure 13: Comparison of different hierarchical models. After patch embedding, the feature map size is $h \times w \times c$. There is a patch merge between every two stages, usually 2×2 patches are merged together, and the number of channels becomes twice.

5 Exterior: Network Architecture

Regardless of the paradigm, convolution layer, self-attentive layer or fully connected layer, they all need to be stacked to form a network and further work on a specific task. According to our investigation, the traditional stacking pattern is also applicable to MLP, and MLP variants can be divided into three categories: (1) single-stage structure inherited from the MLP-Mixer, (2) two-stage structure with smaller patches in the early blocks, and (3) CNN-like pyramid structure. Later in this section, we compare the performance of these variant models on ImageNet1k in detail, as well as the performance of some of them on downstream tasks.

5.1 From Single-stage to Pyramid

MLP-Mixer [36] inherits the "isotropic" design of Vision Transformer [25], that is, each layer in Mixer (except for the initial patch projection layer) takes an input of the same size. After the patch embedding, each block does not change the size of the feature map, which is called a *single-stage* structure. As shown in the leftmost subdiagram in Figure 13. Models of this design also include FeedForward [37], ResMLP [40], gMLP [88], S2MLP [93], CCS [106], RaftMLP [92], Sparse-MLP(MoE) [103]. Due to the limited computing resources, the patch partition during patch embedding of the single-stage model is usually large, e.g. 16×16 or 14×14. The coarse-grained patch partition makes the subsequent feature fineness limited. Although the impact is not significant for single-object classification, it has an impact on many downstream tasks such as object detection and instance segmentation, especially for small targets.

Intuitively, the smaller patches are beneficial to modeling fine-grained details in the images and tend to achieve higher recognition accuracy. Vision Permutator [89] further proposes a *two-stage* configuration based on the single-stage. In more detail, the network takes 7×7 patch slices in the initial patch embedding and performs a 2×2 patch merge after a few layers, as shown in the middle subfigure of Figure 13. During patch merging, the height and width of the feature map get halved while the channels are expanded to twice. Comparing with the 14×14 patch embedding, encoding fine-level patch representations brings slight Top-1 accuracy improvement on ImageNet1k (from 80.6% to 81.5%). S2MLPv2 [94] follows Vision Permutator and achieves similar Top-1 accuracy improvement on ImageNet1k (from 80.9% to 82.0%).

If the initial patch size is further reduced (e.g. 4×4), more patch merging will be needed subsequently in order to reduce the number of patches (or tokens). This promotes the network to adopt a pyramid-like architecture. Specially, the whole architecture contains four stages, where the feature resolution reduces from $H/4 \times W/4$ to $H/32 \times W/32$ and the output dimension increases accordingly. The network based on this design includes Sparse MLP [91], HireMLP [100], AS-MLP [95] and CycleMLP [98]. Patch embedding can be equivalently achieved by a convolution layer with kernel size equal to stride equal to patch size. In the pyramid structure, the researchers find that using a overlapping patch

Table 3: Basic configuration for different structural models.

Structure	Patch Embedding	Stage1	Stage2	Stage3	Stage4	Classification Header
Single-Stage	16×16	$\frac{H}{16} \times \frac{W}{16} \times C$				Global Pooling Full-connection
Two-Stage	7×7	$\frac{H}{7} \times \frac{W}{7} \times C$	$\frac{H}{14} \times \frac{W}{14} \times 2C$			
Pyramid	4×4	$\frac{H}{4} \times \frac{W}{4} \times C$	$\frac{H}{8} \times \frac{W}{8} \times 2C$	$\frac{H}{16} \times \frac{W}{16} \times 4C$	$\frac{H}{32} \times \frac{W}{32} \times 8C$	

embedding gives better results, that is, convolution of 7×7 with stride $= 4$ instead of 4×4, which is similar to the embedding of ResNet[5] [18].

The basic configuration for different designs is shown in Table 3. The non-pyramid model can work well for image classification tasks because there is usually only a single object in the image. But for some computer vision downstream tasks, i.e., object detection and semantic segmentation where there are multiple objects of different sizes in the image, the pyramid structure may be a better choice. It is worth mentioning that all three designs are adoptable regardless of whether the network is sensitive to image resolution or not. For example, S2MLP is insensitive to image resolution, but is a single-stage model. While Sparse MLP is sensitive to image resolution, but is a pyramid model.

5.2 Model Performance

ImageNet [14] is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Since its release, it has been used as a benchmark for evaluating models in computer vision. Table 4 compares the performance of many vision models on ImageNet1k, including accuracy, parameters, FLOPs, where all results are derived from the cited papers. We divide all network architectures into CNN-based, Transformer-based, and MLP-based architectures. As for the MLP-based models we focus on most, we further divide them into three types of configurations based on the number of parameters. The rows are sorted by the month (or year) of publication. Empirical results show that, MLP-based models can achieve competitive performance compared with CNN-based and Transformer-based architectures, with similar costs. In the MLP variant, almost all the pyramid models achieve better performance than the single-stage models. But the pure MLP model *RaftMLP* with pyramid structure is weaker than the single-stage *S2-MLP*, which is more like a CNN than a pure MLP. This leads us to hold that the current amount of training data is still not sufficient for the pure MLP model. And the inductive bias remains important.

In addition to competitive classification performance, two other recent works [113, 110] explore the robustness of CNNs, Vision Transformer, and MLPs. CNNs have been widely known to be vulnerable to adversarial attacks [114, 115], that is, small additive perturbations of the input cause the CNN to misclassify a sample, causing serious concerns for security-sensitive applications. And recently, there is a line of works [75, 76, 77, 78] that have investigated Transformers from the perspective of adversarial robustness. Their main conclusions can be summarized as *the Vision Transformers are more robust than CNNs*. But how about the MLP-Mixer? Benz et al. [110] compare ResNet, ViT and MLP-Mixer under white-box attack (Table 5), as well as both query-based and transfer-based black-box attacks. In all cases, ViT is the most robust architecture, MLP is the next, while CNN is the least robust. More intently, Morrison et al. [75] explore the impact that the same three architectures have on corruption robustness (Table 6). Similar results are obtained that the robustness of MLP-Mixer generally locates in the middle of ViT and CNNs. Notably, both the two works focus on the empirical evaluation and it is out of the scope of these works to theoretically understand why they might be vulnerable. Admittedly, their attempts for explanation are limited and future work is needed for better understanding.

The paradigm of computer vision serves more than just image classification, which is, after all, relatively simple and low-level. Are the features learned by the model just a carefree overfitting of the data or a generic visual representation? The greater freedom makes the MLP model more difficult to interpret. Thus, it is important to transfer the MLP pre-trained model to the downstream tasks. Empirical studies [116, 117, 118] have shown that the pyramid structure aggregates spatial features to extract semantic information, which is useful for some computer vision downstream tasks, i.e., object detection and semantic segmentation. In addition and more importantly, the input resolution for downstream

[5]The initial embedding layers of ResNet is a 7×7 convolution layer with stride $= 2$ followed by a 2×2 max-pooling layer.

Table 4: The experimental results of different networks on ImageNet-1K benchmark without extra data.

Network	Time	Structure	Top-1 (%)	Params(M)	FLOPs(G)	Open Source Code
CNN-based						
ResNet-50 [18]	2016	Pyramid	76.2	25.6	3.8	True
ResNet-152 [18]	2016	Pyramid	78.3	60.2	11.3	True
EfficientNet-B3 [22]	2019	Pyramid	81.1	12	1.8	True
EfficientNet-B4 [22]	2019	Pyramid	82.6	19	4.2	True
EfficientNet-B5 [22]	2019	Pyramid	83.3	30	9.9	True
RegNetY-4GF [23]	2020	Pyramid	80.0	21	4.0	True
RegNetY-8GF [23]	2020	Pyramid	81.7	39	8.0	True
RegNetY-16GF [23]	2020	Pyramid	82.9	84	15.9	True
EfficientNetV2-S [24]	2021	Pyramid	83.9	22	8.8	True
EfficientNetV2-M [24]	2021	Pyramid	85.1	54	24.0	True
Transformer-based						
ViT-B/16 [25]	2020	Single-stage	77.9	86	55.5	True
DeiT-B/16 [29]	2020	Single-stage	81.8	86	17.6	True
PVT-Large [30]	2021	Pyramid	82.3	61	9.8	True
TNT-B [28]	2021	Single-stage	82.8	66	14.1	True
CaiT-S36 [34]	2021	Single-stage	83.3	68	13.9	True
Swin-B [27]	2021	Pyramid	83.5	88	15.4	True
Nest-B [33]	2021	Pyramid	83.8	68	17.9	True
Container [32]	2021	Pyramid	82.7	22	8.1	True
MLP-based						
ResMLP-12 [40]	2021.05	Single-stage	76.6	15	3.0	True
gMLP-S [88]	2021.05	Single-stage	79.4	20	4.5	True
ViP-Small/7 [89]	2021.06	Two-stage	81.5	25	6.9	True
AS-MLP-T [95]	2021.07	Pyramid	81.3	28	4.4	True
CycleMLP-B2 [98]	2021.07	Pyramid	81.6	27	3.9	True
Hire-MLP-Ti [100]	2021.08	Pyramid	78.9	17	2.1	False
S2-MLPv2-Small/7 [94]	2021.08	Two-stage	82.0	25	6.9	False
Sparse-MLP-S [103]	2021.09	Single-stage	71.3	21	-	False
sMLPNet-T [91]	2021.09	Pyramid	81.9	24.1	5.0	False
Mixer-B/16 [36]	2021.05	Single-stage	76.4	59	11.7	True
FF [37]	2021.05	Single-stage	74.9	62	11.4	True
ResMLP-36 [40]	2021.05	Single-stage	79.7	45	8.9	True
S2-MLP-deep [93]	2021.06	Single-stage	80.7	51	9.7	False
Mixer-B/16 +CCS [106]	2021.06	Single-stage	79.8	57	11	False
ViP-Medium/7 [89]	2021.06	Two-stage	82.7	55	16.3	True
AS-MLP-S [95]	2021.07	Pyramid	83.1	50	8.5	True
CycleMLP-B4 [98]	2021.07	Pyramid	83.0	52	10.1	True
Hire-MLP-B [100]	2021.08	Pyramid	83.1	58	8.1	False
RaftMLP-12 [92]	2021.08	Single-stage	78.0	58	12.0	False
S2-MLPv2-Medium/7 [94]	2021.08	Two-stage	83.6	55	16.3	False
Sparse-MLP-B [103]	2021.09	Single-stage	77.9	69	-	False
sMLPNet-B [91]	2021.09	Pyramid	83.4	65.9	14.0	False
gMLP-B [88]	2021.05	Single-stage	81.6	73	15.8	True
S2-MLP-wide [93]	2021.06	Single-stage	80.0	71	14.0	False
ViP-Large/7 [89]	2021.06	Two-stage	83.2	88	24.3	True
AS-MLP-B [95]	2021.07	Pyramid	83.3	88	15.2	True
CycleMLP-B5 [98]	2021.07	Pyramid	83.2	76	12.3	True
Hire-MLP-L [100]	2021.08	Pyramid	83.4	96	13.5	False
Sparse-MLP-L [103]	2021.09	Single-stage	79.2	130	-	False

Table 5: White-box attacks on benchmark models with different epsilons. The clean accuracy on the ImageNet and NeurIPS datasets is displayed. The attack success rate (%) of PGD and FGSM under l_∞ distortion, and the l_2-norm of C&W [108] and DeepFool [109], respectively are reported for the NeurIPS dataset, which was originally introduced in the NeurIPS 2017 adversarial challenge. A model with a lower ASR↓ or higher l_2-norm↑ is considered to be more robust. This table is from [110].

Model	Clean		PGD (l_∞) ↓					FGSM (l_∞) ↓					C&W (l_2) ↑	DeepFool (l_2)↑
	ImageNet	NeurIPS	0.1	0.3	0.5	1	3	0.1	0.3	0.5	1	3		
ResNet-18 [18]	69.8	83.7	46.1	90.0	97.8	99.9	100	42.0	75.2	88.5	95.7	98.2	0.302	0.237
ResNet-50 [18]	76.1	93.0	35.8	86.3	97.9	99.5	100	27.5	63.1	77.6	89.4	93.9	0.371	0.287
ViT-B/16 [25]	81.4	90.7	**22.6**	63.6	86.5	97.5	99.9	**19.1**	38.7	52.8	66.3	79.7	**0.468**	0.425
ViT-L/16 [25]	82.9	89.3	22.8	**60.1**	80.9	95.8	100	19.5	**35.9**	**44.9**	**57.9**	**67.3**	0.459	**0.548**
Mixer-B/16 [36]	76.5	86.2	29.5	63.4	82.0	96.2	100	27.7	49.3	59.5	69.3	78.0	0.375	0.339
Mixer-L/16 [36]	71.8	80.0	41.1	67.3	**80.4**	**92.1**	**99.4**	36.7	51.8	56.9	61.6	67.4	0.297	0.377

Table 6: Evaluating Convolutional Neural Networks against Vision Transformer Architectures and MLP-Mixers. Top-1 accuracy is calculated for ImageNet1k validation set. mCE is calculated using only severity 2 and 5 on ImageNet-C [111]. Shape bias is calculated on Texture-Cue Conflict dataset [112]. A model with a lower mCE↓ is considered to be more robust, and a higher shape bias↑ is more favorable. This table is from [75].

Model	Top-1(%)	mCE(%)↓	Shape bias(%)↑	Params(M)
CNN-based				
ResNet50 [18]	76.02	65.54	26.17	26
AlexNet [15]	56.44	83.18	29.80	61
GoogLeNet [17]	71.70	68.82	28.52	7
VGG16 [16]	69.63	75.10	16.12	138
Transformer-based				
ViT-B/16 [25]	75.73	58.55	49.10	86
ViT-L/16 [25]	79.16	49.02	**55.35**	304
DeiT-B [29]	81.84	42.30	42.32	86
DeiT-S [29]	79.68	47.79	38.26	22
CaiT-s24 [34]	83.28	40.59	38.65	47
CaiT-xxs24 [34]	78.38	49.28	34.24	11
Swin-B [27]	84.90	38.52	36.39	88
Swin-L [27]	85.92	**34.63**	40.20	197
MLP-based				
Mixer-B/16 [36]	72.53	65.54	36.90	60
Mixer-L/16 [36]	68.25	69.65	38.64	208

tasks is usually large, e.g. 512×512 or larger. This requires the model to be resolution-insensitive. To this end, only three of the MLP-based models can currently be used for downstream tasks, Hire-MLP, AS-MLP, and CycleMLP.

Table 7 and Table 8 present the performance of AS-MLP and CycleMLP as a backbone on downstream tasks, including object detection and semantic segmentation[6]. For the same backbone and architecture, the difference in performance mainly stems from the different choices of training strategies. Both AS-MLP and CycleMLP achieve comparable performance with Swin Transformer[7] in the similar resource limitation.

However, in the practice, AS-MLP and CycleMLP can actually be implemented by deformable convolution, and they can be considered as a special kind of CNN rather than a pure MLP backbone. To this end, we do not yet have a pure MLP backbone for downstream tasks. Lacking pre-training, training a pure MLP model directly on a current object detection or segmentation dataset is predictably poor in performance. More importantly, pure MLP cannot be resolution-insensitive. Thence, the direct use of pure MLP models for downstream tasks is still a long way off.

[6]Since the Hire-MLP authors have not experimented on downstream tasks and have not open-sourced it, we do not consider it for now.

[7]Swin Transformer has achieved the state-of-the-art performance on many downstream tasks, as detailed in `https://github.com/microsoft/Swin-Transformer`

Table 7: The object detection and instance segmentation results of different backbones on the COCO val2017 dataset [83]. Mask R-CNN [59] framework is employed. † Results are from [98], and others are from [95].

Backbone	Mask R-CNN							
	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	Params	FLOPs
ResNet50 † [18]	38.0	58.6	41.4	34.4	55.1	36.7	44.2M	260G
ResNet50 [18]	41.0	61.7	44.9	37.1	58.4	40.1	44.2M	260G
PVT-Small † [30]	40.4	62.9	43.8	37.8	60.1	40.3	44.1M	245G
PVT-Small [30]	43.0	65.3	46.9	39.9	62.5	42.8	44.1M	245G
Swin-S [27]	46.0	68.2	50.2	41.6	65.1	44.8	48M	264G
CycleMLP-B2 † [98]	41.7	63.6	45.8	38.2	60.4	41.0	46.5M	-
AS-MLP-S [95]	46.0	67.5	50.7	41.5	64.6	44.5	48M	260G

Table 8: The semantic segmentation results of different backbones on the ADE20K validation set [119]. Semantic FPN [117] and UperNet [120] frameworks are employed. Semantic FPN results are from [98], and UperNet results are from [95].

Backbone	Semantic FPN		UperNet	
	Params	mIoU (%)	Params	mIoU (%)
ResNeXt101-64x4d [19]	86.4M	40.2	-	-
PVT-Large [30]	65.1M	42.1	-	-
Swin-S [27]	53.2M	45.2	81M	49.5
CycleMLP-B4 [98]	55.6M	45.1	-	-
AS-MLP-S [95]	-	-	81M	49.2

5.3 Discussions

MLP-based network architecture gradually evolves from single stage to pyramid, with smaller and smaller patch size and higher feature fineness. The development of the MLP architecture towards a pyramid shape is actually a compromise for computational complexity, which indicates that the computational capability of the moment is not yet able to handle copious small patches. Empirical results show that the performance of the single-stage model is weaker than that of the pyramidal model. Besides, the pyramid MLP achieves performance comparable to Swin Transformer for both classification and downstream tasks. However, none of these high-performing pyramid models are pure MLP models, i.e., they are CNN-like models that combine feature map shifting and channel projection. There is not yet a backbone for a pure MLP architecture, due to the unresolved conflict between the image resolution sensitivity and the global receptive field of Token-mixing MLP, as well as the high computational complexity. For the time being, it is also difficult to apply pure MLP models to more complex tasks. To this end, we still need to make effort on the design of the MLP-based network architecture today.

6 Summary and Outlook

As the history of computer vision has proven, the availability of larger datasets combined with increased computing capacity often leads to a paradigm shift. And within these paradigm shifts, there is a gradual reduction in human intervention, i.e., remove hand-crafted visual features and inductive biases from models and rely further on learning from raw data. From manual features to CNNs, the hand-designed choices from hard-wired features are replaced with flexible and trainable architectures. From CNN-based to Transformer-based, we have seen the models' receptive field expand step by step, and the spatial range considered when encoding features is getting larger and larger. From Transformer to deep MLP, we no longer use similarity as the weight matrix, but let the model learn the weights from the raw data itself. The latest MLP works all seem to suggest that *Deep MLPs are making a strong comeback as the new paradigm*. Stay calm during the craze, we see compromises in the latest MLP development:

(1) The latest proposed deep MLP-based models use patch partition instead of whole input flattening, as the compromise for computation cost. This allows pixel-level full connectivity and global receptive field to be approximated at the patch level. The patch partition forms a two-dimensional matrix $hw \times Cp^2$, instead of a one-dimensional vector $1 \times HWC$ as in the whole input flattening, where $H = ph, W = pw$ are input resolution, p is the patch size, and C is the input channel. Thence, it is necessary for fully connected projection to be performed both in space and

channels. This is in fact an orthogonal decomposition of traditional fully connected projection, just as the full space projection is further orthogonalized into horizontal and vertical directions.

(2) At the module design level, there are two main improvement routes. One type of MLP variants focuses on reducing computational complexity, which is also a compromise on computational power. And this reduction comes at the expense of decoupling full connectivity. Another type of variants tries to address the resolution-sensitivity problem, making it possible to transfer pre-trained models to downstream tasks. These works adopt CNN-like improvements, but the full connectivity is eroded. In these models, the receptive field evolves in the opposite direction, becoming smaller and smaller and backing to the CNN ways. Hence, it is hard to admit that they are still MLPs.

(3) At the architecture level, the traditional block-stacking pattern is also applicable to MLP. And it seems that the pyramid structure is still the best choice. The initial smaller patch size helps to obtain finer features. Note that this comparison is actually unfair. Because the initial patch of the current single-stage model is larger (16×16), and the initial patch of the pyramid model is smaller (4×4). To some extent, the pyramid structure is a compromise between small patches and low computational cost. What if a 4×4 patch partition is used in a single-stage model? Will it be worse than the pyramid model? This is unknowable. What is known is that the cost of calculations will definitely increase significantly, and it is strained for the current computing devices.

(4) The empirical results show that the pure MLP model still needs some inductive bias guidance to close the gap with CNN and Transformer at the same small to medium scale datasets. And the current volume of tens of millions of data is still insufficient for a pure MLP. Empirical results also show that no pure deep MLP model can be transferred to downstream tasks yet, except for those CNN-like MLP variants, mainly because of the unresolved problem of MLP sensitivity to image resolution.

The results of our research suggest that the current amount of data and computation capability is still not enough to support pure MLP models to learn the best. And human intervention still occupies an important place. Based on this conclusion, we would like to elaborate on the important future research directions.

Vision Tailored Designs. With the current amount of data and computation, human guidance remains important, and it seems natural to try to combine the advantages of other architectures [102]. Current deep MLP architectures may remain an either/or choice for short-range and long-range dependencies, and need further intuitions to make them more efficient for visual inputs. We believe that in the future the community should focus on how to combine short-range dependencies and long-range dependencies, rather than keeping only one or the other. This is consistent with human intuition as local details are beneficial for our understanding of individual objects, and the interactions of objects across the whole visual field remain significant for our judgments. Note that the weights of the fully connected layer are position-dependent and also correspond to the image resolution, making it difficult to transfer to downstream tasks. To sum up, we encourage the community to further rethink visual tailored design, i.e., to integrate the global receptive field (long-range dependencies) and local receptive field (short-range dependencies), while maintaining resolution insensitivity.

Hardware Efficient Designs. Moving from the lab to life, MLP-based networks can have intensive power and computation requirements, hindering their deployment on edge devices and resource-constrained environments such as mobile phone platforms. Such hardware efficient designs are currently lacking for the vision MLPs to enable their seamless deployment in resource-constrained devices. How do pure MLP models perform with low precision training and inference? How pure MLP models perform knowledge distillation? How about using Nerual Architecuter Search (NAS) [121] to design more efficient and light-weight MLP models? It will be interesting to see how these questions are answered.

MLP Dedicated Optimizer. With less inductive biases, MLP enjoys greater freedom as well as a larger solution space than that of CNNs and Transformer. We currently have difficulty finding optimal solutions for MLP, which is commonly attributed to computation power and data volume constraints. But does this have anything to do with the optimizer used? We know that SGD [122] is a good optimizer for CNNs and AdamW [123] performs well for Transformer. What is the best choice for MLP? A recent work [124] has conducted a preliminary exploration, and it investigates MLP-Mixers from the lens of loss landscape geometry. We suggest a more in-depth study and exploration of MLP optimization and design MLP dedicated optimizer.

Interpretability. Another optional direction is towards a more in-depth analysis and comparison of the filters learned by the network, as well as the resulting feature maps. Pure MLP models continue the long-lasting trend of removing hand-crafted visual features and inductive biases from models and rely further on learning from raw data. What follows is that the interpretability of the model is getting lower and lower. Both mathematical explanations and visual analysis are possible helpful to understand what neural networks can freely learn from massive amounts of raw data with fewer priors. This can assist in proving whether some of the past artificial priors are correct or incorrect, and potentially guide

the design choices of future networks. In addition, the theoretical understanding of why network might be vulnerable is also a key topic.

Self-supervised Learning Methods. Pure MLP models require a large amount of training data and are easy to over-fit on small data sets. For these reasons, self-supervised pre-training would be very useful. Pretraining helps generalization and the very limited information in the labels is used only to slightly adjust the weights found by pretraining. Considering the computation cost, contrastive methods seem to be better than generative methods. At present, many contrastive learning frameworks [125, 126] are aimed at CNNs, and the similarity of feature vectors is used as a training guide. Are the previous comparative learning methods still effective for pure MLP models? Can a better self-supervised training method be designed for MLP? In addition, can the energy-based pre-training method [13] proposed by Hinton be applied again? We believe that self-supervised learning will be an excellent boost to accelerate the development of MLP models.

7 Conclusion

The proposed pure deep MLP model has stirred up a lot of interest in the vision community, and promoted the community to think about *is the convolution layer or attention layer even necessary*? This survey provides a unique view of the question, that is, *whether we are now ready for a new paradigm shift*? In search of the answer, we provide a detailed review of the latest developments in deep MLP. We first cover the structure and design bottlenecks of the deep MLP pioneering models and later provide a careful analysis of various variants proposed in the subsequent months, from module design to network architecture. We note that the latest deep MLPs compromise on computational complexity, and the latest development direction of deep MLP is back on the path of CNNs, which leads us to believe that manual guidance (inductive bias) is still important at the current computational and data volumes. We systematically highlight the key strengths and limitations of the existing deep MLP models, especially unresolved resolution sensitivity, and then particularly elaborate on the important future research directions. We hope this effort will drive further interest in the vision community to explore vision tailored design networks within the current data and computational constraints, i.e., the combination of global receptive field and local receptive field, while remaining resolution-insensitive. We also expect pure MLP to shine in the future on platforms with more training data and greater computational capability.

References

[1] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

[2] Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY, 1961.

[3] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[4] Allan Pinkus. Approximation theory of the mlp model in neural networks. *Acta numerica*, 8:143–195, 1999.

[5] Eric B Baum. On the capabilities of multilayer perceptrons. *Journal of complexity*, 4(3):193–215, 1988.

[6] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[7] Ken-Ichi Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural networks*, 2(3):183–192, 1989.

[8] M Stinchombe. Universal approximation using feed-forward networks with nonsigmoid hidden layer activation functions. *Proc. IJCNN, Washington, DC, 1989*, pages 161–166, 1989.

[9] Federico Girosi and Tomaso Poggio. Networks and the best approximation property. *Biological cybernetics*, 63(3):169–176, 1990.

[10] Paul Werbos. Beyond regression:" new tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*, 1974.

[11] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[12] Dan Claudiu Cireşan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12):3207–3220, 2010.

[13] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[14] 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.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

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

[17] 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.

[18] 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.

[19] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[20] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[21] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019.

[22] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[23] 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.

[24] Mingxing Tan and Quoc V Le. Efficientnetv2: Smaller models and faster training. *arXiv preprint arXiv:2104.00298*, 2021.

[25] 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. *arXiv preprint arXiv:2010.11929*, 2020.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[27] 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.

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

[29] 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.

[30] 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.

[31] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvtv2: Improved baselines with pyramid vision transformer. *arXiv preprint arXiv:2106.13797*, 2021.

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

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

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

[35] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7(2):187–199, 2021.

[36] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, et al. Mlp-mixer: An all-mlp architecture for vision. *arXiv preprint arXiv:2105.01601*, 2021.

[37] 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.

[38] 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.

[39] Meng-Hao Guo, Zheng-Ning Liu, Tai-Jiang Mu, Dun Liang, Ralph R Martin, and Shi-Min Hu. Can attention enable mlps to catch up with cnns? *arXiv preprint arXiv:2105.15078*, 2021.

[40] 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.

[41] Tom M Mitchell. *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research . . . , 1980.

[42] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234:11–26, 2017.

[43] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8):5455–5516, 2020.

[44] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*, 2020.

[45] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on visual transformer. *arXiv preprint arXiv:2012.12556*, 2020.

[46] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *arXiv preprint arXiv:2101.01169*, 2021.

[47] 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.

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

[49] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[50] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[51] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[52] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 558–567, 2019.

[53] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.

[54] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

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

[56] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[57] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[58] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[59] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[60] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[61] Nai-Sheng Syu, Yu-Sheng Chen, and Yung-Yu Chuang. Learning deep convolutional networks for demosaicing. *arXiv preprint arXiv:1802.03769*, 2018.

[62] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[63] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European conference on computer vision*, pages 391–407. Springer, 2016.

[64] Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Zhaowen Wang, Xinchao Wang, and Thomas Huang. Wide activation for efficient and accurate image super-resolution. *arXiv preprint arXiv:1808.08718*, 2018.

[65] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3883–3891, 2017.

[66] Gordon E Moore et al. Cramming more components onto integrated circuits, 1965.

[67] 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*, pages 843–852, 2017.

[68] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised visual transformers. *arXiv e-prints*, pages arXiv–2104, 2021.

[69] Ali Hatamizadeh, Yucheng Tang, Vishwesh Nath, Dong Yang, Andriy Myronenko, Bennett Landman, Holger Roth, and Daguang Xu. Unetr: Transformers for 3d medical image segmentation. *arXiv preprint arXiv:2103.10504*, 2021.

[70] Jeya Maria Jose Valanarasu, Poojan Oza, Ilker Hacihaliloglu, and Vishal M Patel. Medical transformer: Gated axial-attention for medical image segmentation. *arXiv preprint arXiv:2102.10662*, 2021.

[71] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12299–12310, 2021.

[72] Jiezhang Cao, Yawei Li, Kai Zhang, and Luc Van Gool. Video super-resolution transformer. *arXiv preprint arXiv:2106.06847*, 2021.

[73] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1833–1844, 2021.

[74] Zhendong Wang, Xiaodong Cun, Jianmin Bao, and Jianzhuang Liu. Uformer: A general u-shaped transformer for image restoration. *arXiv preprint arXiv:2106.03106*, 2021.

[75] Ahmed Aldahdooh, Wassim Hamidouche, and Olivier Deforges. Reveal of vision transformers robustness against adversarial attacks. *arXiv preprint arXiv:2106.03734*, 2021.

[76] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. *arXiv preprint arXiv:2103.14586*, 2021.

[77] Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. *arXiv preprint arXiv:2104.02610*, 2021.

[78] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *arXiv preprint arXiv:2105.10497*, 2021.

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

[80] 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.

[81] Peng Gao, Jiasen Lu, Hongsheng Li, Roozbeh Mottaghi, and Aniruddha Kembhavi. Container: Context aggregation network. `https://github.com/allenai/container`, 2021.

[82] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[83] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[84] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[85] Shi-Min Hu, Dun Liang, Guo-Ye Yang, Guo-Wei Yang, and Wen-Yang Zhou. Jittor: a novel deep learning framework with meta-operators and unified graph execution. *Science China Information Sciences*, 63(12):1–21, 2020.

[86] Ruiyang Liu. Jittor-mlp. `https://github.com/liuruiyang98/Jittor-MLP`, 2021.

[87] 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*, 32:8026–8037, 2019.

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

[89] Qibin Hou, Zihang Jiang, Li Yuan, Ming-Ming Cheng, Shuicheng Yan, and Jiashi Feng. Vision permutator: A permutable mlp-like architecture for visual recognition. *arXiv preprint arXiv:2106.12368*, 2021.

[90] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020.

[91] Chuanxin Tang, Yucheng Zhao, Guangting Wang, Chong Luo, Wenxuan Xie, and Wenjun Zeng. Sparse mlp for image recognition: Is self-attention really necessary? *arXiv preprint arXiv:2109.05422*, 2021.

[92] Yuki Tatsunami and Masato Taki. Raftmlp: Do mlp-based models dream of winning over computer vision? *arXiv preprint arXiv:2108.04384*, 2021.

[93] Tan Yu, Xu Li, Yunfeng Cai, Mingming Sun, and Ping Li. S2-mlp: Spatial-shift mlp architecture for vision. *arXiv preprint arXiv:2106.07477*, 2021.

[94] Tan Yu, Xu Li, Yunfeng Cai, Mingming Sun, and Ping Li. S2-mlpv2: Improved spatial-shift mlp architecture for vision. *arXiv preprint arXiv:2108.01072*, 2021.

[95] Dongze Lian, Zehao Yu, Xing Sun, and Shenghua Gao. As-mlp: An axial shifted mlp architecture for vision. *arXiv preprint arXiv:2107.08391*, 2021.

[96] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[97] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 1451–1460. IEEE, 2018.

[98] Shoufa Chen, Enze Xie, Chongjian Ge, Ding Liang, and Ping Luo. Cyclemlp: A mlp-like architecture for dense prediction. *arXiv preprint arXiv:2107.10224*, 2021.

[99] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[100] Jianyuan Guo, Yehui Tang, Kai Han, Xinghao Chen, Han Wu, Chao Xu, Chang Xu, and Yunhe Wang. Hire-mlp: Vision mlp via hierarchical rearrangement. *arXiv preprint arXiv:2108.13341*, 2021.

[101] Jiachen Li, Ali Hassani, Steven Walton, and Humphrey Shi. Convmlp: Hierarchical convolutional mlps for vision. *arXiv preprint arXiv:2109.04454*, 2021.

[102] Jihao Liu, Hongsheng Li, Guanglu Song, Xin Huang, and Yu Liu. Uninet: Unified architecture search with convolution, transformer, and mlp. *arXiv preprint arXiv:2110.04035*, 2021.

[103] Yuxuan Lou, Fuzhao Xue, Zangwei Zheng, and Yang You. Sparse-mlp: A fully-mlp architecture with conditional computation. *arXiv preprint arXiv:2109.02008*, 2021.

[104] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[105] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *arXiv preprint arXiv:2106.05974*, 2021.

[106] Tan Yu, Xu Li, Yunfeng Cai, Mingming Sun, and Ping Li. Rethinking token-mixing mlp for mlp-based vision backbone. *arXiv preprint arXiv:2106.14882*, 2021.

[107] Charles Scott Sherrington. Observations on the scratch-reflex in the spinal dog. *The Journal of physiology*, 34(1-2):1–50, 1906.

[108] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[109] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[110] Philipp Benz, Soomin Ham, Chaoning Zhang, Adil Karjauv, and In So Kweon. Adversarial robustness comparison of vision transformer and mlp-mixer to cnns. *arXiv preprint arXiv:2110.02797*, 2021.

[111] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[112] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[113] Katelyn Morrison, Benjamin Gilby, Colton Lipchak, Adam Mattioli, and Adriana Kovashka. Exploring corruption robustness: Inductive biases in vision transformers and mlp-mixers. *arXiv preprint arXiv:2106.13122*, 2021.

[114] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[115] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[116] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[117] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6399–6408, 2019.

[118] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7036–7045, 2019.

[119] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

[120] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 418–434, 2018.

[121] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European conference on computer vision (ECCV)*, pages 19–34, 2018.

[122] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[123] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018.

[124] Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pretraining or strong data augmentations. *arXiv preprint arXiv:2106.01548*, 2021.

[125] Ankesh Anand. Contrastive self-supervised learning, 2020. `https://ankeshanand.com/blog/2020/01/26/contrative-self-supervised-learning.html`.

[126] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2021.