# MSG-Transformer: Exchanging Local Spatial Information by Manipulating Messenger Tokens

Jiemin Fang[1,2], Lingxi Xie[3], Xinggang Wang[2†], Xiaopeng Zhang[3], Wenyu Liu[2], Qi Tian[3]
[1]Institute of Artificial Intelligence, Huazhong University of Science & Technology
[2]School of EIC, Huazhong University of Science & Technology [3]Huawei Inc.
{jaminfong, xgwang, liuwy}@hust.edu.cn
{198808xc, zxphistory}@gmail.com tian.qi1@huawei.com

## Abstract

*Transformers have offered a new methodology of designing neural networks for visual recognition. Compared to convolutional networks, Transformers enjoy the ability of referring to global features at each stage, yet the attention module brings higher computational overhead that obstructs the application of Transformers to process high-resolution visual data. This paper aims to alleviate the conflict between efficiency and flexibility, for which we propose a specialized token for each region that serves as a messenger (MSG). Hence, by manipulating these MSG tokens, one can flexibly exchange visual information across regions and the computational complexity is reduced. We then integrate the MSG token into a multi-scale architecture named MSG-Transformer. In standard image classification and object detection, MSG-Transformer achieves competitive performance and the inference on both GPU and CPU is accelerated. Code is available at https://github.com/hustvl/MSG-Transformer.*

## 1. Introduction

The past decade has witnessed the convolutional neural networks (CNNs) dominating the computer vision community. As one of the most popular models in deep learning, CNNs construct a hierarchical structure to learn visual features, and in each layer, local features are aggregated using convolutions to produce features of the next layer. Though simple and efficient, this mechanism obstructs the communication between features that are relatively distant from each other. To offer such an ability, researchers propose to replace convolutions by the Transformer, a module which is first introduced in the field of natural language processing [50]. It is shown that Transformers have the potential to learn visual representations and achieve remarkable success in a wide range of visual recognition problems including image classification [13, 39], object detection [4], semantic segmentation [61], *etc*.

The Transformer module works by using a token to formulate the feature at each spatial position. The features are then fed into self-attention computation and each token, according to the vanilla design, can exchange information with all the others at every single layer. This design facilitates the visual information to exchange faster but also increases the computational complexity, as the computational complexity grows quadratically with the number of tokens – in comparison, the complexity of a regular convolution grows linearly. To reduce the computational costs, researchers propose to compute attention in local windows of the 2D visual features. However constructing local attention within overlapped regions enables communications between different locations but causes inevitable memory waste and computation cost; computing attention within non-overlapped regions impedes information communications. As two typical local-attention vision Transformer methods, HaloNet [49] partitions query features without overlapping but overlaps key and value features by slightly increasing the window boundary; Swin Transformer [31] builds implicit connections between windows by alternatively changing the partition style in different layers, *i.e*., shifting the split windows. These methods achieve competitive performance compared to vanilla Transformers, but HaloNet still wastes memories and introduces additional cost in the key and value; Swin Transformer relies on frequent 1D-2D feature transitions, which increase the implementation difficulty and additional latency.

To alleviate the burden, this paper presents a new methodology towards more efficient exchange of information. This is done by constructing a **messenger (MSG) token** in each local window. Each MSG token takes charge of summarizing information in the corresponding window

†Corresponding author.
The work was done during Jiemin Fang's internship at Huawei Inc.

and exchange it with other MSG tokens. In other words, all regular tokens are not explicitly connected to other regions, and MSG tokens serve as the hub of information exchange. This brings two-fold benefits. First, our design is friendly to implementation since it does not create redundant copies of data like [39, 49, 60]. Second and more importantly, the flexibility of design is largely improved. By simply manipulating the MSG tokens (*e.g.*, adjusting the coverage of each messenger token or programming how they exchange information), one can easily construct many different architectures for various purposes. Integrating the Transformer with MSG tokens into a multi-scale design, we derive a powerful architecture named **MSG-Transformer** that takes advantages of both multi-level feature extraction and computational efficiency.

We instantiate MSG-Transformer as a straightforward case that the features of MSG tokens are shuffled and reconstructed with splits from different locations. This can effectively exchange information from local regions and delivered to each other in the next attention computation, while the implementation is easy yet efficient. We evaluate the models on both image classification and object detection, which achieve promising performance. We expect our efforts can further ease the research and application of multi-scale/local-attention Transformers for visual recognition.

We summarize our contributions as follows.

- We propose a new local-attention based vision Transformer with hierarchical resolutions, which computes attention in non-overlapped windows. Communications between windows are achieved via the proposed MSG tokens, which avoid frequent feature dimension transitions and maintain high concision and efficiency. The proposed shuffle operation effectively exchanges information from different MSG tokens with negligible cost.

- In experiments, MSG-Transformers show promising results on both ImageNet [10] classification, *i.e.*, 84.0% Top-1 accuracy, and MS-COCO [28] object detection, *i.e.*, 52.8 mAP, which consistently outperforms recent state-of-the-art Swin Transformer [31]. Meanwhile, due to the concision for feature process, MSG-Transformer shows speed advantages over Swin Transformer, especially on the CPU device.

- Not directly operating on the enormous patch tokens, we propose to use the lightweight MSG tokens to exchange information. The proposed MSG tokens effectively extract features from local regions and may have potential to take effects for other scenarios. We believe our work will be heuristic for future explorations on vision Transformers.

## 2. Related Works

**Convolutional Neural Networks** CNNs have been a popular and successful algorithm in a wide range of computer vision problems. As AlexNet [26] shows strong performance on ImageNet [10] classification, starting the blooming development of CNNs. A series of subsequent methods [17,21,42,44,45] emerge and persist in promoting CNN performance on vision tasks. Benefiting from the evolving of backbone networks, CNNs have largely improved the performance of various vision recognition scenarios including object detection [3,29,30,40,41], semantic/instance segmentation [6, 7, 16], *etc*. As real-life scenarios usually involve resource-constrained hardware platforms (*e.g.*, for mobile and edge devices), CNNs are designed to take less computation cost [19, 34, 46]. Especially, with NAS approaches [2, 14, 53, 62] applied, CNNs achieved high performance with extremely low cost, *e.g.*, parameter number, FLOPs and hardware latency. A clear drawback of CNNs is that it may take a number of layers for distant features to communicate with each other, hence limiting the ability of visual representation. Transformers aim to solve this issue.

**Vision Transformer Networks** Transformers, first proposed by [50], have been widely used in natural language processing (NLP). The variants of Transformers, together with improved frameworks and modules [1, 11], have occupied most state-of-the-art (SOTA) performance in NLP. The core idea of Transformers lies in the self-attention mechanism, which aims at building relations between local features. Some preliminary works [20, 23, 39, 52, 60] explore to apply self-attention to networks for vision tasks and have achieved promising effects. Recently, ViT [13] proposes to apply a pure Transformer to image patch sequences, which matches or even outperforms the concurrent CNN models on image classification. Inspired by ViT, a series of subsequent works [9, 15, 47, 48, 58] explore better designs of vision Transformers and achieve promising promotion. Some works [27, 43, 54, 56] integrated modules from CNNs into vision Transformer networks and also achieve great results. In order to achieve strong results on image classification, many of the above ViT-based methods process features under a constant resolution and compute attentions within a global region. This makes it intractable to apply vision Transformers to downstream tasks, *e.g.*, object detection and semantic segmentation, as multi-scale objects are hard to be represented under a constant resolution, and increased input resolutions cause overloaded computation/memory cost for attention computation.

**Downstream-friendly Vision Transformers** To apply vision Transformers to downstream tasks, two key issues need to be solved, *i.e.*, involving hierarchical resolutions to capture elaborate multi-scale features and decreasing cost

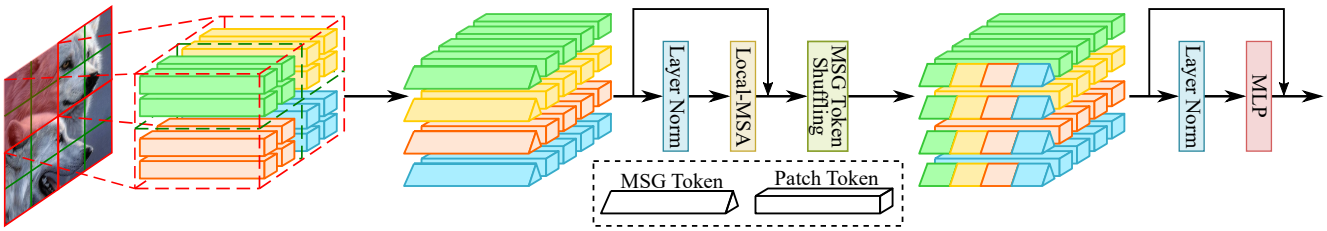



Figure 1. Structure of the MSG-Transformer block. The 2D features are split into local windows (by green lines), and several windows compose a shuffle region (the red one). Each local window is attached with one MSG token. MSG tokens are shuffled to exchange information in each Transformer block and deliver the obtained information to patch tokens in the next self-attention.

brought by global attention computation. PVT [51] proposed to process features under multi-resolution stages and down-samples key and value features to decrease the computation cost. HaloNet [49] and Swin Transformer [31] propose to compute attention in a local window. To overcome the contradiction that non-overlapped windows lack communication while overlapped windows introduce additional memory/computation cost, HaloNet proposes to slightly overlap features in the key and value tokens but leave the query non-overlapped; Swin Transformer alternatively changes the window partition style to implicitly build connections between non-overlapped windows. A series of subsequent works [8, 12, 22, 57] explore new methods for building local-global relations or connecting local regions. We newly propose MSG tokens to extract information from local windows, and use a lightweight method, *i.e.*, shuffle, to exchange information between MSG tokens. This concise manner avoids direct operation on cumbersome patch tokens and shows high flexibility.

## 3. The MSG-Transformer

This section elaborates the proposed approach, MSG-Transformer. The core part is Sec. 3.1 where we introduce the MSG token and explain how it works to simplify information exchange. Then, we construct the overall architecture (*i.e.*, the MSG-Transformer) in Sec. 3.2 and analyze the complexity in Sec. 3.3.

### 3.1. Adding MSG Tokens to a Transformer Block

The MSG-Transformer architecture is constructed by stacking a series of MSG-Transformer blocks, through various spatial resolutions. As shown in Fig. 1, a MSG-Transformer block mainly composes of several modules, *i.e.*, layer normalization (layer norm), local multi-head self-attention (local-MSA), MSG token shuffling and MLP.

Fig. 1 presents how features from a local spatial region are processed. First, the 2D features $X \in \mathbb{R}^{H\times W\times C}$ are divided into non-overlapped windows (by green lines in Fig. 1) as $X_w \in \mathbb{R}^{\frac{H}{w}\times\frac{W}{w}\times w^2\times C}$, where $(H, W)$ denotes the 2D resolution of the features, $C$ denotes the channel dimension, and $w$ denotes the window size. Then $R\times R$ windows compose a shuffle region (boxed in red lines in Fig. 1), namely features are split as $X_r \in \mathbb{R}^{\frac{H}{Rw}\times\frac{W}{Rw}\times R^2\times w^2\times C}$, where $R$ denotes the shuffle region. In vision Transformers [13, 47], image features are commonly projected into patch tokens by the input layer. Besides the patch tokens, which represent the intrinsic information of the images, we introduce an additional token, named messenger (MSG) token, to abstract information from patch tokens in a local window. Each local window is attached with one MSG token as $X'_w \in \mathbb{R}^{\frac{H}{Rw}\times\frac{W}{Rw}\times R^2\times(w^2+1)\times C}$. Then a layer normalization is applied on all the tokens. The multi-head self-attention is performed within each local window between both patch and MSG tokens. MSG tokens can capture information from the corresponding windows with attention. Afterwards, all the MSG tokens $T_{\text{MSG}} \in \mathbb{R}^{\frac{H}{Rw}\times\frac{W}{Rw}\times R^2\times C}$ from a same local region $R\times R$ are shuffled to exchange information from different local windows. We name a region with MSG tokens shuffled as the *shuffle region*. Finally, tokens are processed by a layer normalization and a two-layer MLP.

The whole computing procedure of a MSG-Transformer block can be summarized as follows.

$$X'_w = [T_{\text{MSG}}; X_w] \tag{1}$$
$$X'_w = \text{Local-MSA}(LN(X'_w)) + X'_w \tag{2}$$
$$T_{\text{MSG}} = \text{shuffle}(T_{\text{MSG}}) \tag{3}$$
$$X'_w = \text{MLP}(LN(X'_w)) + X'_w \tag{4}$$

**Local Multi-head Self-Attention** Different from previous vision Transformers [13, 47] which performer attention computation along the global region, we compute self-attention within each local window. Taking a window of $w \times w$ for example, the attention is computed on the token sequence of $X = [t_{\text{MSG}}; x_1; ...; x_{w^2}]$, where $t_{\text{MSG}}$ denotes the MSG token associated with this window and $x_i (1 \le i \le w^2)$ denotes each patch token within the window.

$$\text{Attention}(Q, K, V) = \text{softmax}(Q \cdot K^T/\sqrt{d} + B) \cdot V, \tag{5}$$

where $Q, K, V \in \mathbb{R}^{(w^2+1)\times d}$ denotes the query, key and value matrices projected from sequence $X$ respectively, $d$

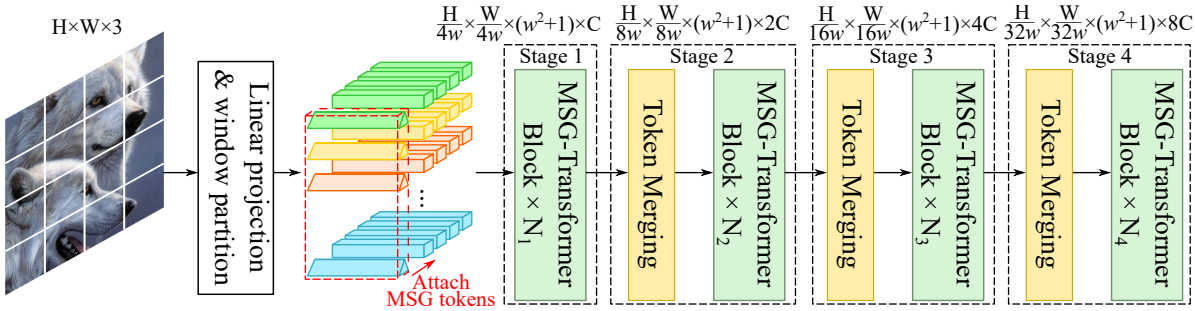



Figure 2. Overall architecture of MSG-Transformer. Patches from the input image are projected into tokens, and token features are partitioned into windows. Then each window is attached with one MSG token, which will participate in subsequent attention computation with all the other patch tokens within the local window in every layer.

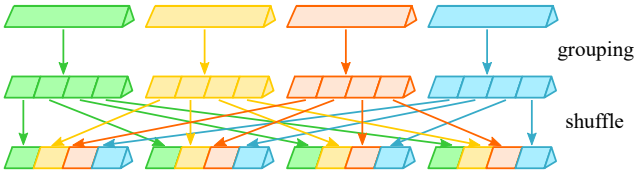



Figure 3. Shuffling MSG tokens, where we inherit the example in Fig. 1 for illustration.

denotes the channel dimension, and $B$ denotes the relative position biases. Following previous Transformer works [20, 31, 38], the relative position biases between patch tokens in $B$ are taken from the bias parameter $b^{rel} \in \mathbb{R}^{(2w-1)\times(2w-1)}$ according to the relative token distances. The position biases between patch tokens and the MSG token $t_{\text{MSG}}$ is all set as equal, which is the same as the manner dealing with the [CLS] token in [24]. Specifically, matrix $B$ are computed as

$$B = \begin{cases} b^{rel}_{i',j'} & i \neq 0, j \neq 0 \\ \theta_1 & i = 0 \\ \theta_2 & i \neq 0, j = 0 \end{cases}, \tag{6}$$

where $i' = i \bmod w - j \bmod w + w - 1, j' = i//w - j//w + w - 1, \theta_1, \theta_2$ are two learnable parameters.

**Exchanging Information by Shuffling MSG Tokens** The MSG tokens allow us to exchange visual information flexibly. Here we instantiate an example using the shuffling operation, while we emphasize that the framework easily applies to other operations (see the next paragraph). In each MSG-Transformer block, MSG tokens in a same shuffle region are shuffled to exchange information from different local windows. Assuming the shuffle region has a size of $R \times R$, it means there exist $R \times R$ MSG tokens in this region and each MSG token is associated with a $w \times w$ local window. As shown in Fig 3, channels of each MSG token are first split into $R \times R$ groups. Then the groups from $R \times R$ MSG tokens are recombined. With shuffle finished, each MSG token obtains information from all the other ones.

With the next attention computing performed, spatial information from the other local windows is delivered to patch tokens in the current window via the MSG token. Denoting MSG tokens in a $R \times R$ shuffle region as $T_{\text{MSG}} \in \mathbb{R}^{R^2 \times d}$, the shuffle process can be formulated as

$$\begin{aligned} T'_{\text{MSG}} &= \text{reshape}(T_{\text{MSG}}),\ T'_{\text{MSG}} \in \mathbb{R}^{R^2 \times R^2 \times \frac{d}{R^2}} \\ T'_{\text{MSG}} &= \text{transpose}(T'_{\text{MSG}}, \text{dim}_0 = 0, \text{dim}_1 = 1), \\ T_{\text{MSG}} &= \text{reshape}(T'_{\text{MSG}}),\ T_{\text{MSG}} \in \mathbb{R}^{R^2 \times d} \end{aligned} \tag{7}$$

where $d$ denotes the channel dimension of the MSG token, which is guaranteed to be divisible by the group number, $R^2$.

Though the shuffle operation has the similar manner with that in convolutional network ShuffleNet [34, 59], the effect is entirely different. ShuffleNet performs the shuffle operation to fuse separated channel information caused by the grouped $1 \times 1$ convolution, while our MSG-Transformer shuffles the proposed MSG tokens to exchange spatial information from different local windows.

**Extensions** There exist other ways of constructing and manipulating MSG tokens. For example, one can extend the framework so that neighboring MSG tokens can overlap, or program the propagation rule so that the MSG tokens are not fully-connected to each other. Besides, one can freely inject complex operators, rather than shuffle-based identity mapping, when the features of MSG tokens are exchanged. Note that some of these functions are difficult to implement without taking MSG tokens as the explicit hub. We will investigate these extensions in the future.

### 3.2. Overall Architecture

Fig. 2 shows the overall architecture of MSG-Transformer. The input image is first projected into patch tokens $T_p \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times C}$ by a $7 \times 7$ convolution with stride 4, where $C$ denotes the channel dimension. The overlapped projection is used for building better relations between patch tokens. Similar manners have also been adopted in

Table 1. Detailed settings for MSG-Transformer architecture variants. 'cls' denotes image classification on ImageNet. 'det' denotes object detection on MS-COCO. 'dim' denotes the embedding channel dimension. '#head' and '#blocks' denote the number of self-attention heads and MSG-Transformer blocks in each stage.

| Stage | Patch Token Resolution | Shuffle Size | | MSG-Transformer-T | | | MSG-Transformer-S | | | MSG-Transformer-B | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | cls | det | dim | #heads | #blocks | dim | #heads | #blocks | dim | #heads | #blocks |
| 1 | $\frac{H}{4} \times \frac{W}{4}$ | 4 | 4 | 64 | 2 | 2 | 96 | 3 | 2 | 96 | 3 | 2 |
| 2 | $\frac{H}{8} \times \frac{W}{8}$ | 4 | 4 | 128 | 4 | 4 | 192 | 6 | 4 | 192 | 6 | 4 |
| 3 | $\frac{H}{16} \times \frac{W}{16}$ | 2 | 8 | 256 | 8 | 12 | 384 | 12 | 12 | 384 | 12 | 28 |
| 4 | $\frac{H}{32} \times \frac{W}{32}$ | 1 | 4 | 512 | 16 | 4 | 768 | 24 | 4 | 768 | 24 | 4 |

previous methods [8,54]. Then the tokens are split into windows with the shape of $w \times w$, and each window is attached with one MSG token, which has an equal channel number with the patch token. The rest part of the architecture is constructed by stacking a series of MSG-Transformer blocks as defined in Sec. 3.1. To obtain features under various spatial resolutions, we downsample features by merging both patch and MSG tokens. Blocks under the same resolution form a stage. For both patch and MSG tokens, we use an overlapped $3 \times 3$ convolution with stride 2 to perform token merging and double the channel dimension in the next stage[1]. For image classification, the finally merged MSG tokens are projected to produce classification scores. And for downstream tasks like object detection, only patch tokens are delivered into the head structure while MSG tokens only serve for exchanging information in the backbone.

In our implementation, we build three architecture variants with different scales. As shown in Tab. 1, MSG-Transformer-T, -S and -B represent tiny, small, and base architectures with different channel numbers, attention head numbers and layer numbers. The window size is set as 7 for all architectures. The shuffle region size is set as $4, 4, 2, 1$ in four stages respectively for image classification and $4, 4, 8, 4$ for object detection. As demonstrated in subsequent studies (Sec. 4.3), our MSG-Transformer prefers deeper and narrower architecture scales than Swin Transformer [31].

### 3.3. Complexity Analysis

Though introduced one MSG token in each local window, the increased computational complexity is negligible. The local attention-based Transformer block includes two main part, *i.e.*, local-MSA and two-layer MLP. Denoting the input patch token features as $T_p \in \mathbb{R}^{\frac{H}{w} \times \frac{W}{w} \times w^2 \times C}$, where $H, W$ denote the 2D spatial resolution, $w$ denotes the local window size, and $C$ denotes the channel number, the total FLOPs are computed as

$$\begin{aligned} \text{FLOPs} &= \text{FLOPs}_{MSA} + \text{FLOPs}_{MLP} \\ &= \frac{HW}{w^2} \times (4w^2C^2 + 2w^4C) + 2\frac{HW}{w^2}w^2 \cdot 4C^2. \end{aligned} \tag{8}$$

With the MSG tokens applied, the total FLOPs change to

$$\begin{aligned} \text{FLOPs}' = &\frac{HW}{w^2}(4(w^2+1)C^2 + 2(w^2+1)^2C) \\ &+ 2\frac{HW}{w^2}(w^2+1) \cdot 4C^2. \end{aligned} \tag{9}$$

The FLOPs increase proportion is computed as

$$\begin{aligned} &\frac{\text{FLOPs}' - \text{FLOPs}}{\text{FLOPs}} \\ &= \frac{\frac{HW}{w^2}(4C^2 + 2(w^2+1)C) + 2\frac{HW}{w^2} \cdot 4C^2}{\frac{HW}{w^2} \times (4w^2C^2 + 2w^4C) + 2\frac{HW}{w^2} \times w^2 \times 4C^2} \\ &= \frac{6C + w^2 + 1}{6w^2C + w^4}. \end{aligned} \tag{10}$$

As the window size $w$ is set as 7 in our implementations, the FLOPs increase proportion becomes $\frac{6C+50}{294C+7^4}$. Taking the channel number as 384 for example, the increased FLOPs only account for $\sim 2.04\%$ which are negligible to the total complexity.

For the number of parameters, all the linear projection parameters are shared between patch and MSG tokens. Only the input MSG tokens introduce additional parameters, but they are shared between shuffle regions, only taking $4^2C = 16C$, *i.e.*, $\sim 0.0015$M for the 96 input channel dimension. In experiments, we prove even with the input MSG tokens not learned, MSG-Transformers can still achieve as high performance. From this, parameters from input MSG tokens can be abandoned.

It is worth noting that due to local region communication is achieved by shuffling MSG tokens, the huge feature matrix of patch tokens only needs to be window-partitioned once in a stage if the input images have a regular size. With MSG tokens assisting, cost from frequent 2D-to-1D matrix transitions of patch tokens can be saved, which cause additional latencies especially on computation-limited devices,

[1]The convolution parameters for merging tokens are shared between patch and MSG tokens.

Table 2. Image classification performance comparisons on ImageNet-1K [10].

| Method | Input size | Params | FLOPs | Imgs/s | CPU latency | Top-1 (%) |
|---|---|---|---|---|---|---|
| **Convolutional Networks** | | | | | | |
| RegY-4G [37] | $224^2$ | 21M | 4.0G | 930.1 | 138ms | 80.0 |
| RegY-8G [37] | $224^2$ | 39M | 8.0G | 545.5 | 250ms | 81.7 |
| RegY-16G [37] | $224^2$ | 84M | 16.0G | 324.6 | 424ms | 82.9 |
| EffNet-B4 [46] | $380^2$ | 19M | 4.2G | 345 | 315ms | 82.9 |
| EffNet-B5 [46] | $456^2$ | 30M | 9.9G | 168.5 | 768ms | 83.6 |
| EffNet-B6 [46] | $528^2$ | 43M | 19.0G | 96.4 | 1317ms | 84.0 |
| **Transformer Networks** | | | | | | |
| DeiT-S [47] | $224^2$ | 22M | 4.6G | 898.3 | 118ms | 79.8 |
| T2T-ViT$_t$-14 [58] | $224^2$ | 22M | 5.2G | 559.3 | 225ms | 80.7 |
| PVT-Small [51] | $224^2$ | 25M | 3.8G | 749.0 | 146ms | 79.8 |
| TNT-S [15] | $224^2$ | 24M | 5.2G | 387.1 | 215ms | 81.3 |
| CoaT-Lite-S [56] | $224^2$ | 20M | 4.0G | - | - | 81.9 |
| Swin-T [31] | $224^2$ | 28M | 4.5G | 692.1 | 189ms | 81.3 |
| MSG-T | $224^2$ | 25M | 3.8G | 726.5 | 157ms | **82.4** |
| DeiT-B [47] | $224^2$ | 87M | 17.5G | 278.9 | 393ms | 81.8 |
| T2T-ViT$_t$-19 [58] | $224^2$ | 39M | 8.4G | 377.3 | 314ms | 81.4 |
| T2T-ViT$_t$-24 [58] | $224^2$ | 64M | 13.2G | 268.2 | 436ms | 82.2 |
| PVT-Large [51] | $224^2$ | 61M | 9.8G | 337.1 | 338ms | 81.7 |
| TNT-B [15] | $224^2$ | 66M | 14.1G | 231.1 | 414ms | 82.8 |
| Swin-S [31] | $224^2$ | 50M | 8.7G | 396.6 | 346ms | 83.0 |
| MSG-S | $224^2$ | 56M | 8.4G | 422.5 | 272ms | **83.4** |
| ViT-B/16 [13] | $384^2$ | 87M | 55.4G | 81.1 | 1218ms | 77.9 |
| ViT-L/16 [13] | $384^2$ | 307M | 190.7G | 26.3 | 4420ms | 76.5 |
| DeiT-B [47] | $384^2$ | 87M | 55.4G | 81.1 | 1213ms | 83.1 |
| Swin-B [31] | $224^2$ | 88M | 15.4G | 257.6 | 547ms | 83.3 |
| MSG-B | $224^2$ | 84M | 14.2G | 267.6 | 424ms | **84.0** |

* "Imgs/s" denotes the GPU throughput which is measured on one 32G-V100 with a batch size of 64. Noting that throughput on 32G-V100 used in our experiments is sightly lower than 16G-V100 used in some other papers.

* The CPU latency is measured with one core of Intel(R) Xeon(R) Gold 6151 CPU @ 3.00GHz.

*e.g.*, CPU and mobile devices, but are unavoidable in most previous local attention-based [31,39] or CNN-attention hybrid Transformers [9, 27, 54].

# 4. Experiments

In experiments, we first evaluate our MSG-Transformer models on ImageNet [10] classification in Sec. 4.1. Then in Sec. 4.2, we evaluate MSG-Transformers on MS-COCO [28] object detection and instance segmentation. Finally, we perform a series of ablation studies and analysis in Sec. 4.3. Besides, we provide a MindSpore [35] implementation of MSG-Transformer.

## 4.1. Image Classification

We evaluate our MSG-Transformer networks on the commonly used image classification dataset ImageNet-1K [10] and report the accuracies on the validation set in Tab. 2. Most training settings follow DeiT [47]. The AdamW [25] optimizer is used with 0.05 weight decay. The training process takes 300 epochs in total with a cosine annealing decay learning rate schedule [33] and 20-epoch linear warmup. The total batch size is set as 1024 and the initial learning rate is 0.001. The repeated augmentation [18] and EMA [36] are not used as in Swin Transformer [31].

We provide the ImageNet classification results in Tab. 2 and compare with other convolutional and Transformer networks. Compared with DeiT [32], MSG-Transformers achieve significantly better trade-offs between accuracy and computation budget. MSG-Transformer-T achieves 2.6 Top-1 accuracy promotion over DeiT-S with 0.8G smaller FLOPs; MSG-Transformer-S promotes the accuracy by 1.6 with only 48.0% FLOPs; MSG-Transformer-B achieves an 84.0% Top-1 accuracy, beating larger-resolution DeiT-B by 0.9 with only 25.6% FLOPs. Compared with the recent state-of-the-art method Swin Transformer [31], our MSG-Transformers achieve competitive accuracies with similar Params and FLOPs. It is worth noting, as frequent 1D-2D feature transitions and partition are avoided, MSG-Transformers show promising speed advantages over Swin Transformers. Especially on the CPU device, the latency improvement is more evident. MSG-Transformer-T is 16.9% faster than Swin-T; MSG-Transformer-S is 21.4% faster than Swin-S; MSG-Transformer-B is 22.5% faster than Swin-B.

Table 3. Object detection and instance segmentation performance comparisons on MS-COCO [28] with Cascade Mask R-CNN [3, 16]. "X101-32" and "X101-64" denote ResNeXt101-32×4d [55] and -64×4d respectively.

| Method | $AP^{box}$ | $AP^{box}_{50}$ | $AP^{box}_{75}$ | $AP^{mask}$ | $AP^{mask}_{50}$ | $AP^{mask}_{75}$ | Params | FLOPs | FPS |
|---|---|---|---|---|---|---|---|---|---|
| DeiT-S | 48.0 | 67.2 | 51.7 | 41.4 | 64.2 | 44.3 | 80M | 889G | - |
| ResNet-50 | 46.3 | 64.3 | 50.5 | 40.1 | 61.7 | 43.4 | 82M | 739G | 10.5 |
| Swin-T | 50.5 | 69.3 | 54.9 | 43.7 | 66.6 | 47.1 | 86M | 745G | 9.4 |
| MSG-T | **51.4** | **70.1** | **56.0** | **44.6** | **67.4** | **48.1** | 83M | 731G | 9.1 |
| X101-32 | 48.1 | 66.5 | 52.4 | 41.6 | 63.9 | 45.2 | 101M | 819G | 7.5 |
| Swin-S | 51.8 | 70.4 | 56.3 | 44.7 | 67.9 | 48.5 | 107M | 838G | 7.5 |
| MSG-S | **52.5** | **71.1** | **57.2** | **45.5** | **68.4** | **49.5** | 113M | 831G | 7.5 |
| X101-64 | 48.3 | 66.4 | 52.3 | 41.7 | 64.0 | 45.1 | 140M | 972G | 6.0 |
| Swin-B | 51.9 | 70.9 | 56.5 | 45.0 | 68.4 | 48.7 | 145M | 982G | 6.3 |
| MSG-B | **52.8** | **71.3** | **57.3** | **45.7** | **68.9** | **49.9** | 142M | 956G | 6.1 |

* FPS is measured on one 32G-V100 with a batch size of 1.

## 4.2. Object Detection

We evaluate our MSG-Transformer networks on MS-COCO [28] object detection with the Cascade Mask R-CNN [3, 16] framework. The training and evaluation are performed based on the MMDetection [5] toolkit. For training, we use the AdamW [25] optimizer with 0.05 weight decay, $1 \times 10^{-4}$ initial learning rate and a total batch size of 16. The learning rate is decayed by 0.1 at the 27 and 33 epoch. The training takes the 3× schedule, *i.e.*, 36 epochs in total. Multi-scale training with the shorter side of the im-

Table 4. Ablation studies about MSG tokens and shuffle operations on ImageNet classification.

| Row | MSG Token | Shuffle Op. | Images / s | Top1 (%) |
|---|---|---|---|---|
| **MSG-Transformer-T (depth=12)** | | | | |
| 1 | ✗ | ✗ | 720.3 | 80.2 |
| 2 | ✓ | ✗ | 702.2 | $80.5_{\uparrow 0.3}$ |
| 3 | ✓ | ✓ | 696.7 | $\mathbf{81.1}_{\uparrow 0.9}$ |
| **MSG-Transformer-S (depth=24)** | | | | |
| 4 | ✗ | ✗ | 412.9 | 81.2 |
| 5 | ✓ | ✗ | 403.9 | $81.9_{\uparrow 0.7}$ |
| 6 | ✓ | ✓ | 401.0 | $\mathbf{83.0}_{\uparrow 1.8}$ |

age resized between 480 and 800 and the longer side not exceeding 1333 is also used. As the input image size is not fixed for object detection, the patch tokens are padded with 0 to guarantee they can be partitioned by the given window size for attention computation. And the shuffle region is alternatively changed at the left-top and right-bottom locations between layers to cover more windows.

As shown in Tab. 3, MSG-Transformers achieve significantly better performance than CNN-based models, *i.e.*, 5.1 $AP^{box}$ better than ResNet-50 [17], 4.4 $AP^{box}$ better than ResNeXt101-32×4d [55], and 4.5 $AP^{box}$ better than ResNeXt101-64×4d. Even though Swin Transformers have achieved extremely high performance on object detection, our MSG-Transformers still achieve significant promotion by 0.9, 0.7, 0.9 $AP^{box}$ and 0.9, 0.8, 0.7 $AP^{mask}$ for T, S, B scales respectively.

### 4.3. Ablation Study

In this section, we perform a series of ablation studies on ImageNet-1K about the shuffling operation, MSG tokens, network scales, and shuffle region sizes[2]. We further visualize the attention map of MSG tokens for better understanding the working mechanism.

**Effects of MSG Tokens & Shuffle Operations** We study the effects of MSG tokens and shuffle operations, providing the results in Tab. 4. As shown in Row 1, with both MSG tokens and shuffle operations removed, the performance degrades by 0.9. With MSG tokens applied in Row 2, the performance is promoted by 0.3 compared to that without both. Though without shuffle operations, MSG tokens can still exchange information in each token merging (downsampling) layer, which leads to slight promotion. However, exchanging information only in token merging layers is too limited to expanding receptive fields. With the same ablation applied on a deeper network MSG-Transformer-S, the performance gap becomes significant. The Top-1 accuracy drops by 1.8 with both modules removed, and drops by 1.1 with shuffle removed. It is worth noting that both MSG tokens and shuffle operations are light enough and cause no evident throughput decay.

[2]Without specified, experiments for ablation study remove the overlapped downsampling and follow the network scales in Swin-Transformer [31] for clear and fair comparisons.

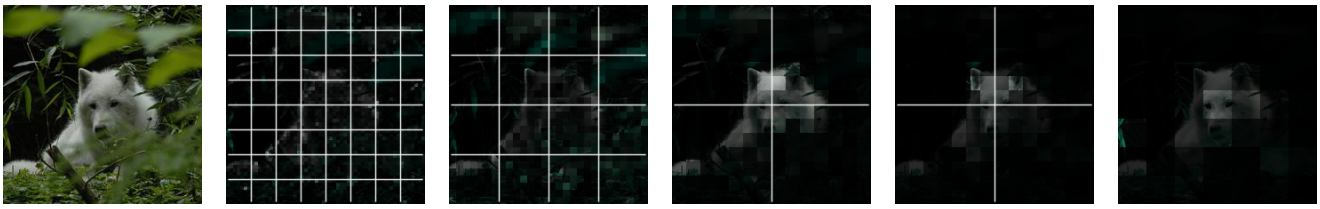
Table 5. Effects of input MSG/CLS token parameters on ImageNet classification.

| Row | Training | Evaluation | Top1 (%) |
|---|---|---|---|
| **MSG-Transformer-T (MSG Token)** | | | |
| 1 | learnable | learned | 80.9 |
| 2 | learnable | random | $80.8_{\downarrow 0.1}$ |
| 3 | random | random | $80.8_{\downarrow 0.1}$ |
| **Deit-S (CLS Token)** | | | |
| 4 | learnable | learned | 79.9 |
| 5 | learnable | random | $77.7_{\downarrow 2.2}$ |

**Input Parameters of MSG Tokens** To further understand the role MSG tokens play in Transformers, we study impacts caused by parameters of **input** MSG tokens. As shown in Row 2 of Tab. 5, we randomly re-initialize parameters of input MSG tokens for evaluation, which are learnable during training, interestingly the accuracy only drops by 0.1%. Then in Row 3, we randomly initialize input MSG tokens and keep them fixed during training. It still induces negligible accuracy drop when input MSG token parameters are also randomly re-initialized for evaluation. This implies input MSG token parameters are not so necessary to be learned. We speculate that if input parameters of CLS tokens in conventional Transformers need to be learned, and perform the same experiment on Deit-S [47]. Then we find randomly re-parameterizing input CLS tokens for evaluation leads to severe degradation to the accuracy, *i.e.*, 2.2% in Row 5.

The above experiments show that the proposed MSG tokens play a different role from conventional CLS tokens, which serve as messengers to carry information from different local windows and exchange information with each other. The input parameters of their own matter little in latter information delivering as they absorb local features layer by layer via attention computing. In other words, with uninterrupted self-attention and information exchanging, patch tokens make what MSG tokens are and MSG tokens are just responsible for summarizing local patch tokens and deliver the message to other locations. Therefore, input parameters of MSG tokens do not affect the final performance.

**Network Scales** Considering different types of architectures fit different network scales, we study the scales of both Swin- and MSG-Transformer as follows. As shown in Tab. 6, two scales are evaluated where one is shallow and wide with 96 input dimension and $[2, 2, 6, 2]$ blocks in each stage, while another one is deep and narrow with 64 dimension and $[2, 4, 12, 4]$ blocks. We observe MSG-Transformer achieves a far better trade-off between computation cost and



Input image Block-2 Block-4 Block-7 Block-10 Block-12

Figure 4. Visualization of attention maps computed between each MSG token and patch tokens within the local window in different blocks.

Table 6. Network scale study on Swin- and MSG-Transformer.

| Model | Dim | #Blocks | Params | FLOPs | Top1 (%) |
|---|---|---|---|---|---|
| Swin | 96 | [2, 2, 6, 2] | 28M | 4.5G | 81.3 |
| | 64 | [2, 4, 12, 4] | 24M | 3.6G | 81.3 |
| MSG | 96 | [2, 2, 6, 2] | 28M | 4.6G | 81.1 |
| | 64 | [2, 4, 12, 4] | 24M | 3.7G | **82.1** |

Table 7. Ablation studies about the shuffle region sizes on ImageNet classification.

| Shuffle Region Sizes | Images / s | Top1(%) |
|---|---|---|
| 2, 2, 2, 1 | 695.1 | 80.6 |
| 4, 2, 2, 1 | 696.1 | 80.8 |
| 4, 4, 2, 1 | 696.7 | **80.9** |

accuracy with the deeper-narrower scale. We analyze the reason as follows. In Swin Transformer, each patch token is involved in two different windows between layers, which requires a wider channel dimension with more attention heads to support the variety. On the contrary, MSG-Transformer uses MSG tokens to extract window-level information and transmit to patch tokens. This reduces the difficulty for patch tokens to extract information from other windows. Thus MSG-Transformer requires a smaller channel capacity to support variety in one window. A deeper and narrower architecture brings a better trade-off for MSG-Transformer.

**Shuffle Region Sizes** We study the impacts of shuffle region sizes on the final performance. As shown in Tab. 7, with the shuffle region enlarged, the final accuracy increases. It is reasonable that larger shuffle region sizes lead to larger receptive fields and are beneficial for tokens capturing substantial spatial information. Moreover, the throughput/latency is not affected by the shuffle size changing.

**Attention Map Visualization of MSG Tokens** To understand the working mechanism of MSG tokens, we visualize attention maps computed between each MSG token and its associated patch tokens within the local window in different blocks. As shown in Fig. 4, local windows in attention maps are split into grids. Though the local window size to the token features is constant, *i.e.* 7 in our settings, with tokens merged, the real receptive field is enlarged when reflected onto the original image. In shallower blocks, attention of MSG tokens is dispersive which tends to capture contour information; in deeper layers, though attention is computed within each local window, MSG tokens can still focus on locations closely related to the object.

## 5. Discussion and Conclusion

This paper proposes MSG-Transformer, a novel Transformer architecture that enables efficient and flexible information exchange. The core innovation is to introduce the MSG token which serves as the hub of collecting and propagating information. We instantiate MSG-Transformer by shuffling MSG tokens, yet the framework is freely extended by simply altering the way of manipulating MSG tokens. Our approach achieves competitive performance on standard image classification and object detection tasks with reduced implementation difficulty and faster inference speed.

**Limitations** We would analyze limitations from the perspective of the manipulation type for MSG tokens. Though shuffling is an efficient communication operation, the specificity of shuffled tokens is not so well as shuffling integrates token segments from different local windows equally on the channel dimension. On the other hand, it is valuable to explore other manipulation types with a better efficiency-specificity trade-off which may further motivate the potential of MSG-Transformer.

**Future work** Our design puts forward an open problem: since information exchange is the common requirement of deep networks, how to satisfy all of capacity, flexibility, and efficiency in the architecture design? The MSG token offers a preliminary solution, yet we look forward to validating its performance and further improving it in visual recognition tasks and beyond.

## Acknowledgement

We thank Yuxin Fang, Bencheng Liao, Liangchen Song, Yuzhu Sun and Yingqing Rao for constructive discussions and assistance. This work was in part supported by NSFC (No. 61876212 and No. 61733007) and CAAI-Huawei MindSpore Open Fund.

## References


[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. 2

[2] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *ICLR*, 2019. 2

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018. 2, 6

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1

[5] 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, and Dahua Lin. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv:1906.07155*, 2019. 6

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017. 2

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017. 2

[8] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. In *NeurIPS 2021*, 2021. 3, 5

[9] Xiangxiang Chu, Bo Zhang, Zhi Tian, Xiaolin Wei, and Huaxia Xia. Do we really need explicit position encodings for vision transformers? *arXiv e-prints*, pages arXiv–2102, 2021. 2, 6

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2, 6

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018. 2

[12] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. *arXiv:2107.00652*, 2021. 3

[13] 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. *ICLR*, 2021. 1, 2, 3, 6

[14] Jiemin Fang, Yuzhu Sun, Qian Zhang, Yuan Li, Wenyu Liu, and Xinggang Wang. Densely connected search space for more flexible neural architecture search. In *CVPR*, 2020. 2

[15] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. In *NeurIPS 2021*, 2021. 2, 6

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 2, 6

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 7

[18] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *CVPR*, 2020. 6

[19] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv:1704.04861*, 2017. 2

[20] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *ICCV*, 2019. 2, 4

[21] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 2

[22] Zilong Huang, Youcheng Ben, Guozhong Luo, Pei Cheng, Gang Yu, and Bin Fu. Shuffle transformer: Rethinking spatial shuffle for vision transformer. *arXiv:2106.03650*, 2021. 3

[23] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*, 2019. 2

[24] Guolin Ke, Di He, and Tie-Yan Liu. Rethinking positional encoding in language pre-training. In *ICLR*, 2021. 4

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 6

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 2

[27] Changlin Li, Tao Tang, Guangrun Wang, Jiefeng Peng, Bing Wang, Xiaodan Liang, and Xiaojun Chang. Bossnas: Exploring hybrid cnn-transformers with block-wisely self-supervised neural architecture search. *arXiv:2103.12424*, 2021. 2, 6

[28] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014. 2, 6

[29] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 2

[30] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 2

[31] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer:

Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1, 2, 3, 4, 5, 6, 7

[32] Zili Liu, Tu Zheng, Guodong Xu, Zheng Yang, Haifeng Liu, and Deng Cai. Training-time-friendly network for real-time object detection. *arXiv:1909.00700*, 2019. 6

[33] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017. 6

[34] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: practical guidelines for efficient CNN architecture design. In *ECCV*, 2018. 2, 4

[35] MindSpore. https://github.com/mindspore-ai/mindspore. 6

[36] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 1992. 6

[37] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020. 6

[38] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020. 4

[39] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019. 1, 2, 6

[40] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 2

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 2

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2014. 2

[43] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. *arXiv:2101.11605*, 2021. 2

[44] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 2

[45] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 2

[46] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv:1905.11946*, 2019. 2, 6

[47] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 2, 3, 6, 7

[48] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, 2021. 2

[49] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. *arXiv:2103.12731*, 2021. 1, 2, 3

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2

[51] 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:2102.12122*, 2021. 3, 6

[52] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 2

[53] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. *arXiv:1812.03443*, 2018. 2

[54] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv:2103.15808*, 2021. 2, 5, 6

[55] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 6, 7

[56] Weijian Xu, Yifan Xu, Tyler Chang, and Zhuowen Tu. Co-scale conv-attentional image transformers. *arXiv:2104.06399*, 2021. 2, 6

[57] Qihang Yu, Yingda Xia, Yutong Bai, Yongyi Lu, Alan Yuille, and Wei Shen. Glance-and-gaze vision transformer. In *NeurIPS*, 2021. 3

[58] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv:2101.11986*, 2021. 2, 6

[59] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *arXiv:1707.01083*, 2017. 4

[60] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, 2020. 2

[61] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021. 1

[62] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018. 2