# CAT: Cross Attention in Vision Transformer

**Hezheng Lin**[12*]  **Xing Cheng**[1*]  **Xiangyu Wu**[1†]  **Fan Yang**[1]
**Dong Shen**[1]  **Zhongyuan Wang**[1]  **Qing Song**[2]  **Wei Yuan**[1]

[1]MMU, KuaiShou Inc.    [2]Beijing University of Posts and Telecommunications, China

[1] {linhezheng,chengxing03,wuxiangyu,yangfan,shendong,
wangzhongyuan,yuanwei05}@kuaishou.com
[2] priv@bupt.edu.cn

## Abstract

Since Transformer has found widespread use in NLP, the potential of Transformer in CV has been realized and has inspired many new approaches. However, the computation required for replacing word tokens with image patches for Transformer after the tokenization of the image is vast(e.g., ViT), which bottlenecks model training and inference. In this paper, we propose a new attention mechanism in Transformer termed Cross Attention, which alternates attention inner the image patch instead of the whole image to capture local information and apply attention between image patches which are divided from single-channel feature maps to capture global information. Both operations have less computation than standard self-attention in Transformer. By alternately applying attention inner patch and between patches, we implement cross attention to maintain the performance with lower computational cost and build a hierarchical network called Cross Attention Transformer(CAT) for other vision tasks. Our base model achieves state-of-the-arts on ImageNet-1K, and improves the performance of other methods on COCO and ADE20K, illustrating that our network has the potential to serve as general backbones. The code and models are available at https://github.com/linhezheng19/CAT.

## 1 Introduction

With the development of deep learning and the application of convolutional neural networks[1], computer vision tasks have improved tremendously. Since 2012, CNN has dominated CV for a long time, as a crucial feature extractor in various vision tasks, and as a task branch encoder in other tasks. A variety of CNN-based networks[2–11] have different improvements and applications, and various downstream tasks also have these multiple methods, such as object detection[12–20], semantic segmentation[21–28].

Lately, Transformer[29], as a new network structure, has achieved significant results in NLP. Benefiting from its remarkable ability to extract global information, it also solves the problem that sequence models hard to be parallelized such as RNN[30] and LSTM[31], making the development of the NLP an essential leap, and also inspiring computer vision tasks.

Recent works[32–41] introduces Transformer into the computer vision as an image extractor. However, the length of the text sequence is fixed in NLP which leads to a decrease in the ability of the Transformer to process images, since the resolution of inputs are variational in different task. In processing images with Transformer, one naive approach is to treat each pixel as a token for global attention similar to work tokens. The iGPT[42] demonstrates that the computation brought by this is

---

*Interns at MMU, KuaiShou Inc.

†Corresponding author.

tremendous. Some works(e.g., ViT, iGPT) take a set of pixels in a region as a token, which reduces the computation to a certain extent. However, the computational complexity increases dramatically as the input size increases(Formula 1), and the feature maps generated in these methods are of the same shape(Figure 1(b)), making these methods unsuitable for use as the backbone of subsequent tasks.

In this paper, we are inspired by the local feature extraction capabilities of CNN, we adopt attention between pixels in one patch to simulate the characteristics of CNN, reducing the computation that increases exponentially with the input size to that is exponentially related to the patch size. Meanwhile, as Figure 3(b) shown, to consider the overall information extraction and communication of the picture, we devised a method of performing attention on single-channel feature maps. Compared with the attention on all channels, there is a significant reduction in the computation as Formula 1, and 3 demonstrated. Cross attention is performed by alternating the internal attention of the patch and the attention of single-channel feature maps. We can build a powerful backbone with the Cross Attention to generate feature maps of different scales, which satisfies the requirements of different granular features of downstream tasks, as Figure 1 shown. We introduce global attention without increasing computation or a small increase in computation, which is a more reasonable method to joint features of Transformer and CNN.

Our base model achieves 82.8% of top-1 accuracy on ImageNet-1K, which is comparable with the current CNN-based network and Transformer-based network of state-of-the-arts. Meanwhile, in other vision tasks, our CAT as the backbone in object detection and semantic segmentation methods can improve their performance.

The features of Transformer and CNN complement each other and that it is our long-term goal to combine them more efficiently and perfectly to take advantage of both. Our proposed CAT is a step in that direction, and hopefully, there will be better developments in that direction.
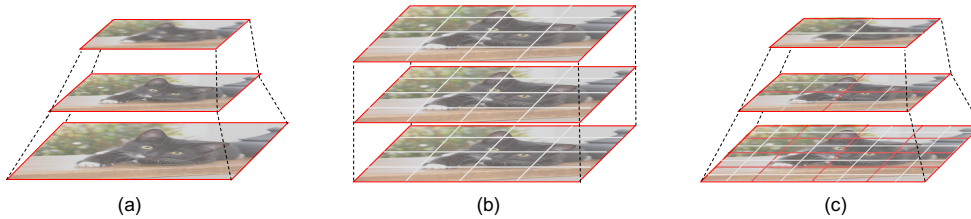


Figure 1: Hierarchical network. (a) Hierarchical networks based on CNN, different stage generates feature with variety scale. (b) Hierarchical network based on Transformer(e.g., Vit), all features are same in shape. (c) Hierarchical networks of CAT(ours), with characteristics of CNN hierarchy network.

## 2  Related work

**CNN/ CNN-based network**  CNN has the characteristics of shared weights, translation, rotation invariance, and locality, which has made great achievements in computer vision instead of the multi-layer perceptron and has become the standard network in vision tasks in last decade. As the first CNN network to achieve great success in computer vision, AlexNet laid the foundation for the later development of the CNN-based network, and [3, 8–11, 43] for performance improvement have become the choice as the backbone in vision tasks. The Inceptions[4, 5, 44–46], MobileNets[6, 47, 48], and ShuffleNets[7, 49] for efficiency improvement are also alternatives in tasks required speed of inference.

**Global attention in Transformer-based network**  Transformer is proposed in NLP for machine translation, where the core multi-head self-attention(MSA[29]) mechanism is vital in extracting the characteristics of relationships between words at multiple levels. As the first few Transformer-based backbones, ViT[33] and Deit[38] divide the image into patches (patch size is 16×16). One patch flattened as a token, and CLS-Token[50] is introduced for classification. Both CvT[35] and CeiT[51] introduce the convolutional layer to replace the linear projection of QKV[29]. CrossViT[34] integrates global features of different granularity through dividing images into different sizes of patches

for two branches. However, these methods put all patches together for MSA, only focusing on the relationship between different patches, and as the input size increases, the computational complexity increases dramatically as demonstrated in Formula 1, which is difficult to be applied to the vision tasks requiring large resolution input.

**Local attention in Transformer-based network**   The relationship between internal information of the patch is vital in vision[52, 53]. Recently, TNT[37] divides each patch into smaller patches. Through the proposed TNT block, the global information and the information inner the patch are captured. Swin[36] treats each patch as a window to extract the internal relevancy of the patch and shifted window is used to catch more features. However, the two methods have their problems. Firstly, to combine the global information interaction and local information interaction, the increase of computation could not be underestimated in [37]. Second, the interaction between local information interaction and adjacent patches lacks global information interaction in [36]. We propose a cross-patch self-attention block to effectively maintain global information interaction while avoiding the enormous increase of computation with the increase of the resolution of inputs.

**Hierarchy networks and downstream tasks**   Transformer has been used successfully for vision tasks[54–57] and NLP tasks[50, 58–63]. However, due to the consistent shape of input and output in typical Transformer, it is difficult to achieve the hierarchical structure similar to CNN-based networks[8, 9, 19, 48] which is significant in downstream tasks. FPNs[14, 20, 64, 65] combined with ResNet[8] have become the standard paradigm in object detection. In semantic segmentation[21–23, 28], the payramidal features are used to improve the performance. Recent PVT[32, 36] and Swin[36] reduce the resolution of feature in different stage similar to ResNet[8], which is also the method we used.

## 3   Method

### 3.1   Overall architecture

Our method aims to combine the attention within the patch and the attention between patches and build a hierarchical network by stacking basic blocks, which could be simply applied in other vision tasks. As shown in Figure 2(a), firstly, we reduced the input image to $H_1 = H/P, W_1 = W/P$ (where $P = 4$ in our experiments), and increase the number of channels to $C_1$ by referring to the patch processing mode in ViT[33] with patch embedding layer. Then, several CAT layers were used for feature extraction at different scales.

After the pretreatment above, the input image enters the first stage. At this point, the number of patches is $H_1/N \times W_1/N$, and the shape of the patch is $N \times N \times C_1$(where N is patch size after patch embedding layer). The shape of feature map output by stage1 is $H_1 \times W_1 \times C_1$ donated as $F_1$. Then, enter the second stage, patch projection layer to execute space to depth operation, which performs pixel block with the shape of $2 \times 2 \times C$ changes from the shape of $2 \times 2 \times C$ to the shape of $1 \times 1 \times 4C$, and then project to $1 \times 1 \times 2C$ through the linear projection layer. After entering several cross-attention blocks in the next stage and generating $F_2$ with a shape of $H_1/2 \times W_1/2 \times C_2$, the length and width of the feature map can be reduced by one time and the dimension is increased to double, similar to the operation in ResNet[8], which is also the practice in Swin[36]. After passing the four stages, we can get $\{F_1, F_2, F_3, F_4\}$, four feature maps of different scales and dimensions. Like typical CNN-based networks[8, 9], feature maps of different granularity can be provided for other downstream vision tasks.

#### 3.1.1   Inner-Patch Self-Attention Block

In computer vision, each pixel needs a specific channel to represent its different semantic features. Similar to word tokens in NLP, the ideal is to take each pixel of feature map as a token (e.g., ViT, DeiT), but the computational cost is too enormous. As Formula 1 shows, the computational complexity increases exponentially with the resolution of the input image. For instance, in the conventional RCNN series[12, 13, 67, 68] of methods, the short edge of the input is at least 800 pixels, while the YOLO series[16, 17, 69] of papers also need images of more than 500 pixels. Most of the semantic segmentation methods[21, 22, 25] also need images with 512 pixels side lengths. The
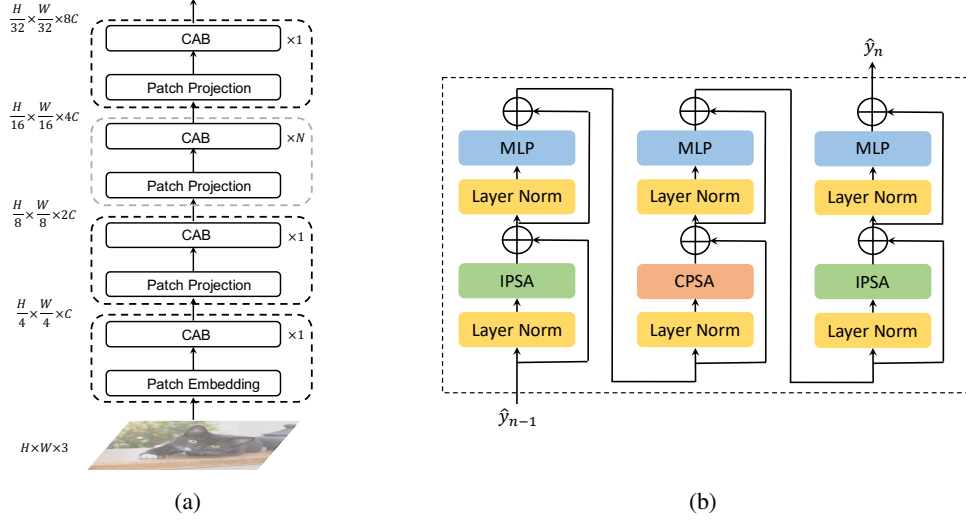
Figure 2: (a) CAT architecture, at the third stage, the number of CABs varies with the size of model. (b) Cross Attention Block(CAB), stacking IPSA and CPSA, both with LN[66], MLP, and shorcut[8].

computation cost is at least 5 times higher than that of 224 pixels in pre-training phase.

$$FLOPs_{MSA} = 4HWC^2 + 2H^2W^2C \qquad (1)$$

Inspired by the characteristics of local feature extraction of CNN, we introduce the locality of convolution method in CNN into Transformer to conduct per-pixel self-attention in each patch called Inner-Patch Self-Attention(IPSA) as shown in Figure 3(a). We treat a patch as an attention scope, rather than the whole picture. At the same time, Transformer can generate different attention-maps according to inputs, which has a significant advantage over CNN with fixed parameters, which is similar to dynamic parameters in convolutional method, and it is be proved gainful in [70]. [37] has revealed that attention between pixels is also vital. Our approach significantly reduces computation while taking into account the relationship between pixels in the patch. The formula of computation as follows:

$$FLOPs_{IPSA} = 4HWC^2 + 2N^2HWC \qquad (2)$$

where $N$ is patch size in IPSA. Compared with MSA in a standard Transformer, the computational complexity decreased from a quadratic correlation(Fornula 1) with the $H \times W$ to a linear correlation with the $H \times W$. Assume that $H, W = 56$, $C = 96$, $N = 7$, $FLOPs_{MSA} \approx 2.0\ G$ following Formula 1, and following Formula 2, $FLOPs_{IPSA} \approx 0.15\ G$, which is much fewer.

### 3.1.2 Cross-Patch Self-Attention Block

Adding the attention mechanism between pixels only ensures that the interrelationships between pixels inner one patch be caught, but the information exchange of the whole picture is also quite crucial. In CNN-based networks, a stacked convolution kernel generally practiced expanding the receptive field. Dilated/Atrous Convolution[24] is proposed for larger receptive field, and the final receptive field expands to the whole picture is expected in practice. Transformer is naturally capable of capturing global information, but efforts like ViT[33] and Deit[38] are ultimately not the best resolution.

Each single-channel feature map naturally has global spatial information. We propose Cross-Patch Self-Attention, separating each channel feature map and dividing each channel into $H/N \times W/N$ patches and using self-attention to get global information in the whole feature map. This is similar to the depth-wise separable convolution used in Xception[46] and MobileNet[6]. The computation of our method could be computed as follows:

$$FLOPs_{CPSA} = 4N^2HWC + 2(HW/N)^2C \qquad (3)$$

4

where $N$ is patch size in CPSA, $H$, $W$ represent height and width of feature map respectively. The computational cost is fewer than ViT(Formula 1) and other global attention based methods. Meanwhile, as shown in Figure 2(b), we combine with MobileNet[6] design, stacking IPSA block and CPSA block to extract and integrate features between pixels in one patch and between patches in one feature map. Compared to the shifted window in Swin[36], which is manually designed, difficult to implement, and has little ability to capture the global information, ours is reasonable and easier to comprehend. The $FLOPs_{CPSA}$ is about $0.1\,G$ computed follow the Formula 3 with the suppose same as the above section, which is much fewer than $2.0\,G$ of MSA.
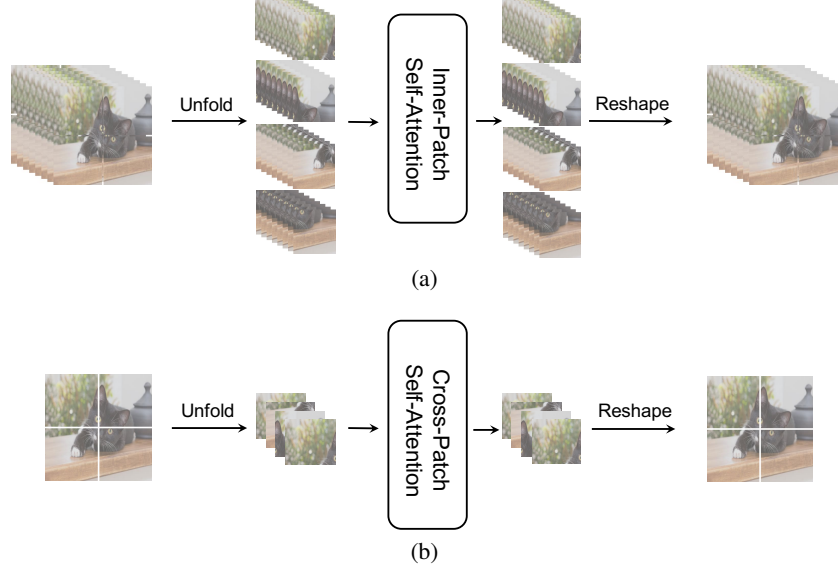


(a)



(b)

Figure 3: The pipeline of IPSA and CPSA. (a) IPSA: unfold the all-channel inputs to $2{\times}2$, and stack them, after IPSA block, reshape to original shape. (b) CPSA: unfold the single-channel input to $2 \times 2$ patches and stack them, after CPSA block, reshape to original shape.

The multi-head self-attention mechanism is proposed in [29]. Each head can notice different semantic information between words in NLP. In computer vision, each head can notice different semantic information between image patches which is similar to channels in CNN-based networks. In the CPSA, we set the number of heads as patch size making the dimension of one head equal to patch size, which is useless to performance, as presented in Table 5. So the single head is the default setting in our experiments.

**Position encoding**   We adopt relative position encoding in IPSA refer to [36, 71, 72], while for CPSA which conduct self-attention on the complete single-channel feature map, we add absolute position encoding to features which embedded in patch embedding layer, which could be formed as follows:

$$y = Patch.Emb(x_{input}) \tag{4}$$
$$y_{temp} = IPSA(y + ab.pos.) \tag{5}$$
$$y_{output} = IPSA(CPSA(y_{temp})) \tag{6}$$

where ab.pos. indicates that absolute position encoding, and Patch.Emb indicates that patch embedding layer in Table 1. Absolute position encoding is useful in CPSA to improve the performance, the results reported in Table 6.

### 3.1.3  Cross Attention based Transformer

Cross Attention block consists of two inner-patch self-attention blocks and a cross-patch self-attention block, as shown in Figure 2(b). CAT Layer is composed of several CABs, and each stage

5

Table 1: Detailed configurations of CATs. down. rate indicates that down-sample rate at each stage. R indicates that dowm-sample rate at specific layer.

| | down. rate | CAT-T | CAT-S | CAT-B |
|---|---|---|---|---|
| stage 1 | 4× | Patch Embedding R=4, Dim=64 | Patch Embedding R=4, Dim=96 | Patch Embedding R=4, Dim=96 |
| | | $\begin{bmatrix} \text{IPSA, head 2,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 2,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ | $\begin{bmatrix} \text{IPSA, head 3,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 3,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ | $\begin{bmatrix} \text{IPSA, head 3,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 3,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ |
| stage 2 | 8× | Patch Projection R=2, Dim=128 | Patch Projection R=2, Dim=192 | Patch Projection R=2, Dim=192 |
| | | $\begin{bmatrix} \text{IPSA, head 4,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 4,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ | $\begin{bmatrix} \text{IPSA, head 6,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 6,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ | $\begin{bmatrix} \text{IPSA, head 6,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 6,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ |
| stage 3 | 16× | Patch Projection R=2, Dim=256 | Patch Projection R=2, Dim=384 | Patch Projection R=2, Dim=384 |
| | | $\begin{bmatrix} \text{IPSA, head 8,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 8,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 3$ | $\begin{bmatrix} \text{IPSA, head 12,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 12,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 3$ | $\begin{bmatrix} \text{IPSA, head 12,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 12,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 6$ |
| stage 4 | 32× | Patch Projection R=2, Dim=512 | Patch Projection R=2, Dim=768 | Patch Projection R=2, Dim=768 |
| | | $\begin{bmatrix} \text{IPSA, head 16,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 16,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ | $\begin{bmatrix} \text{IPSA, head 24,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 24,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ | $\begin{bmatrix} \text{IPSA, head 24,} \\ \text{CPSA, head 1,} \\ \text{IPSA, head 24,} \\ \text{patch. sz. } 7 \times 7 \end{bmatrix} \times 1$ |

of the network is composed of a different number of layers and a patch embedding layer as shown in Figure 2(a), the pipeline of CAB is as follows:

$$\hat{y}_{temp1} = IPSA(LN(\hat{y}_{n-1})) + \hat{y}_{n-1} \tag{7}$$

$$\hat{y}_{temp2} = MLP(LN(\hat{y}_{temp1})) + \hat{y}_{temp1} \tag{8}$$

$$\hat{y}_{temp3} = CPSA(LN(\hat{y}_{temp2})) + \hat{y}_{temp2} \tag{9}$$

$$\hat{y}_{temp4} = MLP(LN(\hat{y}_{temp3})) + \hat{y})_{temp3} \tag{10}$$

$$\hat{y}_{temp5} = IPSA(LN(\hat{y}_{temp4})) + \hat{y}_{temp4} \tag{11}$$

$$\hat{y}_n = MLP(LN(\hat{y}_{temp5})) + \hat{y}_{temp5} \tag{12}$$

where $\hat{y}_{tempi}$ is an output of one block(e.g., IPSA, MLP) with LN. We compare convolution for patch embedding layer in [33], where the convolution kernel size is set to P and the stride is also P, and slicing the inputs as [16], the result is reported in Table 5, both have the same performance. Our default setting is former. According to the number of CABs in stage3 and the dimension of patch projection layer, three models of different computational complexity are designed, which are CAT-T, CAT-S, and CAT-B with 1×, 2×, and 3× of computation, respectively. Table 1 details the configuration.

## 4 Experiment

We conduct image classification, object detection, and semantic segmentation experiments on ImageNet-1K[73], COCO 2017[74], and ADE20K[75] respectively. In the following, we compare the three tasks between CAT architecture and state-of-the-arts architectures, then we report ablation experiments of some designs we adopted in CAT.

### 4.1 Image Classification

**Details** For image classification, we report the top-1 accuracy with a single crop on ImageNet-1K[73], which contains 1.28M training images and 50K validation images from 1000 categories. The setting in our experiments is mostly following [38]. We employ the batch size of 1024, the

initial learning rate of 0.001, and the weight decay of 0.05. We train the model for 300 epochs with AdamW[76] optimizer, cosine decay learning rate scheduler, and linear warm-up of 20 epochs. stochastic depth[77] is used in our training, rate of 0.1, 0.2, and 0.3 for three variants architecture respectively, and dropout[78] is adopted in self-attention of CAB with the rate of 0.2 to avoid over-fitting. We use most of the regularization strategies and augmentation in [38] that similar to [36] to make our results more comparable and convincing.

Table 2: The comparison of CAT with other networks on ImageNet. ‡ indecates that Swin-T without shifted window.

| Model | Resolution | Params(M) | FLOPs(B) | Top-1(%) |
|---|---|---|---|---|
| **CNN-based networks** | | | | |
| ResNet50[8] | $224 \times 224$ | 26 | 4.1 | 76.6 |
| ResNet101[8] | $224 \times 224$ | 45 | 7.9 | 78.2 |
| X50-32x4d[9] | $224 \times 224$ | 25 | 4.3 | 77.9 |
| x101-32x4d[9] | $224 \times 224$ | 44 | 8.0 | 78.7 |
| EffifientNet-B4[11] | $380 \times 380$ | 19 | 4.2 | 82.9 |
| EffifientNet-B5[11] | $528 \times 528$ | 30 | 9.9 | 83.6 |
| EffifientNet-B6[11] | $600 \times 600$ | 43 | 19.0 | 84.0 |
| RegNetY-4G[79] | $224 \times 224$ | 21 | 4.0 | 80.0 |
| RegNetY-8G[79] | $224 \times 224$ | 39 | 8.0 | 81.7 |
| RegNetY-16G[79] | $224 \times 224$ | 84 | 16.0 | 82.9 |
| **Transformer-based networks** | | | | |
| ViT-B/16[33] | $384 \times 384$ | 86 | 55.4 | 77.9 |
| ViT-L/16[33] | $384 \times 384$ | 307 | 190.7 | 76.5 |
| TNT-S[37] | $224 \times 224$ | 24 | 5.2 | 81.3 |
| TNT-B[37] | $224 \times 224$ | 66 | 14.1 | 82.8 |
| CrossViT-15[34] | $224 \times 224$ | 27 | 5.8 | 81.5 |
| CrossViT-18[34] | $224 \times 224$ | 44 | 9.0 | 82.5 |
| PVT-S[32] | $224 \times 224$ | 24.5 | 3.8 | 79.8 |
| PVT-M[32] | $224 \times 224$ | 44.2 | 6.7 | 81.2 |
| PVT-L[32] | $224 \times 224$ | 61.4 | 9.8 | 81.7 |
| Swin-T‡[36] | $224 \times 224$ | 29 | - | 80.2 |
| Swin-T[36] | $224 \times 224$ | 29 | 4.5 | 81.3 |
| Swin-B[36] | $224 \times 224$ | 88 | 15.4 | 83.3 |
| CAT-T(**ours**) | $224 \times 224$ | 17 | 2.8 | 80.3 |
| CAT-S(**ours**) | $224 \times 224$ | 37 | 5.9 | 81.8 |
| CAT-B(**ours**) | $224 \times 224$ | 52 | 8.9 | 82.8 |

**Results**    In Table 2, we present our experimental results, which demonstrates that our CAT-T could achieve the precision of 80.3% top-1 when FLOPs were 65% less than ResNet101[8]. Meanwhile, the top-1 of our CAT-S and CAT-B on images with the resolution of $224 \times 224$ were 81.8% and 82.8%, respectively. Such a result is comparable with the results of state-of-the-arts in the Table For instance, compared with Swin-T[36], which has a similar computation, our CAT-S has improved by 0.5%. In particular, our method has a much stronger ability to catch the relationship between patches than shifted operation in Swin[36]. Swin-T(w. shifted) improves 1.1% top-1 accuracy, and CAT-S surpasses 1.6%.

## 4.2 Object detection

**Details**    For object detection, we conduct experiments on COCO 2017[74] with metric of mAP, which consists of 118k training, 5k validation, and 20k test images from 80 categories. We experiment on the some frameworks to evaluate our architecture. The batch size of 16, the initial learning rate of 1e-4, weight decay of 0.05 are used in our experiments. AdamW[76] optimizer, 1x schedule, and NMS[80] are employed. Other settings are the same as MMDetection[81]. Note that a stochastic depth[77] rate of 0.2 to avoid overfitting. About multi-scale strategy, we trained with randomly select one scale shorter side from 480 to 800 spaced by 32 while the longer side is less than 1333 same as [54, 82].

**Results**    As demonstrated in Table 3, we used CAT-S and CAT-B as backbone in some anchor-based and anchor-free frameworks, both have better performance and comparable or fewer computa-

Table 3: The comparison of CAT with other backbones with various methods on COCO detection. † indicates that trained with multi-scale strategy. FLOPs is evalutated on $800 \times 1280$.

| Method | Backbone | $AP^{box}$ | $AP^{box}_{50}$ | $AP^{box}_{75}$ | $AP^{mask}$ | $AP^{mask}_{50}$ | $AP^{mask}_{75}$ | Params(M) | FLOPs(G) |
|---|---|---|---|---|---|---|---|---|---|
| Mask R-CNN[83] | ResNet50[8] | 38.0 | 58.6 | 41.4 | 34.4 | 55.1 | 36.7 | 44 | 260 |
| | ResNet101[8] | 40.4 | 61.1 | 44.2 | 36.4 | 57.7 | 38.8 | 63 | 336 |
| | CAS-S(**ours**)† | 41.6 | 65.1 | **45.4** | 38.6 | 62.2 | 41.0 | 57 | 295 |
| | CAS-B(**ours**)† | **41.8** | **65.4** | 45.2 | **38.7** | **62.3** | **41.4** | 71 | 356 |

| Method | Backbone | $AP^{box}$ | $AP^{box}_{50}$ | $AP^{box}_{75}$ | $AP^{box}_{S}$ | $AP^{box}_{M}$ | $AP^{box}_{L}$ | Params(M) | FLOPs(G) |
|---|---|---|---|---|---|---|---|---|---|
| FCOS[15] | ResNet50[8] | 36.6 | 56.0 | 38.8 | 21.0 | 40.6 | 47.0 | 32 | 201 |
| | ResNet101[8] | 39.1 | 58.3 | 42.1 | 22.7 | 43.3 | 50.3 | 51 | 277 |
| | CAT-S(**ours**) | 40.0 | 60.7 | 42.6 | 24.5 | 42.7 | 52.4 | 45 | 245 |
| | CAT-B(**ours**) | **41.0** | **62.0** | **43.2** | **25.7** | **43.5** | **53.8** | 59 | 303 |
| ATSS[84] | ResNet50[8] | 39.4 | 57.6 | 42.8 | 23.6 | 42.9 | 50.3 | 32 | 205 |
| | ResNet101[8] | 41.5 | 59.9 | 45.2 | 24.2 | 45.9 | 53.3 | 51 | 281 |
| | CAT-S(**ours**) | 42.0 | 61.6 | 45.3 | 26.4 | 44.6 | 54.9 | 45 | 243 |
| | CAT-B(**ours**) | **42.5** | **62.4** | **45.8** | **27.8** | **45.2** | **56.0** | 59 | 303 |
| RetinaNet[13] | ResNet50[8] | 36.3 | 55.3 | 38.6 | 19.3 | 40.0 | 48.8 | 38 | 234 |
| | ResNet101[8] | 38.5 | 57.8 | 41.2 | 21.4 | 42.6 | 51.1 | 57 | 315 |
| | CAT-S(**ours**) | 40.1 | 61.0 | 42.6 | 24.9 | 43.6 | 52.8 | 47 | 276 |
| | CAT-B(**ours**) | **41.4** | **62.9** | **43.8** | 24.9 | **44.6** | **55.2** | 62 | 337 |
| Cascade R-CNN[85] | ResNet50[8] | 40.4 | 58.9 | 44.1 | 22.8 | 43.7 | 54.0 | 69 | 245 |
| | ResNet101[8] | 42.3 | 60.8 | 46.1 | 23.8 | 46.2 | 56.4 | 88 | 311 |
| | CAT-S(**ours**) | 44.1 | 64.3 | 47.9 | 28.2 | 46.9 | 58.2 | 82 | 270 |
| | CAT-B(**ours**) | 44.8 | 64.9 | 48.8 | 27.7 | 47.4 | 59.7 | 96 | 330 |
| | CAS-S(**ours**)† | 45.2 | 65.6 | 49.2 | 30.2 | 48.6 | 58.2 | 82 | 270 |
| | CAS-B(**ours**)† | **46.3** | **66.8** | **49.9** | **30.8** | **49.5** | **59.7** | 96 | 330 |

tional cost. CAT-S improves FCOS[15] by 3.4%, RetinaNet[13] by 3.7%, and Cascade R-CNN[85] by 4.8% with multi-scale strategy. While for instance segmentation, we use the framework of MASK R-CNN[83], and the mask mAP improves 4.2% with CAT-S. All methods we experimented on have better performance than the original, demonstrating our CAT has a better ability to be feature extraction.

## 4.3 Semantic Segmentation

**Details**  For semantic segmentation, we experiment on ADE20K[75] which has 20k images for training, 2k images for validation, and 3k images for testing. The setting is as follows, the initial learning rate is 6e-5, the batch size is 16 for 160k and 80k iterations in total, the weight decay is 0.05, and the warm-up iteration is 1500. We conduct experiments at the framework of Semantic FPN[86] with the input of $512 \times 512$, and using the basic setting in MMSegmentation[87]. Note that the stochastic depth[77] rate of 0.2 is used in CAT while training.

Table 4: Semantic segmentation performance on ADE20K. † indicates that the model is pre-trained on ImageNet-22k. ‡ indicates that trained with 80k iterations. FLOPs is evaluated on $1024 \times 1024$.

| Method | Backbone | Params(M) | FLOPs(G) | mIoU |
|---|---|---|---|---|
| DANet[88] | ResNet101[8] | 69 | 1119 | 45.0 |
| OCRNet[23] | ResNet101[8] | 56 | 923 | 44.1 |
| OCRNet[23] | HRNet-w48[22] | 71 | 664 | 45.7 |
| DeeplabV3+[28] | ResNet101[8] | 63 | 1021 | 44.1 |
| DeeplabV3+[28] | ResNeSt-101[89] | 66 | 1051 | 46.9 |
| UperNet[90] | ResNet101[8] | 86 | 1029 | 44.9 |
| SETR[90] | T-Large† | 308 | - | 50.3 |
| Semantic FPN[86] | ResNet50[8] | 29 | 183 | 39.1 |
| | ResNet101[8] | 48 | 260 | 40.7 |
| | CAT-S(**ours**) | 41 | 214 | 42.8 |
| | CAT-B(**ours**) | 55 | 276 | 44.9 |
| Semantic FPN‡ | ResNet50[8] | 29 | 183 | 36.7 |
| | ResNet101[8] | 48 | 260 | 38.8 |
| | CAT-S(**ours**) | 41 | 214 | 42.1 |
| | CAT-B(**ours**) | 55 | 276 | 43.6 |

**Results**   As shown in Table 4, we employ CAT-S and CAT-B as the backbone, with the framework of Semantic FPN[90]. Semantic FPN achieves better performance with CAT-S and CAT-B, especially, we achieve 44.9% mIoU with 160k iterations and CAT-B, 4.2% improved compared ResNet101[8] as the backbone, making Semantic FPN obtains comparable performance as other methods, while for 80k iterations, result is enhanced 4.8%, which illustrates that our architecture is more powerful than ResNet[8] to be a backbone.

## 4.4 Ablation Study

In this section, we report results of the ablation experiments for some designs we made in designing the architecture and in conducting the experiments on ImageNet-1K[73], COCO 2017[74], and ADE20K[75].

**Patch Embedding function**   We compare the embedding function in patch embedding layer, convolutional method and method in [16], the former conduct convolutional layer with the kernel size of $4 \times 4$ and stride of 4 to reduce the resolution of input to $1/4$ of origin, the latter slice the input from $H \times W \times H \times C$ to $H/S \times W/S \times SC$, where S in ours is 4 to implement the same as the former. The results in Table 5 show that the two methods have same performance. To better compare with other work[36], we choose the convolutional method as the default setting.

**Multi-head and shifted window**   Multi-head is proposed in [29], which represents different semantic features among words. We set the number of heads equal to patch size in each CPSA, which is useless to the performance, presented in Table 5. To study the shifted window in Swin[36], we also experimented w./wo. the shifted window at the third block of CAB, the result shows that the shifted operation does not perform better in our architecture.

Table 5: Ablation study on multi-head in CPSA, shifted window in second IPSA block in CAB, and slice or convlutional method in patch embedding layer, using CAT-S architecture on ImageNet-1K.

| multi-head | shifted | slice | conv. | Top-1(%) |
|---|---|---|---|---|
| | ✓ | | ✓ | 81.7 |
| ✓ | | | ✓ | 81.6 |
| | | ✓ | | **81.8** |
| | | | ✓ | **81.8** |

Table 6: Ablation study on the absolute position encoding and dropout in self-attention of CPSA on three benchmarks with CAT-S architecture. FCOS[15] with 1x schedule on COCO 2017 and Semanticc FPN[90] with 80k iterations on ADE20K is used. attn.d: dropout of selt-attention. abs.pos.: absolute position encoding.

| | ImageNet | | COCO 2017 | | | ADE20k |
|---|---|---|---|---|---|---|
| | top-1 | top-5 | AP | $AP^{50}$ | $AP^{75}$ | mIoU |
| no attn.d | 81.5 | 95.2 | 39.8 | 60.5 | 43.0 | 42.0 |
| attn.d 0.2 | **81.8** | **95.6** | **40.0** | **60.7** | **43.2** | **42.1** |
| no abs.pos. | 81.6 | 95.3 | 39.6 | 60.2 | 42.9 | 41.8 |
| abs.pos. | **81.8** | **95.6** | **40.0** | **60.7** | **43.2** | **42.1** |

**Absolute position and dropout in self-attention of CPSA**   We conduct ablation study on absolute position encoding for CPSA, and it improves the performance on three benchmarks. To better training, we adopted the dropout[78] of self-attention in CPSA and set the rate of 0.0 and 0.2. The rate of 0.2 achieves the best performance, illustrating there is a little overfitting in CPSA. All results are reported in Table 6.

## 5 Conclusion

In this paper, the proposed Cross Attention is proposed to better combine the virtue of local feature extraction in CNN with the virtue of global information extraction in Transformer, and build a robust backbone, which is CAT. It can generate features at different scales similar to most CNN-based networks, and it can also adapt to different sizes of inputs for other vision tasks. CAT achieves state-of-the-arts performance on various vision task datasets (e.g., ImageNet-1K[73], COCO 2017[74], ADE20K[75]). The key is that we alternate attention inner the feature map patch and attention on the single-channel feature map without quite increasing the computation to capture local and global information. We hope that our work will be a step in the direction of integrating CNN and Transformer to create a multi-domain approach.

# References

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

[2] 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.

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

[4] 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.

[5] 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.

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

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

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

[9] 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.

[10] Shanghua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip HS Torr. Res2net: A new multi-scale backbone architecture. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

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

[12] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[13] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[14] 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.

[15] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9627–9636, 2019.

[16] Ultralytics. Yolov5. https://github.com/ultralytics/yolov5, 2021.

[17] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

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

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[20] 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.

[21] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[22] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5693–5703, 2019.

[23] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. *arXiv preprint arXiv:1909.11065*, 2019.

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

[25] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[26] 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. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

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

[28] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[30] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

[31] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[32] 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.

[33] 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.

[34] Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *arXiv preprint arXiv:2103.14899*, 2021.

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

[36] 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.

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

[38] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[39] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021.

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

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

[42] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, pages 1691–1703. PMLR, 2020.

[43] Andrew Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. *arXiv preprint arXiv:2102.06171*, 2021.

[44] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[45] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[46] 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.

[47] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[48] 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.

[49] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

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

[51] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. *arXiv preprint arXiv:2103.11816*, 2021.

[52] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[53] Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *arXiv preprint arXiv:1904.00760*, 2019.

[54] 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.

[55] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[56] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2012.15840*, 2020.

[57] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. *arXiv preprint arXiv:2012.00759*, 2020.

[58] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

[59] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[60] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[61] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019.

[62] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

[63] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908, 2020.

[64] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018.

[65] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

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

[67] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[68] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[69] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-yolov4: Scaling cross stage partial network. *arXiv preprint arXiv:2011.08036*, 2020.

[70] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. *arXiv preprint arXiv:2003.05664*, 2020.

[71] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3588–3597, 2018.

[72] Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *International Conference on Machine Learning*, pages 642–652. PMLR, 2020.

[73] 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.

[74] 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.

[75] 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.

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

[77] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

[78] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[79] 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.

[80] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 850–855. IEEE, 2006.

[81] 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.

[82] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. *arXiv preprint arXiv:2011.12450*, 2020.

[83] 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.

[84] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9759–9768, 2020.

[85] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018.

[86] 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.

[87] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation tool-box and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020.

[88] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3146–3154, 2019.

[89] 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.

[90] 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.