Focal Attention for Local-Global Interactions in Vision Transformers

Jianwei Yang[1] **Chunyuan Li**[1] **Pengchuan Zhang**[1] **Xiyang Dai**[2] **Bin Xiao**[2]
Lu Yuan[2] **Jianfeng Gao**[1]
[1]Microsoft Research at Redmond, [2]Microsoft Cloud + AI
`{jianwyan,chunyl,penzhan,xidai,bixi,luyuan,jfgao}@microsoft.com`

Abstract

Recently, Vision Transformer and its variants have shown great promise on various computer vision tasks. The ability of capturing local and global visual dependencies through self-attention is the key to its success. But it also brings challenges due to quadratic computational overhead, especially for the high-resolution vision tasks (*e.g.*, object detection). Many recent works have attempted to reduce the cost and improve model performance by applying either coarse-grained global attention or fine-grained local attention. However, both approaches cripple the modeling power of the original self-attention mechanism of multi-layer Transformers, leading to sub-optimal solutions. In this paper, we present *focal attention*, a new attention mechanism that incorporates both fine-grained local and coarse-grained global interactions. In this new mechanism, each token attends its closest surrounding tokens at fine granularity and the tokens far away at coarse granularity, and thus can capture both short- and long-range visual dependencies efficiently and effectively. With focal attention, we propose a new variant of Vision Transformer models, called *Focal Transformers*, which achieve superior performance over the state-of-the-art (SoTA) Vision Transformers on a range of public image classification and object detection benchmarks. In particular, our Focal Transformer models with a moderate size of 51.1M and a large size of 89.8M achieve **83.6**% and **84.0**% Top-1 accuracy, respectively, on ImageNet classification at 224×224. When employed as the backbones, Focal Transformers achieve consistent and substantial improvements over the current SoTA Swin Transformers [43] across 6 different object detection methods. Our largest Focal Transformer yields **58.7/59.0** box mAPs and **50.9/51.3** mask mAPs on COCO mini-val/test-dev, and **55.4** mIoU on ADE20K for semantic segmentation, creating new SoTA on three of the most challenging computer vision tasks. Our code is available at: `https://github.com/microsoft/Focal-Transformer`.

1 Introduction

Nowadays, Transformer [57] has become a prevalent model architecture in natural language processing (NLP) [20, 6]. In the light of its success in NLP, there is an increasing effort on adapting it to computer vision (CV) [47, 50]. Since its promise firstly demonstrated in Vision Transformer (ViT) [21], we have witnessed a flourish of full-Transformer models for image classification [55, 60, 64, 43, 76, 56], object detection [8, 85, 79, 18] and semantic segmentation [58, 62]. Beyond these static image tasks, it has also been applied on various temporal understanding tasks, such as action recognition [40, 78, 10], object tracking [13, 59], scene flow estimation [38].

The self-attention mechanism is arguably the key component that differentiates Transformers from the widely used convolutional neural networks (CNNs) [37] in computer vision. At each Transformer layer, self-attention enables global content-dependent interactions among different image regions for

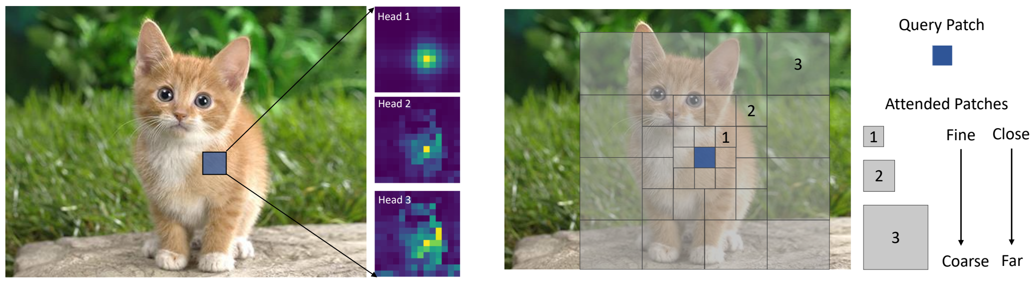


Figure 1: Left: Visualization of the attention maps of the three heads at the given query patch (blue) in the first layer of the DeiT-Tiny model [55]. Right: An illustrative depiction of focal attention mechanism. Three granularity levels are used to compose the attention region for the blue query.

modeling short- and long-range dependencies, respectively. Through the visualization of full self-attention results[1], we indeed observe that self-attention learns to attend local surroundings (like CNNs) and the global contexts at the same time, as illustrated in Fig. 1 (Left). Nevertheless, when dealing with high-resolution vision tasks such as object detection or segmentation, an efficient implementation of a global and fine-grained self-attention becomes non-trivial due to the quadratic computational cost with respect to the number of grids in feature maps. Recent works have alternatively exploited either a coarse-grained global self-attention [60, 64] or a fine-grained local self-attention [43, 76, 56], for the sake of reducing the computational cost. However, both approaches cripple the power of the original full self-attention *i.e.*, the ability to simultaneously capture local and global visual dependencies.

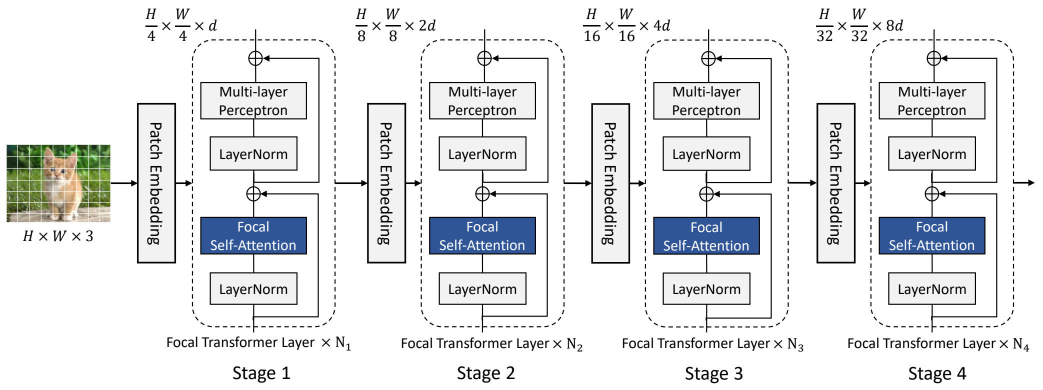
In this paper, we present a new attention mechanism to capture both local and global interactions in Transformer layers for high-resolution input images. Considering that the visual dependencies between the nearby (local) regions are usually much stronger than the dependencies between the regions that are far away, we perform the fine-grained attention only in local regions while the coarse-grained attention globally. As depicted in Fig. 1 (Right), a query token in the feature map attends its closest local surroundings at the finest granularity as itself. However, when it goes to the regions far away, it attends to *summarized* tokens to capture coarse-grained visual dependencies. We call this new mechanism *focal attention*, as each token attends the others in a focal manner. We will show in this study that focal attention allows to effectively model visual dependencies among all regions covering the whole high-resolution feature maps while introducing much less number of tokens in the computation than that in the standard self-attention mechanism.

Equipped with focal attention, a series of *Focal Transformers* are developed and validated via a comprehensive empirical study across three core vision tasks, including image classification, object detection and segmentation. Results show that Focal Transformers consistently outperform the SoTA Vision Transformers across various settings (i.e., in model sizes and complexities). Notably, the small Focal Transformer with 51.1M parameters achieves 83.6% top-1 accuracy on ImageNet-1K, and the base model with 89.8M parameters obtains 84.0% top-1 accuracy. In the fine-tuning experiments for object detection, Focal Transformers consistently outperform the SoTA Swin Transformers [43] across six popular object detection methods. Our largest Focal Transformer model achieves **59.0** box mAP and **51.3** mask mAP on COCO test-dev for object detection and instance segmentation, respectively, and **55.4** mIoU on ADE20K for semantic segmentation. These results demonstrate that focal attention is highly effective in modeling the local-global interactions in Vision Transformers.

2 Related work

Vision Transformers. Vision Transformer (ViT) is first introduced in [21]. It applies a standard Transformer, originally developed for NLP [57], to encode an image by analogously splitting the image into a sequence of visual tokens. It has demonstrated superior performance to CNNs such as ResNet [33] on multiple image classification benchmarks, when trained with sufficient data [21] and carefully designed data augmentation and regularization methods [55]. The results thus inspire researchers to explore the applications of ViT on various vision tasks beyond image classification, such as self-supervised learning [14, 9, 39], object detection [8, 85, 79, 18] and semantic segmentation [58, 62, 81]. There are also increasing number of studies for improving ViT via data-efficient

[1]DeiT-Tiny model, checkpoint downloaded from `https://github.com/facebookresearch/deit`.



$H \times W \times 3$
$\frac{H}{4} \times \frac{W}{4} \times d$
$\frac{H}{8} \times \frac{W}{8} \times 2d$
$\frac{H}{16} \times \frac{W}{16} \times 4d$
$\frac{H}{32} \times \frac{W}{32} \times 8d$
Patch Embedding
Multi-layer Perceptron
LayerNorm
Focal Self-Attention
LayerNorm
Focal Transformer Layer × N_1
Focal Transformer Layer × N_2
Focal Transformer Layer × N_3
Focal Transformer Layer × N_4
Stage 1
Stage 2
Stage 3
Stage 4

Figure 2: Model architecture for our Focal Transformers. As highlighted in light blue boxes, our main innovation is the proposed focal attention in each Transformer layer.

training [55], improved patch embedding/encoding [16, 71, 31], integrating convolutional projections into transformers [64, 70], and using multi-scale architectures and efficient self-attention mechanisms for high-resolution vision tasks [60, 64, 43, 76, 15]. Recent surveys include [36, 30, 36]. This paper focuses on improving the self-attention mechanism of ViT for encoding high-resolution images.

Efficient global and local self-attention. In many real-world tasks, Transformers need to cope with a large number of input tokens, such as long documents in NLP and high-resolution images in computer vision (CV). Recently, many efficient self-attention mechanisms have been proposed to deal with the quadratic computational and memory cost incurred by the standard self-attention mechanism. On one hand, a number of works in both NLP and CV resort to coarse-grained global self-attention (i.e., attending the down-sampled or summarized tokens) to capture the long-range interactions [49, 46, 60, 64, 31, 23]. Although this approach improves the model efficiency, it loses the detailed context information surrounding the query tokens. On the other hand, to make the computational cost manageable, various local fine-grained attention mechanism (i.e., attending neighboring tokens within a pre-set window size) are used for both NLP [3, 74, 1] and CV [56, 43, 76]. In this paper, we argue that both global and local attentions are important for model performance. This is also validated by some recent studies that aim to improve CNNs by incorporating ways of modeling global attentions [35, 63, 61, 68, 2, 7, 51]. The standard self-attention mechanism used by ViT can indeed learned both types of attentions, as shown in Fig. 1 (Left). But it often incurs a prohibitively high cost for high-resolution images. To the best of our knowledge, the proposed focal self-attention provides the first mechanism to incorporate global and local self-attention in a single Transformer layer [2]. It can capture both local and global interactions as standard self-attention but in a much more efficient and effective way, especially for high-resolution images.

3 Method

3.1 Model architecture

To accommodate high-resolution dense prediction tasks, we employ a multi-scale model architecture as in [60, 76, 43]. As shown in Fig. 2, an image $I \in \mathcal{R}^{H\times W\times 3}$ is first partitioned into patches of size 4×4, resulting in $\frac{H}{4}\times\frac{W}{4}$ visual tokens with dimension $4\times 4\times 3$. Then, we use a patch embedding layer, consisting of a convolutional layer with filter size and stride both equal to 4, to project these patches into hidden features with dimension d. We then pass this spatial feature map to the four stages of Focal Transformer blocks. In each stage $i \in \{1,2,3,4\}$, the Focal Transformer block consists of N_i Focal Transformer layers. After each stage, we use a patch embedding layer to reduce the spatial size of feature map by factor 2 and increase the feature dimension by 2. For image classification tasks, we take the average of the output from the last stage and send it to a classification layer. For object detection, the feature maps from the last 3 or all 4 stages are fed to a particular object detector head,

[2]A similar focal mechanism has been used in CNNs for NLP [27].

depending on the specific detection method we choose to use. The model capacity can be customized by varying the input feature dimension d and the number of Focal Transformer layers.

Standard self-attention can capture both short- and long-range interactions at fine-grain, but suffers from high computational cost when it performs attention on high-resolution feature maps as noted in [76]. Take stage 1 in Fig. 2 as an example. For a feature map of size $\frac{H}{4} \times \frac{W}{4} \times d$, the complexity of self-attention is $\mathcal{O}((\frac{H}{4} \times \frac{W}{4})^2 d)$, resulting in an explosion of time and memory cost, considering that $\min(H, W)$ could be 800 or even larger for object detection. In the next section, we describe how we address this issue with the proposed focal attention mechanism.

3.2 Token-wise focal attention

Focal attention is proposed to make the Transformer layers suitable for encoding high-resolution input images. Instead of attending all tokens at fine-grain, we attend the fine-grain tokens only locally, but the summarized ones (i.e., the coarse-grained tokens generated by sub-window pooling, which is illustrated in Fig. 4 and will be described later) globally. As such, focal attention can cover the same amount of image regions as standard self-attention but with much less cost. In Fig. 3, we show the size of the receptive field for standard self-attention and our focal attention as a function of the number of attended tokens. For a given query position, by reducing the granularity of its surroundings based on their distance to the query, focal attention can have significantly larger receptive fields at the same cost measured by the number of visual tokens, compared to the standard self-attention mechanism.

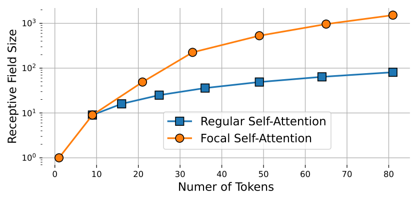


Figure 3: The size of receptive field (y-axis) as a function of the number of used visual tokens (x-axis) in regular (standard) self-attention and focal attention. When plotting the curve for focal attention, we increase the focal window size by 2 for each focal level up to the maximal window size of 8.

Theoretically, the focal attention mechanism enables global interaction with much less time and memory cost, because it attends a much smaller number of surrounding (summarized) tokens. In practice, however, extracting the surrounding tokens for each query position could incur high time cost since we need to duplicate the extraction of each token for all queries that the token surrounds. This issue had been extensively discussed in [56, 76, 43] and a common solution is to partition the input feature map into windows. Thus, in our Focal Transformers, we resort to performing focal attention at the window level. We elaborate the window-wise focal attention in the following.

3.2.1 Window-wise focal attention

Given a feature map of $x \in \mathcal{R}^{M \times N \times d}$ with spatial size $M \times N$, we first partition it into a grid of windows of size $s_p \times s_p$. Then, we extract the surroundings for each window rather than each individual token. The proposed window-wise focal attention is illustrated in Fig. 4. To clarify, we first define three terms:

- **Focal level** L refers to the granularity level at which we extract the tokens for focal attention.
- **Focal window size** s_w^l is the size of sub-window on which the summarized tokens are formed via sub-window pooling at granularity level of $l \in \{1, ..., L\}$.
- **Focal region size** s_r^l denotes the number of sub-windows that are filled up horizontally (or vertically) in an attended region at level l.

Now, we detail how window-wise focal attention works in the following two steps, sub-window pooling and attention computing.

Sub-window pooling. Consider input feature map $x \in \mathcal{R}^{M \times N \times d}$, where $M \times N$ is the spatial dimension and d the feature dimension. We perform sub-window pooling for all L levels. At focal level l, we first split the input feature map x into a grid of sub-windows with size $s_w^l \times s_w^l$. Then we use a linear projection layer f_p^l to pool the sub-windows spatially by

$$x^l = f_p^l(\hat{x}) \in \mathcal{R}^{\frac{M}{s_w^l} \times \frac{N}{s_w^l} \times d}, \quad \hat{x} = \text{Reshape}(x) \in \mathcal{R}^{(\frac{M}{s_w^l} \times \frac{N}{s_w^l} \times d) \times (s_w^l \times s_w^l)}. \tag{1}$$

The pooled feature maps $\{x^l\}_1^L$ at different levels l provide rich information at both fine-grain and coarse-grain. Since we set $s_w^l = 1$ for the first focal level which has the same granularity as the input

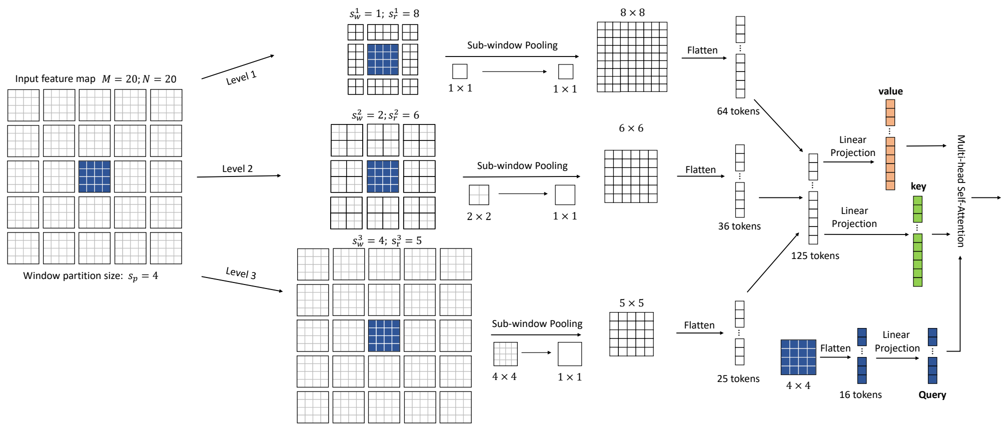


Figure 4: An illustration of focal attention at window level. Each of the square cells represents a visual token that is either from the original feature map or a summarized token formed by sub-window pooling. Suppose we have an input feature map of size 20×20. We first partition it into 5×5 windows of size 4×4. Take the 4×4 blue window in the middle as the query set, we extract its surrounding tokens at three granularity levels as its keys and values. For the first level, we extract the 8×8 tokens which are closest to the blue window at the finest grain. At the second level, we expand the attention region and pool the surrounding 2×2 sub-windows to form summarized tokens, which results in 6×6 summarized tokens. At the third level, we attend a larger region covering the whole feature map and pool 4×4 sub-windows, which leads to 5×5 summarized tokens. Finally, these three levels of tokens are concatenated to compute the keys and values for the $4 \times 4 = 16$ tokens (queries) in the blue window.

feature map, there is no need to perform any sub-window pooling. Considering that the focal window size is usually very small (7 maximally in our settings), the number of extra parameters introduced by sub-window pooling is negligible.

Attention computing. Once we obtain the pooled feature maps $\{x^l\}_1^L$ at all L levels, we compute the query at the first level, and key and value for all levels using three linear projection layers f_q, f_k and f_v, respectively, as

$$Q = f_q(x^1), \quad K = \{K^l\}_1^L = f_k(\{x^1, ..., x^L\}), \quad V = \{V^l\}_1^L = f_v(\{x^1, ..., x^L\}). \tag{2}$$

To perform focal attention, we need to first extract the surrounding tokens for each query token in the feature map. As mentioned earlier, tokens inside a window partition $s_p \times s_p$ share the same set of surroundings. For the queries inside the i-th window $Q_i \in \mathcal{R}^{s_p \times s_p \times d}$, we extract the $s_r^l \times s_r^l$ keys and values from K^l and V^l surrounding the window which the query lies in, and then gather the keys and values from all L levels to obtain $K_i = \{K_i^1, ..., K_i^L\} \in \mathcal{R}^{s \times d}$ and $V_i = \{V_i^1, ..., V_i^L\} \in \mathcal{R}^{s \times d}$, where s is the sum of focal regions from all levels, i.e., $s = \sum_{l=1}^L (s_r^l)^2$. Note that a *canonical* implementation of focal attention following Fig. 1 requires to exclude the overlapped regions across different levels. In our implementation, we intentionally keep them in order to capture the pyramid information for the overlapped regions. Finally, we follow [43] to include a relative position bias and compute the focal attention for Q_i by

$$\text{Attention}(Q_i, K_i, V_i) = \text{Softmax}(\frac{Q_i K_i^T}{\sqrt{d}} + B)V_i, \tag{3}$$

where $B = \{B^l\}_1^L$ is the learnable relative position bias. It consists of L subsets for L focal levels. Similar to [43], for the first level, we parameterize it as $B^1 \in \mathcal{R}^{(2s_p-1)\times(2s_p-1)}$, considering that the horizontal and vertical position ranges are both in $[-s_p + 1, s_p - 1]$. For the other focal levels, considering that they have different granularity with respect to the queries, we treat all the queries inside a window equally and use $B^l \in \mathcal{R}^{s_r^l \times s_r^l}$ to represent the relative position bias between the query window and each of $s_r^l \times s_r^l$ summarized tokens. Since the focal attention for each window can be performed independent of the others, we can compute Eq. (3) in parallel. Once we obtain attention scores for the whole input feature map, we send them to LayerNorm and the MLP block.

	Output Size	Layer Name	Focal-Tiny	Focal-Small	Focal-Base
stage 1	56×56	Patch Embedding	$p_1 = 4; c_1 = 96$	$p_1 = 4; c_1 = 96$	$p_1 = 4; c_1 = 128$
	56×56	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 7\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 7\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 7\} \end{bmatrix} \times 2$
stage 2	28×28	Patch Embedding	$p_2 = 2; c_2 = 192$	$p_2 = 2; c_2 = 192$	$p_2 = 2; c_2 = 256$
	28×28	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 5\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 5\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 5\} \end{bmatrix} \times 2$
stage 3	14×14	Patch Embedding	$p_3 = 2; c_3 = 384$	$p_3 = 2; c_3 = 384$	$p_3 = 2; c_3 = 512$
	14×14	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 3\} \end{bmatrix} \times 6$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 3\} \end{bmatrix} \times 18$	$\begin{bmatrix} s^0_{w,r} = \{1, 13\} \\ s^1_{w,r} = \{7, 3\} \end{bmatrix} \times 18$
stage 4	7×7	Patch Embedding	$p_4 = 2; c_4 = 768$	$p_4 = 2; c_4 = 768$	$p_4 = 2; c_4 = 1024$
	7×7	Transformer Block	$\begin{bmatrix} s^0_{w,r} = \{1, 7\} \\ s^1_{w,r} = \{7, 1\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 7\} \\ s^1_{w,r} = \{7, 1\} \end{bmatrix} \times 2$	$\begin{bmatrix} s^0_{w,r} = \{1, 7\} \\ s^1_{w,r} = \{7, 1\} \end{bmatrix} \times 2$

Table 1: Model configurations for Focal Transformers. We use three configurations with different model capacities: Focal-Tiny, Focal-Small and Focal-Base.

3.2.2 Complexity analysis

We analyze the computational complexity for the two steps of focal attention described above. For the input feature map $x \in \mathcal{R}^{M \times N \times d}$, we have $\frac{M}{s^l_w} \times \frac{N}{s^l_w}$ sub-windows at focal level l. For each sub-window, the pooling operation in Eq.1 has the complexity of $\mathcal{O}((s^l_w)^2 d)$. Aggregating all sub-windows brings us $\mathcal{O}((MN)d)$. Then for all focal levels, we have the complexity of $\mathcal{O}(L(MN)d)$ in total, which is independent of the sub-window size at each focal level. Regarding the attention computation in Eq. 3, the computational cost for a query window $s_p \times s_p$ is $\mathcal{O}((s_p)^2 \sum_l (s^l_r)^2 d)$, and $\mathcal{O}(\sum_l (s^l_r)^2 (MN)d)$ for the whole input feature map. To sum up, the overall computational cost for focal attention is $\mathcal{O}((L + \sum_l (s^l_r)^2)(MN)d)$. In an extreme case, one can set $s^L_r = 2 \times \max(M, N)/s^L_w$ to ensure a global receptive field for all queries (including both corner and middle queries) in this layer.

3.3 Model configurations

For fair comparison, we consider three network configurations for Focal Transformers, following [60, 64, 43]. Specifically, we follow the design of the Tiny, Small and Base models in Swin Transformer [43], as shown in Table 1. Our models take 224×224 images as inputs and the window partition size is set to 7 to make our models comparable to Swin Transformers. For the focal attention layer, we introduce two levels, one for fine-grained local attention and the other for coarse-grained global attention. Except for the last stage, the focal region size is set to 13 for the window partition size of 7, which means that we expand 3 tokens for each window partition. For the last stage, since the whole feature map is 7×7, the focal region size at level 0 is set to 7, which is sufficient to cover the entire feature map. For the coarse-grained global attention, we set its focal window size the same as the window partition size 7, but gradually decrease the focal region size to get $\{7, 5, 3, 1\}$ for the four stages, respectively. For the patch embedding layer, the spatial reduction ratio p_i for the four stages are all $\{4, 2, 2, 2\}$. Note that Focal-Base has a higher hidden dimension c_i, compared to Focal-Tiny and Focal-Small.

4 Experiments

4.1 Image classification on ImageNet-1K

We compare different methods on ImageNet-1K [19]. For fair comparison, we follow the training recipes in [55, 60]. All models are trained for 300 epochs with batch size 1024. The initial learning rate is set to 10^{-3} with 20 epochs of linear warm-up starting from 10^{-5}. For optimization, we use AdamW [44] as the optimizer with a cosine learning rate scheduler. The weight decay is set to 0.05 and the maximal gradient norm is clipped to 5.0. We use the same set of data augmentation and regularization strategies used in [55] after excluding random erasing [82], repeated augmentation [4, 34] and exponential moving average (EMA) [48]. The stochastic depth drop rates are set to 0.2, 0.2 and 0.3 for our tiny, small and base models, respectively. During training, we crop images randomly to 224×224, while a center crop is used during evaluation on the validation set.

Model	#Params.	FLOPs	Top-1 (%)
ResNet-50 [33]	25.0	4.1	76.2
DeiT-Small/16 [55]	22.1	4.6	79.9
PVT-Small [60]	24.5	3.8	79.8
ViL-Small [76]	24.6	5.1	82.0
CvT-13 [64]	20.0	4.5	81.6
Swin-Tiny [43]	28.3	4.5	81.2
Focal-Tiny (Ours)	28.9	4.9	**82.2**
ResNet-101 [33]	45.0	7.9	77.4
PVT-Medium [60]	44.2	6.7	81.2
CvT-21 [64]	32.0	7.1	82.5
ViL-Medium [76]	39.7	9.1	83.3
Swin-Small [43]	49.6	8.7	83.1
Focal-Small (Ours)	51.1	9.4	**83.6**
ResNet-152 [33]	60.0	11.0	78.3
ViT-Base/16 [21]	86.6	17.6	77.9
DeiT-Base/16 [55]	86.6	17.5	81.8
PVT-Large [60]	61.4	9.8	81.7
ViL-Base [76]	55.7	13.4	83.2
Swin-Base [43]	87.8	15.4	83.4
Focal-Base (Ours)	89.8	16.4	**84.0**

Table 2: Comparison of image classification on ImageNet-1K for different models. Except for ViT-Base/16, all other models are trained and evaluated on 224×224 resolution.

Backbone	RetinaNet	Mask R-CNN	
	AP^b	AP^b	AP^m
ResNet-50 [33]	36.3	38.0	34.4
PVT-Small	40.4	40.4	37.8
ViL-Small [76]	41.6	41.8	38.5
Swin-Tiny [43]	42.0	43.7	39.8
Focal-Tiny (Ours)	**43.7** (+1.7)	**44.8** (+1.1)	**41.0** (+1.3)
ResNet-101 [33]	38.5	40.4	36.4
ResNeXt101-32x4d [67]	39.9	41.9	37.5
PVT-Medium [60]	41.9	42.0	39.0
ViL-Medium [76]	42.9	43.4	39.7
Swin-Small [43]	45.0	46.5	42.1
Focal-Small (Ours)	**45.6** (+0.6)	**47.4** (+0.9)	**42.8** (+0.7)
ResNeXt101-64x4d [67]	41.0	42.8	38.4
PVT-Large [60]	42.6	42.9	39.5
ViL-Base [76]	44.3	45.1	41.0
Swin-Base [43]	45.0	46.9	42.3
Focal-Base (Ours)	**46.3** (+1.3)	**47.8** (+0.9)	**43.2** (+0.9)

Table 3: Comparisons with CNN and Transformer baselines and SoTA methods on COCO object detection. The box mAP (AP^b) and mask mAP (AP^m) are reported for RetinaNet and Mask R-CNN trained with $1\times$ schedule. More detailed comparisons with $3\times$ schedule are in Table 4.

In Table 2, we summarize the results for baseline models and the state-of-the-art models on image classification task. We can see that Focal Transformers consistently outperform other methods with similar model sizes (#Params.) and computational complexities (GFLOPs). Specifically, Focal-Tiny improves over the Transformer baseline DeiT-Small/16 by 2.3%. Meanwhile, using the same model configuration (2-2-6-2) and a few extra parameters and computations, Focal-Tiny improves over Swin-Tiny by 1.0 point. For small and base models, Focal-Small with 51.1M parameters can reach 83.6% which is better than all the counterpart small and base models using much less parameters. By increasing the model size, Focal-Base model achieves 84.0%, surpassing all the other models with comparable parameters and FLOPs.

To compare with the large-scale models, we further build Focal-Large Transformer by increasing the hidden dimension in Focal-Base from 128 to 196 while keeping all the other hyperparameters the same. We follow the common practice to pretrain our Focal-Large Transformer on ImageNet-22K and transfer it to detection and segmentation tasks [64, 43].

4.2 Object detection and instance segmentation

We benchmark our models on object detection with COCO 2017 [42]. The pretrained models are used as visual backbones and then plugged into two representative pipelines, RetinaNet [41] and Mask R-CNN [32]. All models are trained on the 118k training images and the results are reported on 5K validation set. We use the two standard training schedules, $1\times$ with 12 epochs and $3\times$ with 36 epochs. For the $1\times$ schedule, we resize image's shorter side to 800 while keeping its longer side no more than 1,333. For the $3\times$ schedule, we use the multi-scale training strategy by randomly resizing its shorter side to the range of $[480, 800]$. Considering this higher input resolution, we adaptively increase the focal sizes at four stages to $(15, 13, 9, 7)$, to ensures that the focal attention covers more than half of the image region at the first two stages, and the whole image at the last two stages. With the focal size increased, the relative position biases are accordingly up-sampled to the corresponding sizes using bilinear interpolation. During training, we use AdamW [44] for optimization with initial learning rate 10^{-4} and weight decay 0.05. Similarly, we use 0.2, 0.3 and 0.5 stochastic depth drop rates to regularize the training for our Tiny, Small and Base models, respectively. Since Swin Transformer does not report the results on RetinaNet, we obtain the results by ourselves using their official code with the same hyper-parameters as that of Focal Transformers.

In Table 3, we show the performance for both CNN-based models and the current Transformer-based state-of-the-art models. The bbox mAP (AP^b) and mask mAP (AP^m) are reported. We see that Focal Transformers outperform the CNN-based models consistently with the gap of 4.8-7.1 points. Compared with the other methods which also use multi-scale Transformer architectures,

Backbone	#Params	FLOPs	RetinaNet 3x schedule + MS						Mask R-CNN 3x schedule + MS					
	(M)	(G)	AP^b	AP^b_{50}	AP^b_{75}	AP_S	AP_M	AP_L	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet50 [33]	37.7/44.2	239/260	39.0	58.4	41.8	22.4	42.8	51.6	41.0	61.7	44.9	37.1	58.4	40.1
PVT-Small[60]	34.2/44.1	226/245	42.2	62.7	45.0	26.2	45.2	57.2	43.0	65.3	46.9	39.9	62.5	42.8
ViL-Small [76]	35.7/45.0	252/174	42.9	63.8	45.6	27.8	46.4	56.3	43.4	64.9	47.0	39.6	62.1	42.4
Swin-Tiny [43]	38.5/47.8	245/264	45.0	65.9	48.4	29.7	48.9	58.1	46.0	68.1	50.3	41.6	65.1	44.9
Focal-Tiny (Ours)	39.4/48.8	265/291	**45.5**	**66.3**	**48.8**	**31.2**	**49.2**	**58.7**	**47.2**	**69.4**	**51.9**	**42.7**	**66.5**	**45.9**
ResNet101 [33]	56.7/63.2	315/336	40.9	60.1	44.0	23.7	45.0	53.8	42.8	63.2	47.1	38.5	60.1	41.3
ResNeXt101-32x4d [67]	56.4/62.8	319/340	41.4	61.0	44.3	23.9	45.5	53.7	44.0	64.4	48.0	39.2	61.4	41.9
PVT-Medium [60]	53.9/63.9	283/302	43.2	63.8	46.1	27.3	46.3	58.9	44.2	66.0	48.2	40.5	63.1	43.5
ViL-Medium [76]	50.8/60.1	339/261	43.7	64.6	46.4	27.9	47.1	56.9	44.6	66.3	48.5	40.7	63.8	43.7
Swin-Small [43]	59.8/69.1	335/354	46.4	67.0	50.1	31.0	50.1	60.3	48.5	70.2	53.5	43.3	67.3	46.6
Focal-Small (Ours)	61.7/71.2	367/401	**47.3**	**67.8**	**51.0**	**31.6**	**50.9**	**61.1**	**48.8**	**70.5**	**53.6**	**43.8**	**67.7**	**47.2**
ResNeXt101-64x4d [67]	95.5/102	473/493	41.8	61.5	44.4	25.2	45.4	54.6	44.4	64.9	48.8	39.7	61.9	42.6
PVT-Large[60]	71.1/81.0	345/364	43.4	63.6	46.1	26.1	46.0	59.5	44.5	66.0	48.3	40.7	63.4	43.7
ViL-Base [76]	66.7/76.1	443/365	44.7	65.5	47.6	29.9	48.0	58.1	45.7	67.2	49.9	41.3	64.4	44.5
Swin-Base [43]	98.4/107	477/496	45.8	66.4	49.1	29.9	49.4	60.3	48.5	69.8	53.2	43.4	66.8	46.9
Focal-Base (Ours)	100.8/110.0	514/533	**46.9**	**67.8**	**50.3**	**31.9**	**50.3**	**61.5**	**49.0**	**70.1**	**53.6**	**43.7**	**67.6**	**47.0**

Table 4: COCO object detection and segmentation results with RetinaNet [41] and Mask R-CNN [33]. All models are trained with $3\times$ schedule and multi-scale inputs (MS). The numbers before and after "/" at column 2 and 3 are the model size and complexity for RetinaNet and Mask R-CNN, respectively.

Focal Transformers show substantial gains across all settings and metrics. Particularly, Focal Transformers brings 0.7-1.7 points of mAP against the current best approach Swin Transformer [43] at comparable settings. Different from the other multi-scale Transformer models, Focal Transformers can simultaneously enable short-range fine-grain and long-range coarse-grain interactions for each visual token, and thus capture richer visual contexts at each layer for better dense predictions. To have more comprehensive comparisons, we train all models using the $3\times$ schedule and show the detailed numbers for RetinaNet and Mask R-CNN in Table 4. As we can see, even with the $3\times$ schedule, Focal Transformers can still achieve 0.3-1.1 gain over Swin Transformer models in comparable settings.

Comparison with large SoTA detection models. We follow Swin Transformers to use HTC [11] as the detection method in that it reported SoTA performance on COCO detection when using Swin Transformer as the backbone. For fair comparison, we also use soft-NMS [5], instaboost [24] and a multi-scale training strategy with the shorter side in range $[400, 1400]$ and the longer side no more than 1600. We train the model using AdamW [44] with base learning rate 1e-4 and weight decay 0.1. The model is trained using the standard $3\times$ schedule. The box and mask mAPs on COCO validation set and test-dev are reported in Table 5, where both single-scale evaluation and multi-scale evaluation results are presented. Our Focal-Large model with multi-scale test achieves 58.1 box mAP and 50.9 mask mAP on mini-val set, which is better than the reported numbers for Swin-Large in [43]. When evaluating our model on the test-dev set, it achieves 58.4 box mAP and 51.3 mask mAP, which is slightly better than Swin Transformer. Note that because our model does not include the global self-attention layer used in Swin Transformer at the last stage, it has a smaller model size and fewer FLOPs. More recently, DyHead [17] achieves new SoTA on COCO, when combined with Swin-Large. We replace the Swin-Large model with the Focal-Large model, and use the same $2\times$ training schedule as in [17]. We report the box mAPs for both mini-val and test-dev. Focal-Large achieves **58.7** and **59.0** on mini-val and test-dev, respectively.

4.3 Semantic Segmentation

In addition to the instance segmentation results, we also evaluate our models on the semantic segmentation task which usually takes high-resolution input images and requires capturing long-range interactions. We benchmark our methods on ADE20K [83]. We use UperNet [65] as the segmentation method and Focal Transformers as the backbones. We train three models as Focal-Tiny, Focal-Small, Focal-Base, respectively. For all the models, we use a standard recipe that sets the input size to 512×512 and trains the model for 160k iterations with batch size 16. Table 6 shows the comparison results. We see that Focal-Tiny, Focal-Small and Focal-Base models consistently outperform Swin Transformers of the similar size in single-scale and multi-scale mIoUs.

Comparison with large SoTA semantic segmentation models. We use the pretrained Focal-Large model as the backbone for semantic segmentation. Follow the setting in [43], we use input image size 640×640 and train the model for 160k iterations with a batch size of 16. We set the initial learning to 6e-5 and use a polynomial learning rate decay. The weight decay is set to 0.01. For

Method	#Param	FLOPs	mini-val		test-dev	
			AP^b	AP^m	AP^b	AP^m
X101-64x4d [67]	155M	1033G	52.3	46.0	-	-
EfficientNet-D7 [54]	77M	410G	54.4	-	55.1	-
GCNet* [7]	-	1041G	51.8	44.7	52.3	45.4
ResNeSt-200 [75]	-	-	52.5	-	53.3	47.1
Copy-paste [28]	185M	1440G	55.9	47.2	56.0	47.4
BoTNet-200 [51]	-	-	49.7	-	-	-
SpineNet-190 [22]	164M	1885G	52.6	-	52.8	-
CenterNet2 [84]	-	-	-	-	56.4	-
Swin-L (HTC++) [43]	284M	1470G	57.1	49.5	57.7	50.2
Swin-L (DyHead) [17]	213M	965G	56.2	-	-	-
Swin-L† (HTC++) [43]	284M	-	58.0	50.4	58.7	51.1
Swin-L† (DyHead) [17]	213M	-	58.4	-	58.7	-
Swin-L† (QueryInst) [25]	-	-	56.1	-	56.1	-
Focal-L (HTC++) (Ours)	265M	1165G	57.0	49.9	-	-
Focal-L (DyHead) (Ours)	229M	1081G	56.4	-	-	-
Focal-L† (HTC++) (Ours)	265M	-	58.1	**50.9**	58.4	**51.3**
Focal-L† (DyHead) (Ours)	229M	-	**58.7**	-	**59.0**	-

Table 5: Comparison with state-of-the-art methods on COCO object detection and instance segmentation. The numbers are reported on 5K val set and test-dev. Augmented HTC [11] (denoted by HTC++) and DyHead [17] are used as the detection methods. † means multi-scale evaluation.

Backbone	Method	#Param	FLOPs	mIoU	+MS
ResNet-101	DANet [45]	69M	1119G	45.3	-
ResNet-101	ACNet [26]	-	-	45.9	-
ResNet-101	DNL [69]	69M	1249G	46.0	-
ResNet-101	UperNet [65]	86M	1029G	44.9	-
HRNet-w48 [53]	OCRNet [73]	71M	664G	45.7	-
ResNeSt-200 [75]	DLab.v3+ [12]	88M	1381G	48.4	-
Swin-T [43]	UperNet [65]	60M	945G	44.5	45.8
Swin-S [43]	UperNet [65]	81M	1038G	47.6	49.5
Swin-B [43]	UperNet [65]	121M	1188G	48.1	49.7
Twins-SVT-L [15]	UperNet [65]	133M	-	48.8	50.2
MiT-B5 [66]	SegFormer [66]	85M	-	51.0	51.8
ViT-L/16† [21]	SETR [80]	308M	-	50.3	-
Swin-L‡ [43]	UperNet [65]	234M	3230G	52.1	53.5
ViT-L/16‡ [21]	Segmenter [52]	334M	-	51.8	53.6
Swin-L‡ [43]	K-Net [77]	-	-	-	54.3
Swin-L‡ [43]	PatchDiverse [29]	234M	-	53.1	54.4
VOLO-D5 [72]	UperNet [65]	-	-	-	54.3
Focal-T (Ours)	UperNet [65]	62M	998G	45.8	47.0
Focal-S (Ours)	UperNet [65]	85M	1130G	48.0	50.0
Focal-B (Ours)	UperNet [65]	126M	1354G	49.0	50.5
Focal-L‡ (Ours)	UperNet [65]	240M	3376G	**54.0**	**55.4**

Table 6: Comparison with SoTA methods for semantic segmentation on ADE20K [83] val set. Single- and multi-scale evaluations are reported in the last two columns. ‡ means ImageNet-22K is used as the pretraining dataset.

Model	W-Size	FLOPs	Top-1 (%)	AP^b	AP^m
Swin-Tiny	7	4.5	81.2	43.7	39.8
	14	4.9	82.1	44.0	40.5
Focal-Tiny	7	4.9	82.2	44.9	41.1
	14	5.2	82.3	45.5	41.5

Table 7: Impact of different window sizes (W-Size). We alter the default size 7 to 14 and observe consistent improvements for both methods.

Model	W-Shift	Top-1 (%)	AP^b	AP^m
Swin-Tiny	-	80.2	38.8	36.4
	✓	81.2	43.7	39.8
Focal-Tiny	-	82.2	44.8	41.0
	✓	81.9	44.9	41.1

Table 8: Impact of window shift (W-Shift) on Swin Transformer and Focal Transformer. Tiny models are used.

multi-scale evaluation, we use the same scaling ratios $[0.5, 0.75, 1.0, 1.25, 1.5, 1.75]$ as in previous works. The results in Table 6 show that Focal-Large achieves significantly better performance than Swin-Large. In both single-scale and multi-scale evaluations, Focal-Large leads to more than 1 point mIoU improvement, creating new SoTA for semantic segmentation on ADE20K.

4.4 Ablation studies

We conduct a series of ablation studies to inspect the model's capacity from different aspects. We use Focal-Tiny and the image classification and object detection tasks.

Effect of varying the window size. We have demonstrated that it is crucial to model both short- and long-range interactions. Thus, a related question is whether increasing the window size helps as it leads to a larger receptive field. Table 7 shows the performance of Swin-Tiny and Focal-Tiny with window sizes 7 and 14. Clearly, a larger window size is beneficial for both methods measured in all three metrics, and Focal-Tiny consistently outperforms Swin-Tiny in both window sizes. Comparing the second and third row, we find that Focal-Tiny outperforms Swin-Tiny even with a smaller window size (7 *v.s.* 14). We suspect that the gain is attributed to Focal-Tiny's superior capability of capturing long-range dependencies among visual tokens.

The necessity of window shift. In [43], window shift is proposed to capture cross-window interactions between two successive layers. In contrast, visual tokens in Focal Transformers can always communicate with each other across windows at both fine- and coarse-grain. Thus, it is interesting to investigate whether adding window shift to Focal Transformers can lead to any improvement. To answer the question, we remove window shift from Swin Transformer while adding it to Focal Transformers. As shown in Table 8, Swin Transformer shows a severe degradation after removing the window shift. However, adding window shift to Focal Transformer hurts classification performance. The result indicates that window shift is unnecessary for Focal Transformers. While in Swin Trans-

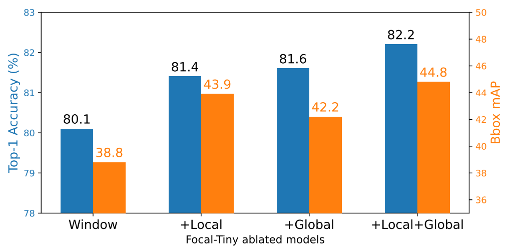


Figure 5: Ablating Focal-Tiny model by adding local, global and both interactions, respectively. Blue bars are image classification results and orange bars object detection results. This figure is better viewed in color.

Depths	Model	#Params.	FLOPs	Top-1 (%)	AP^b	AP^m
2-2-2-2	Swin	21.2	3.1	78.7	38.2	35.7
	Focal	21.7	3.4	79.9	40.5	37.6
2-2-4-2	Swin	24.7	3.8	80.2	41.2	38.1
	Focal	25.4	4.1	81.4	43.3	39.8
2-2-6-2	Swin	28.3	4.5	81.2	43.7	39.8
	Focal	29.1	4.9	82.2	44.8	41.0

Table 9: Impact of the change of model depth. We gradually reduce the number of transformer layers at the third stage from original 6 to 4 and further 2. Our Focal Transformers has much slower drop rate than Swin Transformer.

formers, there should always be an even number of layers in each stage for the alternative window shift operation, Focal Transformers do not have such a constraint.

Contributions of local and global interactions. To investigate the relative contributions of capturing local fine-grain and global coarse-grain interactions in Focal Transformers, we have developed several variants of Focal-Tiny: a) Focal-Tiny-Window merely performs attention inside each window; b) Focal-Tiny-Local attends the additional fine-grain surrounding tokens and c) Focal-Tiny-Global attends the extra coarse-grain summarized tokens. We train these models using the same setting as Focal-Tiny and report their performance on image classification and object detection using Mask R-CNN $1\times$ schedule. As shown in Fig. 5, Focal-Tiny-Window suffers from a significant performance drop on both image classification (82.2$\rightarrow$80.1) and object detection (44.8$\rightarrow$38.3). This is expected since the communication across windows is completely cut off at each Transformer layer. After we enable either the local fine-grain or global coarse-grain interactions (middle two columns), we observe significant performance boost. When we combine local and global interactions, we observe additional improvements on both tasks. This implies that these two type of interactions are complementary and both are beneficial to model performance.

Model capacity against model depth. Focal attention allows a Transformer model to capture local and global interactions at each Transformer layer. An interesting question is whether Focal Transformers need fewer layers to obtain a similar modeling capacity as the Transformer models that does not use focal attention, such as Swin Transformer. To answer this question, we conduct an experiment by training a series of Swin-Tiny and Focal-Tiny models by varying the number of Transformer layers at stage 3. As shown in Table 9, Focal-Tiny outperforms Swin-Tiny consistently with the same depth. More importantly, using fewer layers, Focal-Tiny can sometimes achieve comparable or even better performance than Swin Transformer. For example, Focal-Tiny with (2-2-4-2) achieves 81.4 on image classification which is better than Swin-Tiny with (2-2-6-2).

5 Conclusion

In this paper, we have presented a new focal attention mechanism that enables efficient local-global interactions in Vision Transformers. Different from previous works, it performs the local attention at fine-grain and global attention at coarse-grain, providing an effective way of capturing both short- and long-range context with a manageable computational cost. By applying focal attention into a multi-scale Transformer architecture, we propose Focal Transformers as general-purpose backbones for a wide range of dense vision tasks. A comprehensive empirical study shows that our Focal Transformers outperform the SoTA Vision Transformers on a range of vision tasks including image classification, object detection and segmentation.

Limitations and future work. Although our experiments show that focal attention can significantly boost the performance on image classification and dense prediction tasks, focal attention does introduce extra computational and memory cost, since each query token needs to attend global (summarized) tokens in addition to local tokens. A cost-effective implementation of Focal Transformer is necessary to make it more applicable to many real-world scenarios. This study focuses on incorporating focal attention into multi-scale Vision Transformers for CV tasks. However, we notice that focal attention is an effective sparse attention mechanism that is widely applicable to all attention-based neural network models that are developed for processing natural language, images, videos etc. This is an exciting future direction.

References

[1] Joshua Ainslie, Santiago Ontanon, Chris Alberti, Philip Pham, Anirudh Ravula, and Sumit Sanghai. Etc: Encoding long and structured data in transformers. *arXiv preprint arXiv:2004.08483*, 2020.

[2] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3286–3295, 2019.

[3] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[4] Maxim Berman, Hervé Jégou, Andrea Vedaldi, Iasonas Kokkinos, and Matthijs Douze. Multigrain: a unified image embedding for classes and instances. *arXiv preprint arXiv:1902.05509*, 2019.

[5] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017.

[6] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[7] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[8] 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.

[9] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[10] Shuning Chang, Pichao Wang, Fan Wang, Hao Li, and Jiashi Feng. Augmented transformer with adaptive graph for temporal action proposal generation. *arXiv preprint arXiv:2103.16024*, 2021.

[11] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4974–4983, 2019.

[12] 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.

[13] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. *arXiv preprint arXiv:2103.15436*, 2021.

[14] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised visual transformers. *arXiv preprint arXiv:2104.02057*, 2021.

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

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

[17] Xiyang Dai, Yinpeng Chen, Bin Xiao, Dongdong Chen, Mengchen Liu, Lu Yuan, and Lei Zhang. Dynamic head: Unifying object detection heads with attentions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7373–7382, 2021.

[18] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. *arXiv preprint arXiv:2011.09094*, 2020.

[19] 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.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 2019.

[21] 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.

[22] Xianzhi Du, Tsung-Yi Lin, Pengchong Jin, Golnaz Ghiasi, Mingxing Tan, Yin Cui, Quoc V Le, and Xiaodan Song. Spinenet: Learning scale-permuted backbone for recognition and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11592–11601, 2020.

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

[24] Hao-Shu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 682–691, 2019.

[25] Yuxin Fang, Shusheng Yang, Xinggang Wang, Yu Li, Chen Fang, Ying Shan, Bin Feng, and Wenyu Liu. Instances as queries, 2021.

[26] Jun Fu, Jing Liu, Yuhang Wang, Yong Li, Yongjun Bao, Jinhui Tang, and Hanqing Lu. Adaptive context network for scene parsing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6748–6757, 2019.

[27] Jianfeng Gao, Patrick Pantel, Michael Gamon, Xiaodong He, and Li Deng. Modeling interestingness with deep neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2–13, Doha, Qatar, October 2014. Association for Computational Linguistics.

[28] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. *arXiv preprint arXiv:2012.07177*, 2020.

[29] Chengyue Gong, Dilin Wang, Meng Li, Vikas Chandra, and Qiang Liu. Vision transformers with patch diversification, 2021.

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

[31] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer, 2021.

[32] 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.

[33] 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.

[34] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8129–8138, 2020.

[35] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

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

[37] 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.

[38] Bing Li, Cheng Zheng, Silvio Giancola, and Bernard Ghanem. Sctn: Sparse convolution-transformer network for scene flow estimation. *arXiv preprint arXiv:2105.04447*, 2021.

[39] Chunyuan Li, Jianwei Yang, Pengchuan Zhang, Mei Gao, Bin Xiao, Xiyang Dai, Lu Yuan, and Jianfeng Gao. Efficient self-supervised vision transformers for representation learning. *arXiv preprint arXiv:2106.09785*, 2021.

[40] Xiangyu Li, Yonghong Hou, Pichao Wang, Zhimin Gao, Mingliang Xu, and Wanqing Li. Trear: Transformer-based rgb-d egocentric action recognition. *arXiv preprint arXiv:2101.03904*, 2021.

[41] 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.

[42] 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 *ECCV*, 2014.

[43] 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.

[44] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[45] Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. Dual attention networks for multimodal reasoning and matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 299–307, 2017.

[46] R. Pappagari, P. Zelasko, J. Villalba, Y. Carmiel, and N. Dehak. Hierarchical transformers for long document classification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 838–844, 2019.

[47] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR, 2018.

[48] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

[49] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

[50] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*, 2019.

[51] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition, 2021.

[52] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. *arXiv preprint arXiv:2105.05633*, 2021.

[53] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019.

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

[55] 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.

[56] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12894–12904, 2021.

[57] 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.

[58] 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.

[59] Ning Wang, Wengang Zhou, Jie Wang, and Houqaing Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. *arXiv preprint arXiv:2103.11681*, 2021.

[60] 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.

[61] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[62] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv preprint arXiv:2011.14503*, 2020.

[63] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[64] 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.

[65] 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.

[66] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021.

[67] 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.

[68] Jianwei Yang, Zhile Ren, Chuang Gan, Hongyuan Zhu, and Devi Parikh. Cross-channel communication networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1297–1306, 2019.

[69] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *European Conference on Computer Vision*, pages 191–207. Springer, 2020.

[70] 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.

[71] 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.

[72] Li Yuan, Qibin Hou, Zihang Jiang, Jiashi Feng, and Shuicheng Yan. Volo: Vision outlooker for visual recognition. *arXiv preprint arXiv:2106.13112*, 2021.

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

[74] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*, 2020.

[75] 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.

[76] Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. *arXiv preprint arXiv:2103.15358*, 2021.

[77] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-net: Towards unified image segmentation, 2021.

[78] Jiaojiao Zhao, Xinyu Li, Chunhui Liu, Shuai Bing, Hao Chen, Cees GM Snoek, and Joseph Tighe. Tuber: Tube-transformer for action detection. *arXiv preprint arXiv:2104.00969*, 2021.

[79] Minghang Zheng, Peng Gao, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *arXiv preprint arXiv:2011.09315*, 2020.

[80] 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.

[81] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6881–6890, 2021.

[82] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13001–13008, 2020.

[83] 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.

[84] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Probabilistic two-stage detection. *arXiv preprint arXiv:2103.07461*, 2021.

[85] 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.