Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions

Mariha Siddika Ahmad
University of Arkansas, Fayetteville
ma135@uark.edu

Marie Louise Uwibambe
University of Arkansas, Fayetteville
uwibambe@uark.edu

December 12, 2022

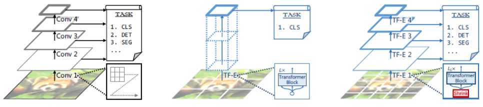
Abstract

Despite the fact that convolutional neural networks (CNNs) a great deal of success in computer vision, this paper examines a less complicated, more practical backbone network in numerous complex prediction problems. In contrast to the newly planned developed for use with the Vision Transformer (ViT) Specifically, authors have offered the Pyramid for picture classification. Vision Transformer (PVT),a model that gets around obstacles a Transformer porting to various dense prediction tasks. PVT has a number of advantages over the present situation. the creative. (1) Unlike ViT, which often produces low-resolution produces a lot of outputs and uses a lot of memory PVT may be trained on dense partitions at low costs. A high output resolution of an image, which is crucial for detailed forecasting, but also employs a pyramid reduction to speed up huge feature computations maps. (2) PVT gains the benefits of both CNN and combining with Transformer to create a uniform framework for various dense prediction tasks and can be used for numerous visual tasks without convolutions as an exact substitute for CNN backbones. (3) Authors have verified that PVT has demonstrated through thorough research that it increases many downstream activities' performance, including segmentation by the object, instance, and semantics. PVT might, in theory, be used as an alternative and helpful framework for predictions at the pixel level and encourage additional research.

1 Introduction

The astounding results that Convolutional Neural Networks (CNNs) have they have had success with computer vision, making them a flexible and prevalent method for nearly all problems. However, this work seeks to investigate a backbone network other than CNN, which can be utilized for complex prediction tasks, including object detectiontion , semantic and instance segmentation , in addition to image classification [6].

The Vision Transformer was very recently introduced by Dosovitskiy et al. Transformation (ViT) for classifying images. This is a fascinating and serious effort to take over the CNN backbone using a model without convolutions. As depicted in Figure 1, (b), ViT contains coarse image patches and a columnar structure. Even if ViT can be used for picture classification, direct conversion to pixel-level is difficult. dense forecasts like object segmentation and detection, due to the fact that (1) its output feature map is single-scale, limited resolution, and (2) its memory and processing expenses even for common input image sizes, are comparatively high (e.g.,shorter edge of 800 pixels in the COCO benchmark). To overcome the aforementioned restrictions, this paper suggests a Pyramid Vision Transformer is a pure Transformer at its core. (PVT), a possible replacement for the CNN



(a) CNNs: VGG [54], ResNet [22], *etc.* (b) Vision Transformer [13] (c) Pyramid Vision Transformer (ours)

Figure 1: Figure 1: Comparisons of different architectures, where "Conv" and "TF-E" stand for "convolution" and "Transformer encoder", respectively. (a) Many CNN backbones use a pyramid structure for dense prediction tasks such as object detection (DET), instance and semantic segmentation (SEG). (b) The recently proposed Vision Transformer (ViT) [7] is a "columnar" structure specifically designed for image classification (CLS). (c) By incorporating the pyramid structure from CNNs, we present the Pyramid Vision Transformer (PVT), which can be used as a versatile backbone for many computer vision tasks, broadening the scope and impact of ViT. Moreover, our experiments also show that PVT can easily be combined with DETR [5] to build an end-to-end object detection system without convolutions.

Various downstream tasks, including image-level support, both pixel-level dense predictions and prediction. Specifically, Figure 1 (c) shows how our PVT circumvents the problems with the traditional Transformer by taking 44 pixels per patch, or fine-grained image patches, were used as input. to acquire the crucial knowledge of high-resolution representation for complicated prediction tasks; (2) introducing a reducing the pyramid to shorten the Transformers sequence as the network gets deeper, thereby lowering the computing expense, and (3) using a spatial-reduction approach to attention (SRA) layer to further cut down on resource usage when learning features with high resolution.

The proposed PVT has the following advantages overall. First, in comparison to the standard CNN backbones (see Figure 1 (a)), which have progressively larger local receptive fields Using the network depth, their PVT consistently generates a a more adequate global receptive field for detection segmentation, too. In addition, in contrast to ViT (see Figure 1 (b)), because of its sophisticated pyramidal design, their approach is easier to integrate into numerous representative RetinaNet [14] and other dense prediction pipelines R-CNN Mask [9]. Third, they are able to create a convolution-free use of PVT in conjunction with other task-specific Transformer decoders for objects, such as PVT+DETR [5] detection. As far as we are aware, this is the first fully pipeline for detecting objects without convolution.

2 Related Work

2.1 CNN Backbones

In vision recognition, CNNs are the workhorses of deep neural networks. To identify between handwritten numbers, the standard CNN was originally introduced in [12]. Convolutional kernels in the model have a specific receptive field that can capture advantageous visual context. The weights of convolutional kernels are distributed over the full picture space to guarantee translation equivariance. Recently, thanks to the quick advancement of computational resources (such the GPU), it has been practical to train stacked convolutional blocks on big picture classification datasets, like ImageNet. A convolutional operator with many kernel routes, for instance, can achieve very competitive performance, as shown by GoogleNet. The vision Transformer backbone is still in its early stages of development, unlike the fully developed CNNs. By creating a new versatile system, we attempt to broaden the use of Vision Transformer in this work.

2.2 Dense Prediction Tasks

Preliminary. The goal of the dense prediction challenge is to classify or regress a feature map at the pixel level. Two typical dense prediction challenges are object detection and semantic segmentation.

Object Detection.CNNs [12], which include single-stage detectors like SSD , RetinaNet [14], FCOS [, GFL, PolarMask , and OneNet , as well as multi-stage detectors like Faster R-CNN , Mask R-CNN [9], Cascade R-CNN [4], and Sparse R-CNN, have emerged as the industry standard for object detection in the deep learning era To achieve good detection performance, the majority of these well-known object detectors are based on high-resolution or multi-scale feature maps. Recently, the CNN backbone and the Transformer decoder were merged by DETR [5] and deformable DETR to provide an end-to-end object detector.For precise object detection, they also need high-resolution or multi-scale feature maps.

Semantic Segmentation.CNNs are crucial for semantic segmentation as well. For a given image of any size, FCN first developed a fully convolutional architecture to produce a spatial segmentation map. Following that, Noh et al. presented the deconvolution operation, which demonstrated outstanding performance on the PASCAL VOC 2012 dataset. UNet, which bridges the information flow between comparable low-level and high-level feature maps of the same spatial sizes, was proposed for the medical image segmentation domain especially and was inspired by FCN. Zhao et al. [17] designed a pyramid pooling module over multiple pooling scales, while Kirillov et al. [11] created a lightweight segmentation head known as Semantic FPN, based on FPN [13], to investigate richer global context representation.

2.3 SelfAttention and Transformer in Vision

Convolutional filter weights cannot be dynamically adjusted to different inputs because they are often set after training. Numerous solutions have been suggested to solve this issue, including self-attention procedures [16] and dynamic filters. The non-local block makes an effort to simulate long-range spatial and temporal dependencies, which has been found to be useful for precise video classification. The non-local operator is hampered by high computational and memory costs, notwithstanding its success. By creating sparse attention maps along a criss-cross path, criss-cross further minimizes complexity. Local self-attention units were suggested as a replacement for convolutional layers by Ramachandran et a.When the self-attention and convolutional processes are combined, AANet [3] produces results that are competitive. In place of the CNN's convolution, LambdaNetworks [2] uses the lambda layer, an effective self-attention technique. DETR [5] successfully does away with the necessity for custom procedures like NMS by modeling object identification as an end-to-end dictionary lookup problem with learnable queries. Deformable DETR builds on DETR and adds a deformable attention layer to concentrate on a small number of contextual cues for quicker convergence and improved performance. Recent image categorization software, Vision Transformer (ViT) [7], uses a pure Transformer [16] model and treats each picture as a series of patches. DeiT [15] uses a fresh distillation method to further extend ViT.Instead of a task-specific head or an image classification model, like in earlier models, this study integrates the pyramid structure into Transformer to present a pure Transformer backbone for dense prediction problems.

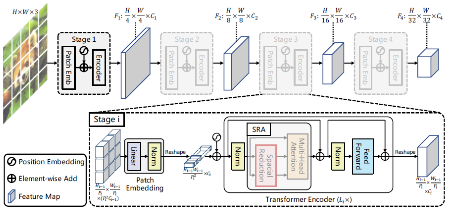

H×W×3
F_1: $\frac{H}{4}\times\frac{W}{4}\times C_1$
F_2: $\frac{H}{8}\times\frac{W}{8}\times C_2$
F_3: $\frac{H}{16}\times\frac{W}{16}\times C_3$
F_4: $\frac{H}{32}\times\frac{W}{32}\times C_4$
Stage 1
Patch Emb
Encoder
Stage 2
Stage 3
Stage 4
Stage i
Position Embedding
Element-wise Add
Feature Map
$\frac{H_{i-1}}{P_i}\times\frac{W_{i-1}}{P_i}\times(P_i^2C_{i-1})$
Linear
Norm
Patch Embedding
Reshape
$\frac{H_{i-1}W_{i-1}}{P_i^2}\times C_i$
SRA
Spacial Reduction
Multi-Head Attention
Norm
Feed Forward
Transformer Encoder ($L_i\times$)
$\frac{H_{i-1}}{P_i}\times\frac{W_{i-1}}{P_i}\times C_i$

Figure 2: **Overall architecture of Pyramid Vision Transformer (PVT).** The entire model is divided into four stages, each of which is comprised of a patch embedding layer and a L_i-layer Transformer encoder. Following a pyramid structure, the output resolution of the four stages progressively shrinks from high (4-stride) to low (32-stride).

3 Pyramid Vision Transformer (PVT)

3.1 Overall Architecture

In order to enable the Transformer framework to produce multi-scale feature maps for dense prediction tasks (such as item detection and semantic segmentation), we plan to incorporate the pyramid structure into it. Figure 2 depicts the PVT in general. Our technique comprises four phases that produce feature maps at various scales, much like CNN backbones [10]. The architecture of each stage is the same and consists of layers for the Li Transformer encoders and the patch embedding layer.

In the first stage, we first divide an input image of size $H \times W \times 3$ into $\frac{HW}{4^2}$ patches, each of which is $4 \times 4 \times 3$. The flattened patches are then fed through a linear projection to produce embedded patches that are $\frac{HW}{4^2} \times C1$ in size. After that, a Transformer encoder with L1 layers is used to encode the embedded patches and a position embedding, and the output is reshaped into a feature map F1 with dimensions $\frac{H}{4} \times \frac{W}{4} \times C1$. Similar to how we obtained feature maps F1, F2, and F3 with strides of 8, 16, and 32 pixels with regard to the input image, we also obtained feature maps F2, F3, and F4 using the feature map from the previous step as input.This method is simple to use for the majority of downstream tasks, such as picture classification, object recognition, and semantic segmentation, due to the feature pyramid F1, F2, F3, F4.

3.2 Feature Pyramid for Transformer

PVT uses a progressive shrinking technique to regulate the scale of feature maps by patch embedding layers, in contrast to CNN backbone networks [54, 22], which employ various convolutional steps to create multi-scale feature maps.

Here, we refer to the i-th stage's patch size as P_i. During step i the input feature map $F_{i+1} \in \mathbb{R}^{H_{i+1} \times W_{i+1} \times C_{i+1}}$ is first evenly divided into $\frac{H_{i+1} \times W_{i+1}}{P_i^2}$ patches, and then each patch is flattened and projected to a Ci-dimensional embedding.The shape of the embedded patches can be seen as $\frac{H_{i+1}}{P_i} \times \frac{W_{i+1}}{P_i} \times C_i$ after the linear projection, where the height and width are P_i times smaller than the input.By doing this, they were able to design a feature pyramid for Transformer by being able to adapt the scale of the feature map in each stage with flexibility.

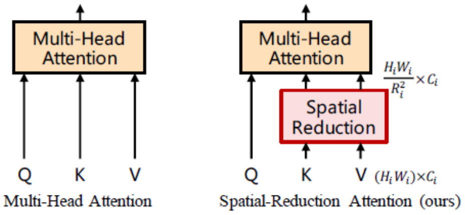


Figure 3: Figure 3: **Multi-head attention (MHA) vs. spatialreduction attention (SRA).** With the spatial-reduction operation, the computational/memory cost of our SRA is much lower than that of MHA.

3.3 Transformer Encoder

Each of the L_i encoder levels in the stage i of the Transformer encoder is made up of a feed-forward layer and an attention layer [16]. Authors have suggested a spatial-reduction attention (SRA) layer to replace the conventional multi-head attention (MHA) layer [16] in the encoder because PVT needs to handle high-resolution (e.g., 4-stride) feature maps. The SRA takes as inputs a query Q, a key K, and a value V and produces a refined feature, much like MHA. The distinction is that, prior to the attention operation, this SRA shrinks the spatial scale of K and V (see Figure 3), substantially reducing the computational and memory overhead. The SRA's stage i specifics can be expressed as follows:

$$SRA(Q, K, V) = Concat(head_0, ..., head_{N_i})W^O, \tag{1}$$

$$head_j = Attention(QW_j^Q, SR(K)W_j^K, SR(V)W_j^V), \tag{2}$$

Concat(.) is the concatenation operation used in [16], where it is used. The linear projection parameters are $W_j^Q \in \mathbb{R}^{C_i \times d_{head}}$, $W_j^K \in \mathbb{R}^{C_i \times d_{head}}$, $W_j^V \in \mathbb{R}^{C_i \times d_{head}}$, and $W^O \in \mathbb{R}^{C_i \times d_{head}}$ The attention layer's head number in Stage i is N_i. As a result, each head's dimension (d_{head}) is equal to $\frac{C_i}{N_i}$. The operation SR(.) reduces the spatial dimension of the input sequence (i.e., K or V), and is denoted by the following symbol:

$$SR(x) = Norm(Reshape(x, R_i)W^S), \tag{3}$$

Here, an input sequence is represented by $x \in \mathbb{R}^{(H_iW_i) \times C_i}$, and the reduction ratio of the attention layers in Stage i is shown by R_i. $Reshape(x, R_i)$ is an operation that transforms the input sequence x into a sequence of size $\frac{H_i W_i}{R_i^2} \times (R_i^2 C_i)$. Linear projection $W_s \in \mathbb{R}^{(R_i^2 C_i) \times C_i}$ decreases the input sequence's dimension to C_i. Layer normalization is referred to as Norm(.) [1]. The formula for our attention operation Attention(.) is the same as in the original Transformer [16].

$$Attention(\mathbf{q}, \mathbf{k}, \mathbf{v}) = Softmax(\frac{\mathbf{q}\mathbf{k}^{\mathbf{T}}}{\sqrt{d_{head}}})\mathbf{v}, \qquad (4)$$

Due to the fact that this attention operation's computational and memory costs are R_i^2 times lower than those of MHA, so SRA can handle larger input feature maps and sequences while using fewer resources.

4 Experimental Results

The most traditional imagelevel prediction problem is image classification. Authors have created a number of PVT models with various sizes, including PVT-Tiny(which we are going to use for our demo), -Small, -Medium, and -Large, whose parameter counts are comparable to ResNet18, 50, 101, and 152, respectively. In order to classify images, they have used a fully connected (FC) layer in accordance with ViT [7] and DeiT [15] to add a learnable classification token to the input of the previous stage and then perform classification on top of the token.

4.1 Image Classification

Settings. On the ImageNet 2012 dataset, which consists of 1.28 million training images and 50K validation images from 1,000 categories, image classification experiments are conducted. All models are trained on the training set and report the top-1 error on the validation set in order to allow for fair comparison.As data augmentations, they have followed DeiT [15] and used random cropping, random horizontal flipping, label-smoothing regularization, mixup, CutMix, and random erasing. They have a used AdamW with a momentum of 0.9, a mini-batch size of 128, and a weight decay of 5×10^{-2} to train models and optimize them. The cosine schedule is followed as the initial learning rate, which is set at 1×10^{-3}, gradually declines. On 8 V100 GPUs, all models are trained from scratch over 300 epochs. On the validation set, we apply a center crop to benchmark, cropping a 224 × 224 patch to assess the classification accuracy.

Method	#Param (M)	GFLOPs	Top-1 Err(%)
ResNet18	11.7	1.8	31.5
DeiT-Tiny/16	5.7	1.3	27.8
PVT-Tiny	13.2	1.9	24.9

Table 1: Image classification performance on the ImageNet validation set

Results. Table 1 shows that, for similar parameter values and computational budgets, PVT model outperform traditional CNN backbones. For example, when the GFLOPs are roughly similar, the top-1 error of PVT-Tiny reaches 24.9, which is a lot lower than both ResNet18 and DeiT-Tiny/16.

4.2 Object Detection

Settings. On the difficult COCO benchmark, object detection tests are run [40]. All models are tested on val2017 after being trained on COCO train2017 (118k pictures) (5k images). They tested the efficacy of PVT backbones on top of RetinaNet [14] and Mask R-CNN [9], two common detectors. Prior to training, they initialized the backbone using weights pre-trained on ImageNet, and the newly additional layers using Xavier [8]. Their models are optimized by AdamW with an initial learning rate of $1x10^{-4}$ and trained with a batch size of 16 on 8 V100 GPUs.

Results. As demonstrated in Table 2, when employing RetinaNet for object detection, we find that the PVT-based models greatly outperform their competitors under conditions with similar numbers of parameters. For instance, using the 1 training plan, PVT-AP Tiny's is 4.9 points higher than ResNet18's (36.7 vs. 31.8).

Backbone	#Param (M)	RetinaNet 1x			RetinaNet 3x + MS		
		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
ResNet18	21.3	31.8	49.6	33.6	35.4	53.9	37.6
PVT-Tiny	23.0	36.7(+4.9)	56.9	38.9	39.4(+4.0)	59.8	42.0

Table 2: Object detection performance on COCO val2017

Backbone	Semantic FPN		
	Param (M)	GFLOPs	mIoU (%)
ResNet18	15.5	32.2	32.9
PVT-Tiny	17.0	33.2	35.7(+2.8)

Table 3: Semantic segmentation performance on the ADE20K validation set

4.3 Semantic Segmentation

Settings. They used the difficult scene parsing dataset ADE20K to measure the effectiveness of semantic segmentation. With 20,210, 2,000 and 3,352 images for training, validation, and testing, respectively, ADE20K has 150 fine-grained semantic categories. We assess our PVT backbones using Semantic FPN [11], a straightforward segmentation technique devoid of dilated convolutions. The backbone is initialized in the training phase using weights that have already been trained on ImageNet [6], and additional recently added layers are initialized with Xavier [8]. We use AdamW to optimize our models, with a $le-4$ starting learning rate. In accordance with accepted procedures [32, 8], we train our models on four V100 GPUs for 80k iterations with a batch size of 16.With a power of 0.9, the learning rate decays in accordance with the polynomial decay schedule. For training, we randomly resize and crop the image to 512×512, then for testing, we rescale it to have a shorter side of 512 pixels.

Results. As seen in Table 3, PVT-based models regularly outperform models based on ResNet [10] or ResNeXt when utilizing Semantic FPN [11] for semantic segmentation.For instance, PVT-Tiny outperforms ResNet-18 by 2.8 points while having practically identical parameters and GFLOPs.

5 Demo

For our demo, we have choosen Malware Analysis dataset for image classification.

5.1 Malware and Benign binary files classification

Binary planting is an where the attacker places a binary file containing malicious code to a local or remote file system in order for a vulnerable application to load and execute it.Neural networks have proven to be effective in detecting the presence of malware in binaries. The sequence of binaries is converted into decimal numbers, where every number represent a pixel.

5.2 Dataset: Binary file to image

We have used dataset of Angelo Oliveira, November 7, 2019, ”Malware Analysis Datasets: Raw PE as Image”, IEEE Dataport, doi: https://dx.doi.org/10.21227/8brp-j220. Malware examples were obtained from virusshare.com. Benign examples were obtained from portableapps.com **Dataset preparation.** Binary files consisting of a series of zeros and ones have been read. The sequence was divided into 8 bit subsequences for each file, and each subsequence was converted into a decimal number between 0 and 255, where each decimal number represented a pixel value. Using the Nearest Neighbor Interpolation algorithm, images were rescaled to a 32 x 32 greyscale image and then flattened to a 1024 bytes vector.

5.3 Results

The precision of detecting malware, as well as the accuracy of classifying malware and benign samples, are used to assess PVT's effectiveness. PVT achieved 0.94 precision and 94.8 percent accuracy. Figure 4 depicts PVT's training performance on our dataset.

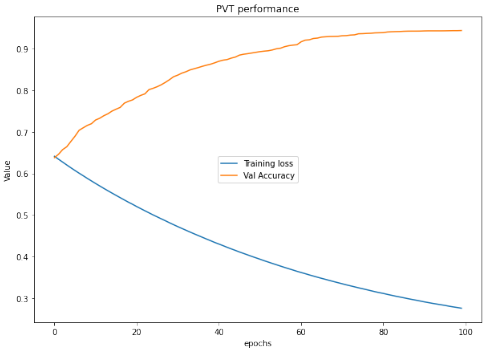


Figure 4: PVT performance on our dataset.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 5

[2] Irwan Bello. Lambdanetworks: Modeling long-range interactions without attention. *arXiv preprint arXiv:2102.08602*, 2021. 3

[3] 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

[4] 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. 2

[5] 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. 2, 3

[6] 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. 1, 6

[7] 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. 2, 3, 5

[8] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010. 5, 6

[9] 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. 2, 5

[10] 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. 4, 6

[11] 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. 3, 6

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 2

[13] 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. 3

[14] 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. 2, 5

[15] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 3, 5

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3, 4, 5

[17] 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. 3