

A Survey of Visual Transformers

Yang Liu*, Yao Zhang*, Yixin Wang*, Feng Hou*, Jin Yuan*,
 Jiang Tian, Yang Zhang, Zhongchao Shi, Jianping Fan, Zhiqiang He

Abstract—Transformer, an attention-based encoder-decoder architecture, has revolutionized the field of natural language processing. Inspired by this significant achievement, some pioneering works have recently been done on adapting Transformer-like architectures to Computer Vision (CV) fields, which have demonstrated their effectiveness on various CV tasks. Relying on competitive modeling capability, visual Transformers have achieved impressive performance on multiple benchmarks such as ImageNet, COCO and ADE20k as compared with modern Convolution Neural Networks (CNN). In this paper, we have provided a comprehensive review of over one hundred different visual Transformers for three fundamental CV tasks (classification, detection, and segmentation), where a taxonomy is proposed to organize these methods according to their motivations, structures, and usage scenarios. Because of the differences in training settings and oriented tasks, we have also evaluated these methods on different configurations for easy and intuitive comparison instead of only various benchmarks. Furthermore, we have revealed a series of essential but unexploited aspects that may empower Transformer to stand out from numerous architectures, e.g., slack high-level semantic embeddings to bridge the gap between visual and sequential Transformers. Finally, three promising future research directions are suggested for further investment.

Index Terms—Visual Transformer, self-attention, encoder-decoder, visual recognition, survey.

I. INTRODUCTION

TRANSFORMER [1], as an attention-based structure, has first demonstrated the tremendous force in sequence modeling and machine translation tasks. As illustrated in Fig. 1, Transformers have gradually emerged as predominant deep learning models for natural language processing (NLP). The most recent dominant models are some self-supervised Transformers pre-trained from sufficient datasets and then fine-tuned on the small and specific downstream tasks [2]–[9]. The Generative Pre-trained Transformer (GPT) families [2]–[4] leverage the Transformer decoder to perform the auto-regressive language modeling tasks, while the Bidirectional Encoder Representations from Transformers (BERT) [5] and its variants [6], [7] serve as auto-encoder language models built on the Transformer encoder.

In the Computer Vision (CV) fields, prior to visual Transformer models, Convolution Neural Networks (CNN) has

Yang Liu, Yao Zhang, Yixin Wang, and Feng Hou are with Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100000, China and also with University of Chinese Academy of Sciences, Beijing, 100000, China (e-mail: liuyang20c@mails.ucas.ac.cn).

Jin Yuan is with Southeast University, Nanjing, 214135, China.

*The work was done at AI Lab, Lenovo Research.

Jiang Tian, Zhongchao Shi, and Jianping Fan are with AI Lab, Lenovo Research, Beijing, 100000, China.

Yang Zhang, Zhiqiang He are with Lenovo Ltd., Beijing, 100000, China (e-mail: hezq@lenovo.com; zhangyang20@lenovo.com).

Corresponding authors: Zhiqiang He; Yang Zhang.

emerged as a dominant paradigm [10]–[12]. Inspired by the great success of the self-attention mechanism in NLP [1], [13], some CNN-based models attempted to capture the long-range dependencies via an additional self-attention layer in either spatial [14]–[16] or channel level [17]–[19], while others try to replace the traditional convolutions entirely with global [20] or local self-attention blocks [21]–[25]. Although Cordonnier et al. theoretically demonstrated the effectiveness and efficiency of self-attention block [26], these pure attention models have still been inferior to the current state-of-the-art (SOTA) CNN models on prevailing benchmarks.

As mentioned above, the attention-based models have received a surge of attentions in the visual recognition field, while vanilla Transformer achieving great successes in the NLP field. Inspired by these, numerous works have recently migrated Transformer to the CV tasks and achieved comparable results. For example, Dosovitskiy et al. [27] propose a pure Transformer by using image patches as the input for image classification, which has achieved SOTA on many image classification benchmarks. Additionally, visual Transformers have also obtained great performance for other CV tasks, such as detection [28], segmentation [29], tracking [30], image generation [31], and enhancement [32].

As illustrated in Fig. 1, following [27], [28], hundreds of Transformer-based models have been proposed for various areas tremendously within the last year. Thus, a systematic literature survey is strongly desired to identify, categorize, and critically evaluate the performance of these newly emerged visual Transformers. Considering that the readers may come from different areas, we focus on these existing visual Transformers for three fundamental CV tasks, including classification, detection, and segmentation. As illustrated in Fig. 2, this survey categorizes all these existing methods into multiple groups according to their tasks, motivations, and structural characteristics. Some of them may partially overlap. For example, some kinds of improvements not only enhance the performance of backbone in image classification but also in dense prediction tasks (i.e., detection and segmentation), and many deep and hierarchical methods are also implemented by the improvements of both CNN and attention.

Several reviews of Transformer have been published in the last year, where Tay et al. [86] review the efficiency of Transformers in NLP, Khan et al. [87] and Han et al. [88] summarize the early visual Transformers and previous attention models, as well as some language models without a systematic approach. The most recent review of Transformer is introduced by Lin et al., which provides a systematic review of various variants of Transformers and mentions visual applications sketchily [89]. Based on these observations, this paper aims to provide a full review of the recent visual Transformers and to categorize

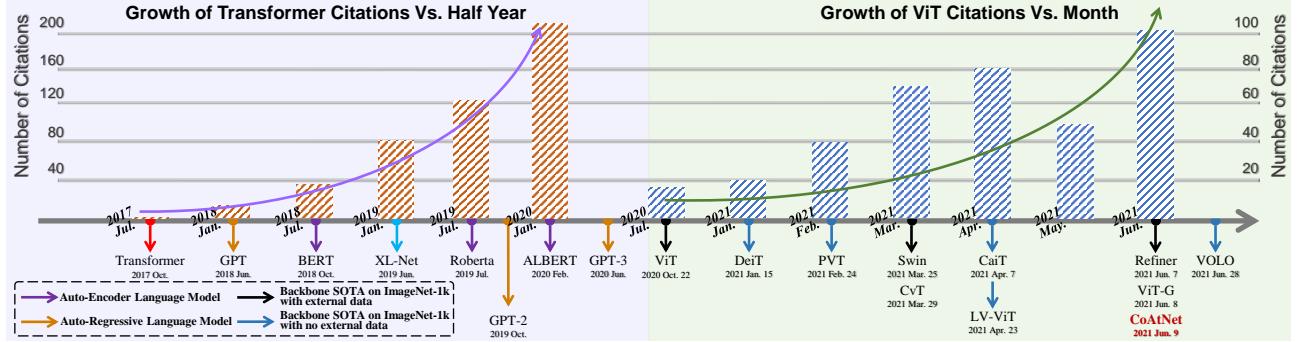


Fig. 1. Odyssey of Transformer application & Growth of both Transformer [1] and ViT [27] citations according to Google Scholar. (Upper Left) Growth of Transformer citations in multiple conference publication including: NIPS, AICL, ICML, IJCAI, ICLR, and ICASSP. (Upper Right) Growth of ViT citations in Arxiv publications. (Bottom Left) Odyssey of language model [1]–[8]. (Bottom Right) Odyssey of visual Transformer backbone where the black [27], [33]–[37] is the SOTA with external data and the blue [38]–[42] refers to the SOTA without external data (best viewed in color).

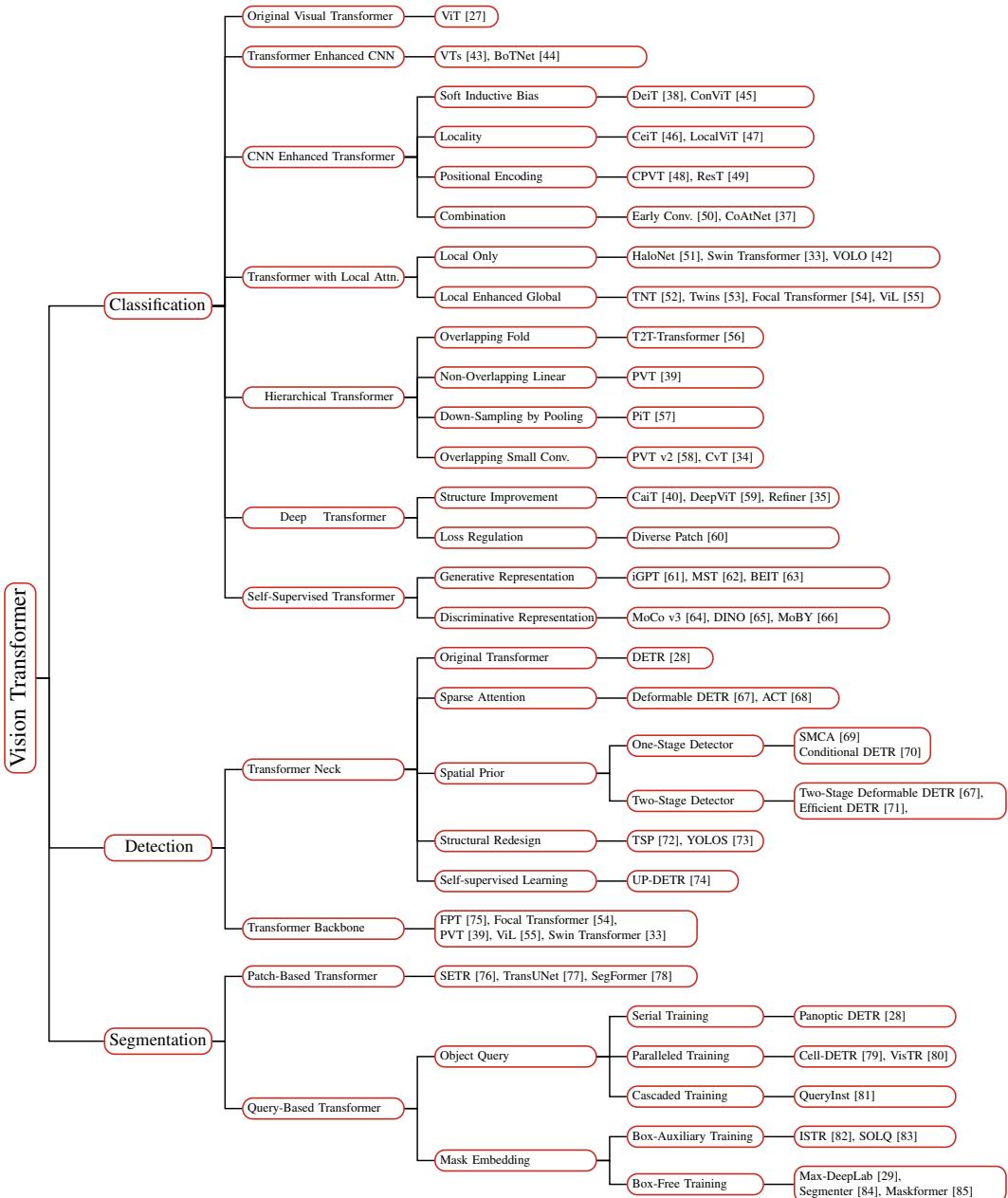


Fig. 2. Taxonomy of Transformers

these existing methods systematically:

- (1) *Comprehensiveness & Readability.* This paper comprehensively reviews more than 100 visual Transformers for three basic tasks: classification, detection, and segmentation. We select and analyze more than 50 representative models as summarized in Fig. 2. We not only make an exhausted analysis for each model in a single perspective but also build their internal connection by certain senses such as progressive, contrastive, and multi-view analysis.
- (2) *Intuitive Comparison.* As these Transformers follow different training schemes and hyper-parameter settings in various tasks, this survey presents multiple lateral comparisons by separating them over different datasets and restrictions. More importantly, we summarize a series of promising components designed for each task, including: *shallow local convolution with hierarchical structure for backbone*, *spatial prior acceleration with sparse attention for neck detector*, and *general-purpose mask prediction scheme for segmentation*.
- (3) *In-depth Analysis.* We further provides significant insights in the following aspects: the transformation process from sequential to visual tasks, the correspondence between Transformer and other visual networks, and the correlation of learnable embeddings (i.e., class token, object query, mask embedding) adopted in different tasks. Finally, we outline the future research directions. For example, the encoder-decoder Transformer backbone can unify the three sub-tasks via learned embeddings.

The rest of this paper is organized as follows. An overview architecture and critical components for the original Transformer are introduced in Sec. II. A comprehensive taxonomy of Transformer backbones is summarized in Sec. III with a brief discussion for image classification. We then review contemporary Transformer detectors, including Transformer necks and backbones in Sec. IV. Sec. V clarifies the mainstream Transformer variants in the segmentation field according to embeddings' form (i.e., patch embedding and query embedding). Moreover, Sec. II- IV also briefly analyze a specific aspect of their corresponding fields with performance evaluation. Sec. VI provides three aspects for further discussion and points out future research directions for further investigation.

II. ORIGINAL TRANSFORMER

The original Transformer [1] is first applied to the sequence-to-sequence auto-regressive tasks. Compared with previous sequence transduction models [90], [91], Transformer inherits the encoder-decoder structure but discards recurrence and convolutions entirely by using *multi-head attention mechanisms* and *point-wise feed-forward networks*. In the following sub-sections, we will describe the four key components and provide an architectural overview of the original Transformer.

A. Attention Mechanism

As an essential component of Transformer, the attention mechanism can be grouped into two parts. 1) *A transformation layer* maps input sequences $X \in \mathbb{R}^{n_x \times d_x}$, $Y \in \mathbb{R}^{n_y \times d_y}$ to three different sequential vectors (query Q , key K , and value

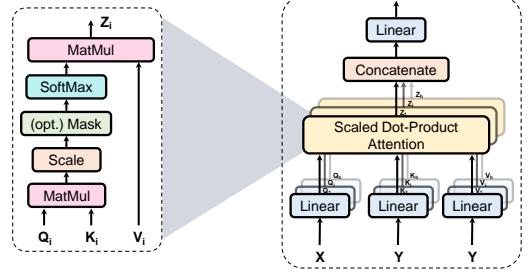


Fig. 3. The structure of the attention layer. Left: Scaled Dot-Product Attention. Right: Multi-Head Attention Mechanism.

V), where n and d are the length and dimension of the input sequences, respectively. Each vector is generated as

$$Q = XW^Q, \quad K = YW^K, \quad V = YW^V, \quad (1)$$

where $W^Q \in \mathbb{R}^{d_x \times d^k}$, $W^K \in \mathbb{R}^{d_y \times d^k}$, and $W^V \in \mathbb{R}^{d_y \times d^v}$ are linear matrices, d^k is dimension of the query and key, and d^v is dimension of the value. The query is projected from X , while the key and value are projected from Y . This two sequences input scheme is used to refer to as cross-attention mechanism. Specifically, it can be regarded as self-attention when $Y = X$. Additionally, self-attention is applied to both encoder and decoder, while cross-attention severs as a junction within the decoder. 2) *An attention layer*, as shown in Fig. 3, explicitly aggregates the query with the corresponding key, assigns them to the value, and updates the output vector. We can formulate the process into a unified function as

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (2)$$

where the attention weights are generated by a dot-product operation between the query and the key, a scaling factor $\sqrt{d_k}$ and a softmax operation are supplied to translate the attention weights into a normalized distribution. The resulting weights are assigned to the corresponding elements of the value, thereby yielding the final output vector.

B. Multi-Head Attention Mechanism

Because of the restricted feature subspace, the modeling capability of a single-head attention block is coarse. To tackle this issue, as shown in Fig. 3, Vaswani et al. propose a multi-head self-attention mechanism (MHSA) that linearly projects the input into multiple feature subspaces and processes them by several independent attention heads (layers) parallelly. The resulting vectors are concatenated and mapped to the final output. The process of MHSA can be formulated as

$$\begin{aligned} Q_i &= XW^{Q_i}, \quad K_i = XW^{K_i}, \quad V_i = XW^{V_i}, \\ Z_i &= \text{Attention}(Q_i, K_i, V_i), \quad i = 1 \dots h, \\ \text{MultiHead}(Q, K, V) &= \text{Concat}(Z_1, Z_2, \dots, Z_h)W^O, \end{aligned} \quad (3)$$

where h is the head number, $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ denotes the output projected matrix, Z_i denotes the output vector of each head, $W^{Q_i} \in \mathbb{R}^{d_{model} \times d_k}$, $W^{K_i} \in \mathbb{R}^{d_{model} \times d_k}$, and $W^{V_i} \in \mathbb{R}^{d_{model} \times d_v}$ are three different groups of linear matrices.

Similar to the sparse connection of convolution, the multi-head attention separates the input into h independent attention heads with d_{model}/h -dimensional vectors, and integrates each head feature parallelly. Without extra computational costs, multi-head attention enriches the diversity of feature subspace.

C. Position-wise Feed-Forward Networks

The output of MHSAs is then fed into two successive feed-forward networks (FFN) with a ReLU activation as

$$\text{FFN}(x) = \text{RELU}(W_1x + b_1)W_2 + b_2. \quad (4)$$

This position-wise feed-forward layer can be viewed as a point-wise convolution which treats each position equally but uses different parameters between each layer.

D. Positional Encoding

Since Transformer/Attention operates on the input embedding simultaneously and identically, the order of the sequence is neglected. To make use of the sequential information, a common solution is to append an extra positional vector to the inputs, hence the term “positional encoding”. There are many choices of positional encoding. For example, a typical choice is sine and cosine functions of different frequencies as

$$PE_{(pos,i)} = \begin{cases} \sin(pos \cdot \omega_k) & \text{if } i = 2k \\ \cos(pos \cdot \omega_k) & \text{if } i = 2k + 1, \end{cases} \quad (5)$$

$$\omega_k = \frac{1}{10000^{2k/d}}, \quad k = 1, \dots, d/2,$$

where i and d are the index and length of the vector, respectively, and pos is the position of each element in the sequence.

E. Transformer Model

Fig. 4 shows the overall encoder-decoder architecture of the Transformer model. Specifically, Transformer consists of $N = 6$ successive encoder blocks, each of which is composed of two sub-layers. An MHSAs layer aggregates the relationship within encoder embeddings. A position-wise FFN layer extracts feature representation. For the decoder, it also involves six consecutive blocks that follow the stack of encoders. Compared with the encoder, each decoder block appends to a multi-head cross-attention layer to aggregate the decoder embeddings and outputs of the encoder, where Y corresponds to the former, and X is the latter as shown in Equation (1). Moreover, all of the sub-layers in the encoder and decoder employ a residual connection [11] and a Layer Normalization [92] to enhance the scalability of Transformer. In order to record the sequential information, each input embedding is attached with a positional encoding at the beginning of the encoder and decoder stacks. Finally, a linear layer and a softmax operation are used for the next word prediction.

As an auto-regressive language model, Transformer originated from machine translation tasks. Given a sequence of words, Transformer vectorizes the input sequence into word embeddings, adds positional encodings, and feeds the resulting sequence of vectors to an encoder. During training, as illustrated in Fig. 4, Vaswani et al. design a masking operation

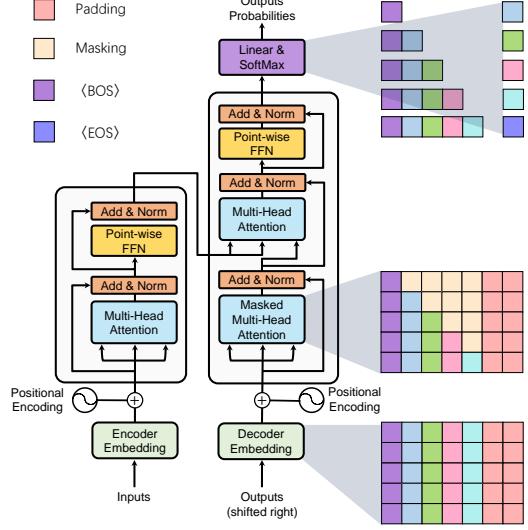


Fig. 4. The overall architecture of Transformer [1] that follows an encoder-decoder structure. The 2D lattice visualizes the states of each part of the decoder during training (best viewed in color).

according to the auto-regressive task rule that the current position can only depend on the previous position outputs. Based on this masking, Transformer decoder is able to process the input label sequence parallelly. During the inference time, the previous prediction word sequence processed by the same operation is fed into the decoder to generate the next word.

III. TRANSFORMER FOR CLASSIFICATION

Inspired by the prominent developments of Transformer in NLP [2]–[5], [8], some researchers attempt to introduce Transformer into image classification. Vision Transformer (ViT) [27] first achieves similar or even superior performance on mainstream classification benchmarks as compared with traditional CNNs. This section comprehensively reviews more than 40 Transformer backbones which are published before Jun. 2021, and they are grouped into six categories according to their motivations and implementations, as shown in Fig. 5. Based on our taxonomy, we start with introducing ViT, the *Original Visual Transformer* for image classification. Then we discuss *Transformer Enhanced CNN* methods that utilize Transformer to enhance the long-range dependency for CNN backbone. Transformers have strong capability on global modeling but neglect local information at the early stage. Therefore, *CNN Enhanced Transformer* methods leverage an appropriate convolutional inductive bias to augment the Transformer, while *Local Attention Enhanced Transformer* methods redesign the patch partition and attention block to enhance the locality of the Transformer and maintain a convolution-free architecture. Moreover, CNN empirically benefits from hierarchical and deep structures in both performance and computational efficiency [93]. Inspired by these, *Hierarchical Transformer* and *Deep Transformer* methods are proposed. The former replaces fixed-resolution columnar structure with a pyramid stem, while the latter prevents the attention map from over-smooth and increases its diversity in the deep layer. Additionally, we also review currently available self-supervision

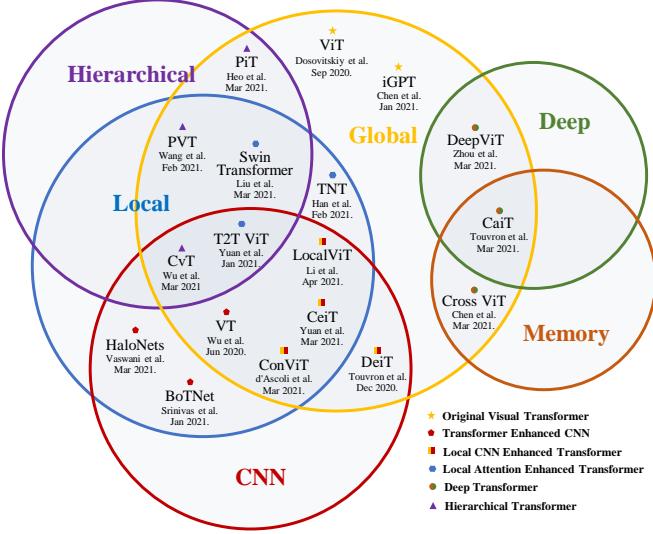


Fig. 5. Taxonomy of Visual Transformer Backbone (best viewed in color).

methods for visual Transformer. Finally, we evaluate these Transformers performance, analyze promising improvements, and answer a common question for further investigation.

A. Original Visual Transformer

ViT, proposed by Dosovitskiy et al., is the first Transformer backbone for image classification [27]. Because the vanilla Transformer requires a token sequence input, the input image is firstly split into a series of non-overlapped patches and then projected into patch embeddings. Similar to the original operations of Transformer, a 1D learnable positional encoding is added to each patch to retain their spatial information, and the joint embeddings are fed into the encoder (Fig. 6). Similar to BERT [5], ViT inserts a learned [*class*] embedding whose state at the output of the Transformer encoder serves as a representation to perform classification. Moreover, a 2D interpolation complements the pre-trained positional encoding to maintain the consistent order of patches when feeding images with arbitrary resolution. By pre-training with a large-scale private dataset (JFT-300M consisting of 300 million images), ViT achieves similar or even better results than most prevailing CNN methods on multiple image recognition benchmarks (i.e., ImageNet, CIFAR-10, and CIFAR-100). ViT has demonstrated the effectiveness of Transformer in CV tasks, although it is unable to generalize well with insufficient training data.

B. Transformer Enhanced CNN

As mentioned before, Transformer has two key parts: *MHSA* and *FFN*. Recently, Cordonnier et al. have proved that a convolutional layer can be approximated by MHSA with sufficient heads [26]. Dong et al. have shown that MHSA may possess a strong inductive bias towards “token uniformity” without skip connection and FFN [94]. Therefore, Transformer has theoretically more powerful modeling capability than CNN. However, it has inevitably heavy computational cost, especially for the shallow layers, brought by the self-attention

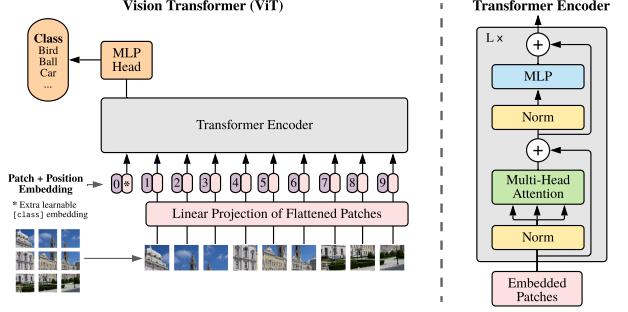


Fig. 6. Illustration of ViT. The flatten image patches with an additional class token are fed into the vanilla Transformer encoder after positional encoding. Only the class token can be predicted for classification. (from [27].)

mechanism, which grows quadratically with the feature resolutions. Similar to previous attention-based methods [14], [22], [95], some methods attempt to insert Transformer into CNN backbones or replace part of convolution blocks with Transformer layers [43], [44].

VTs: Considering that Convolution equivalently matches all pixels regardless of their priority, Visual Transformer (VT) [43] decouples semantic concepts of the input image into different channels and relates them densely through Transformer encoder blocks. In detail, a VT-block consists of three parts. 1) A tokenizer decouples the inputs into different semantic sets of visual tokens by using a scaling attention layer. 2) A Transformer encoder aggregates semantic information between these visual tokens. 3) A projector reintegrates original pixel-space feature implemented by a tokens-image cross-attention layer. Moreover, it builds alternative Visual-Transformer-ResNets (VT-ResNets) by substituting the last convolution stage of ResNet [11] with VT-blocks. Compared with standard ResNet, VT-ResNet achieves 4.6-7% higher accuracy on ImageNet with fewer parameters and FLOPs.

BoTNet: Compared with previous attention-based modules that only structurally replace the final-stage convolutions with attention blocks, Vaswani et al. propose a conceptual redefinition that the successive bottleneck blocks with self-attention mechanism can be regarded as Bottleneck Transformer (BoTNet) [44] blocks, although the short-connection form is different. Inspired by the effectiveness of relative position encoding [96] in [22]–[24], [95], BoTNet leverages this positional awareness to further approximate Transformer. Based on ResNet-50, BoTNet outperforms most CNN models with similar parameter settings on the ImageNet benchmark, achieving 84.7% top-1 accuracy with 75.1M parameters. This mimic Transformer methods further demonstrates the efficacy of Transformer on the top of standard convolution models.

C. CNN Enhanced Transformer

Inductive bias can be expressed as a set of assumptions about either the data distribution or the solution space, whose manifestations within convolution are locality and translation invariance. Locality focuses on the spatially close elements and isolates them from the distal. Translation invariance manifests the reuse of the same matching rules across localities of the inputs [97]. Because the covariance within local neighborhoods

is large and tends to be gradually stationary across an image, these convolutional biases can process image data effectively. Nevertheless, strong biases also limit the upper bound of CNN with sufficient datasets. Recent efforts attempt to leverage an appropriate convolutional bias to enhance Transformer and accelerate its convergence. These applications can be summarized as follows: soft approximation [38], [45], direct locality processing [46], [47], direct replacements of the positional encoding [48], [49], and structural combinations [37], [50].

DeiT: To moderate the dependence of ViT on the large datasets, Touvron et al. propose a Data-efficient image Transformer (DeiT) [38] to improve its applicability when training on ImageNet-1k. Based on ViT-B [27], DeiT-B achieves 83.1% top-1 accuracy on ImageNet by using existing data augmentation and regularization strategies. Moreover, a teacher-student strategy is applied in pre-training, which is a distilled token similar to a class token in form but supervised by pseudo-labels of the teacher. Empirically, CNN is a better teacher than Transformer, and an interesting finding is that the distilled model outperforms its teacher. These observations can be explained by [98]: the CNN teacher can transfer its inductive bias in a soft way to the Transformer student through knowledge distillation. Based on this token-based distillation method, DeiT-B attains the top-1 accuracy of 85.2% with no external data. Furthermore, it is worthy to study a problem whether co-training [99] can be used as a new way to integrate inductive bias for Transformer in semi-supervision.

ConViT: Similar to dual-path attention-based models [22], [100], ConViT [45] attaches a parallel convolution branch to Transformer branch to impose convolutional inductive biases softly via a Gated Positional Self-Attention (GPSA). Specifically, GPSA can be divided into vanilla self-attention weights and mimic convolutional weights. It is firstly initialized to approximate the locality of convolutional layers and then explicitly gives each attention head freedom to escape locality by adjusting a learned gating parameter. The function of GPSA can be represented as

$$\begin{aligned} \mathbf{A}_{ij}^h := & (1 - \sigma(\lambda_h)) \operatorname{softmax} (\mathbf{Q}_i^h \mathbf{K}_j^{h\top}) \\ & + \sigma(\lambda_h) \operatorname{softmax} (\mathbf{v}_{pos}^{h\top} \mathbf{r}_{ij}), \end{aligned} \quad (6)$$

where $\mathbf{v}_{pos}^h \in \mathbb{R}^{D_{pos}}$ is a learned embedding to mimic convolution, $\mathbf{r}_{qk} \in \mathbb{R}^{D_{pos}}$ is a fixed relative position embedding, λ_h is a learnable gating parameter that regulates the attention paid to position versus content information. After all, ConViT surpasses the DeiT by 0.6-3.2% Top-1 accuracy on ImageNet.

CeIT & LocalViT: Besides the methods learning inductive bias softly [38], [45], there are also some straightforward methods. CeIT [46] and LocalViT [47] extract the locality by directly adding a depth-wise convolution in FFN. Based on the equivalence between point-wise convolution and position-wise FFN, LocalViT extends this convolutional version FFN to an inverted residual block [101] to build a depth-wise convolutional framework. With the same operations, CeIT also redesigns a patch-to-tokens scheme and attaches a Layer-wise Class token Attention (LCA) at the top of the Transformer to aggregate multi-level representations. In this way, both of them yield higher performance than original DeiT [38].

CPVT & ResT: Some methods try using the inherent positional information of convolution to generalize variable-resolution inputs. From one perspective, ResT [49] assumes that there is a correlation between positional encodings and inputs. Thus, the summation operation between them can be viewed as a multiplication that weights a pixel-wise input, implemented by a 3×3 depth-wise convolution with padding of 1. From the other perspective, following the observation of [102] that the borders of convolution with zero paddings can encode the absolute positional information, CPVT [48] replaces positional encoding with a series of convolutions. Both methods benefit from this convolutional position embedding, especially when the model is small, boosting ResT-Lite and PVT-Tiny by 1.3% and 1.4%, respectively.

Early Conv. & CoAtNet: In addition to the “internal” fusion, more current methods focus on an “apparent” combination according to different visual Transformer’s structures. For standard columnar structure, Xiao et al. substitute the original patchify stem (single non-overlapped large kernel) with several stacked stride-2 3×3 kernels [50]. This conceptually simple yet powerful stem improves 1-2% top-1 accuracy on ImageNet-1k and facilitates the stability and generalization of ViT for downstream tasks. For hierarchical structures, Dai et al. [37] investigate an optimal combination of hybrid models to benefit performance trade-off. By comparing a series of hybrid models, they propose a Convolution and Attention Network (CoAtNet) to join the strength of both CNN and Transformer. Concretely, they observe that depth-wise convolution can be naturally integrated into the attention block, and vertically stacking convolution in the shallow layer is more effective than original hierarchical methods. CoAtNet achieves the latest SOTA performance across multiple datasets.

D. Local Attention Enhanced Transformer

ViT [27] regards the input image as a sequence of patches. This coarse patch embedding process neglects the gap between language and image, which may damage local information of the image. As a local extractor, convolution aggregates features through a relatively fixed filter. This template matching process can handle most small datasets efficiently but face a combinatorial explosion of representations when processing a huge dataset. Compared with convolution, the local attention mechanism can dynamically generate attention weights according to the relationship between local elements [51], [95]. To enhance local feature extraction capability and retain a convolution-free structure, some efforts [33], [52], [53], [56] attempt to accommodate the patch structure by local self-attention mechanism.

TNT: ViT [27] only focuses on the global patch aggregation but neglects its internal interaction. Similar to Network In Network (NIN) series [103], Han et al. leverage a Transformer-iN-Transformer (TNT) [52] model to aggregate both patch- and pixel-level representations. Specifically, each layer of TNT consists of two successive blocks, an inner block models the pixel-wise interaction within each patch, and an outer block extracts the global information from patch embeddings. They are linked by a linear projection layer that maps the pixels

to their corresponding patch. Therefore, more richer local features are preserved by TNT at the shallow layer than before.

Swin Transformer: Temporal Shift Module (TSM) [104] facilitates information interchange among neighboring frames by shifting part of the channels along the temporal dimension (Fig. 7(a)). Similar to a 2D TSM, Liu et al present a Shifted windows (Swin) Transformer [33] that utilizes a shifted window along the spatial dimension to model global and boundary features. In detail, a hierarchical structure performs spatial reduction and channel extension, implemented by patch partition and patch merging operations. Furthermore, two successive window-wise attention layers facilitate cross-window interactions (Fig. 7(b)-(c)), similar to the receptive field expansion concept in CNN. It reduces the computational complexity from $O(2n^2C)$ to $O(4M^2nC)$ in the attention layer, where n and M denotes the patch length and window size, respectively. Swin Transformer achieves 84.2% top-1 accuracy on ImageNet-1K and the latest SOTA on multiple dense prediction benchmarks, e.g., COCO and ADE20k.

Twins & ViL: As a local-global separate Transformer, Twins [53] replaces the complicated design of Swin Transformer [33] with a Spatially Separable Self-Attention mechanism (SSSA). It is formally similar to a depth-wise convolution [101] or a window-wise TNT block [52]. In detail, a local attention layer aggregates neighboring patches within each sub-window to enhance fine-grained features, and a global sub-sampled attention layer is devoted to capturing long-distance features. The other separate form is ViL [55] that replaces the single global token with a series of local embeddings (termed as global memory). Each local embedding only interacts with others and their corresponding 2D spatial neighbors. Benefiting from such a simple form, both achieve competitive performance with typical Swin Transformer.

VOLO: Vision Outlooker (VOLO) [42] uses outlook attention to focus on finer-level features than other attention-based modules. Formally, it consists of three operations: unfold, linear-wights attention, and refold. The pipeline is similar to a patch-wise dynamic convolution although VOLO emphasizes it as a CNN-free model. Based on LV-ViT [41], VOLO improves LV-ViT by 0.4%-1.2% top-1 accuracy, a new SOTA on ImageNet-1k benchmark with no external data.

E. Hierarchical Transformer

As ViT [27] inherits the original columnar structure with a fixed resolution across the entire network, it neglects the fine-grained features and brings heavy computational costs. Following hierarchical CNNs, recent works [34], [39], [55]-[57] apply a similar structure to Transformer.

T2T-ViT: The paradigm of hierarchical Transformer is first introduced by Tokens-to-Token ViT (T2T-ViT) ViT [56]. In T2T-ViT, a layer-wise T2T transformation is used to aggregate neighboring tokens into one single token. Such surrounding aggregation implemented by an overlapping unfold operation can perform hierarchical structure and locality simultaneously. However, the transformation layer is burdensome for memory and computation because of the overlapped redundancy.

PVT: Another exemplification of hierarchical Transformer is Pyramid Vision Transformer (PVT) [39]. As mentioned

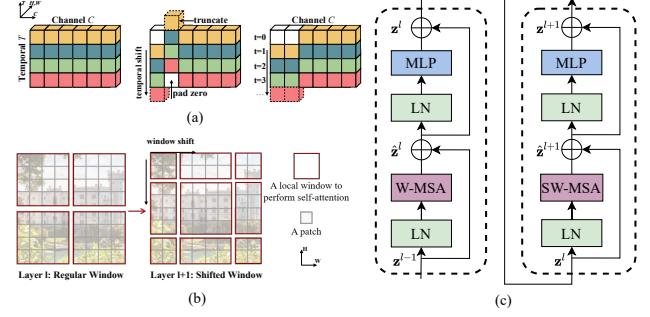


Fig. 7. An overview of Swin Transformer and TSM. (a) TSM with bi-direction and uni-direction operation. (b) The shifted window method. (c) Two successive Transformer blocks of Swin Transformer. The regular and shifted window correspond to W-MSA and SW-MSA, respectively. (from [33], [104]).

before, the reuse of redundant tokens leads to T2T-ViT's inefficiency. Unlike the overlapping unfold transformation, PVT relies on a non-overlapping patch partition to reduce the sequence length and a linear patch embedding to keep the channel dimensions consistent. This pyramid architecture can adapt Transform to dense prediction tasks that demand large inputs and fine-grained features with computational efficiency. Specifically, the spatial-reduction attention (SRA) layer is applied to significantly reduce the computational complexity by learning low-resolution key-value pairs in each attention block (Fig. 9). PVT demonstrates the availability of hierarchical Transformer on many benchmarks.

PiT & CvT: Similar to the shrinking strategy of PVT [39], Pooling-based Vision Transformer (PiT) [57] and Convolutional vision Transformer (CvT) [34] utilize pooling and convolution to perform token embedding, respectively. Moreover, CvT improves the SRA of PVT by replacing the linear layer with a convolutional projection. Based on these local context information introduced by convolutions, CvT can generalize the variable-inputs without positional encoding.

F. Deep Transformer

Empirically, increasing model's depth enables networks to learn more complex representations [11]. Recent works apply such a deep structure to Transformer and conduct massive experiments to investigate its scalability by analyzing cross-patch [60] and cross-layer [35], [59] similarities, as well as the contribution of residual blocks [40]. In deep Transformer, the features in deeper layers tend to be less representative (attention collapse [59]), and patches are mapped into indistinguishable latent representations (patch over-smoothing [60]). To compensate for the above limitations, these methods also present corresponding solutions in multiple aspects.

CaiT: From the structure aspect, Touvron et al. present efficient Class-attention in image Transformers (CaiT [40]), including two stages. 1) Multiple self-attention stages without class token. In each layer, a learned diagonal matrix initialized by small values is exploited to update the channel weights dynamically, thereby offering a degree of freedom for channel adjustment. 2) Last few class-attention stages with frozen patch embeddings. A later class token is inserted to model

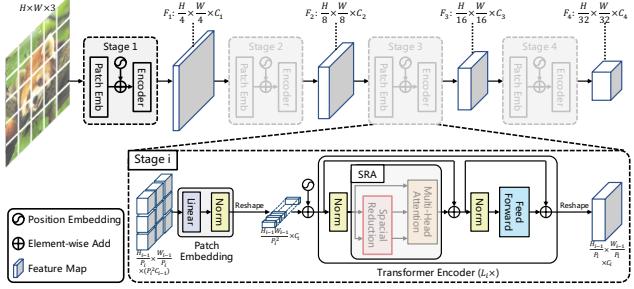


Fig. 8. An overview of PVT. The hierarchical structure consists of four stages. Each stage is composed to patch partition, patch embedding, L_i -layer Transformer encoder and reshape operation (from [39]).

global representations, similar to DETR with an encoder-decoder structure (Fig. 9(c)). This explicit separation is based on the assumption that the class token is invalid for the gradient of patch embeddings in the forward pass. With distillation training strategy [38], CaiT achieves a new SOTA on imagenet-1k (86.5% top-1 accuracy) with no external data.

DeepViT & Refiner: Deep Transformer suffers from attention collapse and over-smoothing problems, but still largely preserves the diversity of attention map between different heads. Based on this observation, Zhou et al. propose Deep Vision Transformer (DeepViT) [59] that aggregates cross-head attention maps and re-generates a new one by using a linear layer to increase cross-layer feature diversity. Furthermore, Refiner [35] applies a linear layer to expand the dimension of the attention maps (indirectly increase heads number) for diversity promotion. Then, a Distributed Local Attention (DLA) is employed to better model local and global features, which is implemented by a head-wise convolution effecting on the attention map. After all, Refiner achieves 86% top-1 accuracy on ImageNet with 81M parameters.

Diverse Patch: From the aspect of training strategy, Gong et al. present three patch-wise loss functions for deep Transformer that can significantly encourage patches' diversity and offset oversmoothing problem [60]. Similar to [105], a patch-wise cosine loss minimizes pairwise cosine similarity among patches. A patch-wise contrastive loss regularizes the deeper patches by their corresponding one in the early layer. Inspired by Cutmix [106], a patch-wise mixing loss mixes two different images and forces each patch to only attend to the patches from the same image and ignore unrelated ones. Compared with LV-ViT [41], they have similar loss function but distinctive motivations. The former focuses on the patch diversity, while the latter focuses on data augmentation about token labeling.

G. Transformers with Self-Supervised Learning

Self-supervised Transformers are successful in NLP [5], but the supervised pre-trained Transformers still dominate CV area [33], [38]. Recent works also attempt to design various self-supervised learning schemes for visual Transformer in the generative [61]–[63] and discriminative [64]–[66].

iGPT: For generative tasks, Chen et al. propose an image Generative Pre-training Transformer (iGPT) [61] for visual self-supervised learning. Different from the patch embedding

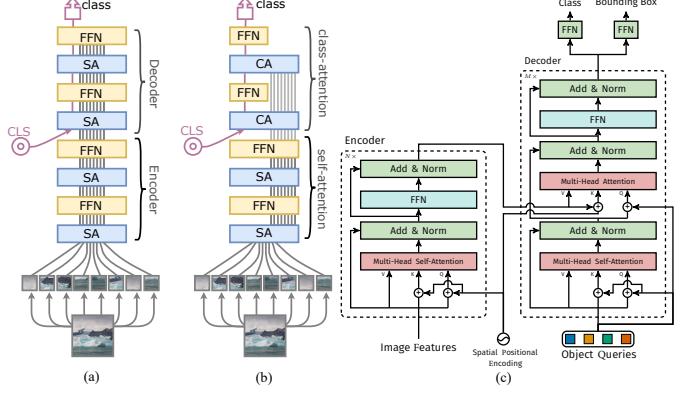


Fig. 9. The structure comparison between CaiT and DETR. (a) The architecture of improved ViT by inserting class token latter. (b) Illustration of CaiT structure with class-attention mechanism. (c) The pipeline of DETR with encoder-decoder architecture. (from [28], [40].)

of ViT [27], iGPT directly resizes the image to a lower resolution, flattens it to a 1D pixel sequence, and then inputs the resulting sequence into a GPT-2 [4] for auto-regressive pixel prediction task. iGPT is able to directly model pixel-level information and achieves a moderate accuracy on low-resolution datasets, but it requires a considerable amount of computation (roughly 2500 V100-days¹ for pretraining). Nevertheless, the generative visual Transformer is still a promising way deserving further investigation.

BEiT: Rather than generating raw pixels directly, Bao et al. propose a BERT-style [5] visual Transformer (BEiT) [63] by reconstructing the masked image in latent space. Similar to the dictionary in BERT, an image tokenizer based on discrete variational autoencoder (dVAE) [107] vectorizes the image into discrete visual tokens. These tokens serve as a set of pseudo labels for pre-training. Then, an image with randomly masked patches is fed into a Transformer backbone. Each output-masked embedding aims to recover its corresponding visual token by maximizing the posterior probability log-likelihood during the pre-training process. In this way, BEiT avoids to learn the redundant pixel-wise representations and outperforms the latest self-supervised approaches (DINO [65] with 0.4%) as well as typical supervised methods (DeiT-B [38] with 1.4%), achieving 83.2% top-1 accuracy on ImageNet-1K.

MoCo v3: For discriminative tasks, Chen et al. [64] go back to basics and investigate the effects of several fundamental components (e.g., batch size, learning rate, and normalization) for self-supervised ViT training. By monitoring ViT's accuracy curves densely, they observe that it is “apparently good”, but suffers a partial sharp drop (termed as dip) during the training process, which mildly affects the eventual performance. To trace the unstable issue, they monitor the gradient magnitude and find that the sudden gradient changes in the first layer (patch projection) will delay a couple of iterations and cause the accuracy dip eventually. Therefore, they propose MoCo v3, a series of fundamental improvements (e.g., freezing the patch projection layer, BatchNorm, and small patch size) for ViT, which surpasses ResNets, especially on large models.

¹<https://openai.com/blog/image-gpt/>

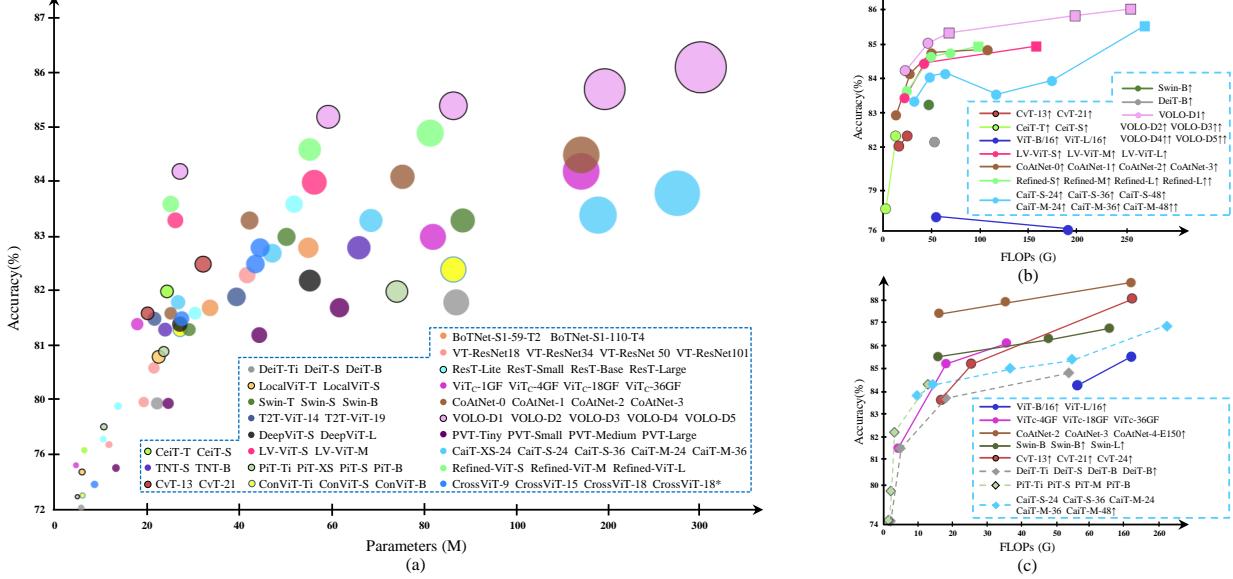


Fig. 10. Comparisons of recent visual Transformers on ImageNet-1k benchmark, including ViT [27], DeiT [38], BoTNet [44], VTs [43], ConViT [45], CeIT [46], LocalViT [47], TNT [52], Swin [33], PiT [57], T2T-ViT [56], PVT [39], CvT [34], DeepViT [59], CaiT [40], Cross ViT [108] (best viewed in color). (a) The bubble plot of the mentioned models with 224^2 resolution input, the size of cycle denotes GFLOPs. (b) Comparison of visual Transformers with high-resolution inputs, the square indicates 448^2 input resolution. (c) The accuracy plot of some pre-trained models on ImageNet-21k.

DINO: Caron et al. [38] demonstrate the effectiveness of distillation for supervised Transformer. Recently, they further extend this teacher-student recipe to self-supervised learning and propose DINO [65]. The core concepts of DINO can be summarized into three points. A momentum encoder with additional centering and sharpening layers serves as a teacher model that outputs centered pseudo labels with an average representation over the batch, as inherited from their early SwAV [109]. An online encoder without the prediction head serves as a student model to fit the teacher’s output. A standard cross-entropy loss connects self-training and knowledge distillation. DINO reaches 80.1% top-1 accuracy on ImageNet in linear evaluation. More interestingly, self-supervised ViT can learn flourishing features for segmentation, which are normally unattainable by supervised Transformer or CNN.

H. Discussion

1) Experimental Evaluation and Comparative Analysis:

Based on our taxonomy, the reviewed supervised models can be grouped into six categories. Table I summarizes the remarkable Transformers on the mainstream classification benchmarks, with a dedicated color assigned to each model. To evaluate them objectively and intuitively, the following three figures illustrate their comparison on ImageNet-1k under different configurations (e.g., model size, FLOPs, and training schemes). Fig. 10(a) summarizes the performance of each model under 224^2 input resolution. Since the FLOPs in Transformer grows quadratically with the input size, Fig. 10(b) takes the FLOPs as a horizontal axis and focuses on their performance under higher-resolution. Fig. 10(c) focuses on the pre-trained models with external datasets. According to these figures, we briefly analyze several improvements beneficial to model performance, efficiency, and scalability as follows:

- In general, most structure-improved methods are optimized for certain model-sizes, problems or specific input resolution, while the other basic training strategies, such as DeiT [38] and LV-ViT [41], are more universal for various models, tasks, and inputs.
- The locality is indispensable for Transformer, which is reflected by the dominant of VOLO [42] and Swin [33] in classification and dense prediction tasks, respectively.
- The convolutional patchify stem (ViT_c [50]) and early convolutional stage (CoAtNet [37]) significantly boost Transformers’ accuracy, especially on large models, because such combinations can give mutual help for capturing fine-grained local features for the shallow layers.
- The deep Transformer is great potential, such as Refined-ViT [35] and CaiT [40]. As the model size grows quadratically with channel dimension, the trade-off between them in deep Transformer is worthy of further investigation.
- CeIT [46] and CvT [34] show significant advantages in training a small or medium model (0–40M), which suggests that such kinds of hybrid attention blocks for lightweight models are worth further exploring.

2) Overview of Development Trend on Visual Transformers:

Transformer backbones sprang up within the last year. When our systematics matches the timeline of these models, we can clearly trace the development tendency of Transformer for image classification (Fig. 1). As a type of self-attention mechanism, visual Transformers are mainly redesigned according to either the vanilla structure in NLP (ViT [27] and iGPT [61]) or attention-based model in CV (VTs [43] and BoTNet [44]).

Then, many approaches start to extend the hierarchical or deep structure of CNN to visual Transformer. T2T-ViT [56], PVT [39], CvT [34] and PiT [57] share a motivation that transferring the hierarchical structure into Transformer but

TABLE I

TOP-1 ACCURACY COMPARISON OF VISUAL TRANSFORMERS ON IMAGENET-1K, CIFAR-10 AND CIFAR-100. “1K ONLY” DENOTES TRAINING ON IMAGENET-1K ONLY; “21K PRE-TRAIN” DENOTES PRE-TRAINING ON IMAGENET-21K AND FINETUNING ON IMAGENET-1K; “DISTILL.” DENOTES APPLYING DISTILLATION TRAINING SCHEME OF DEiT [38], AND “TYPES” REFER TO THE PERSPECTIVE REVIEWED IN THIS PAPER CORRESPONDING TO THE FIG. 4; THE COLOR OF “LEGEND” CORRESPONDING TO EACH MODEL ALSO DENOTES SAME MODEL IN FIG. 10.

Method	Type	Epochs	Batch Size	#Params. (M)	FLOPs (G)	Training Scheme	Image Size		ImageNet-1k Top-1 Acc.		CIFAR Top-1 Acc.		Legend
							Train	Test	1k only	21k pre-train. / Distill.	CIFAR 10	CIFAR 100	
ViT-B/16↑ [27]	OVT	300	4096	86	743	ViT [27]	224	384	77.9	83.97	98.1	87.1	●
ViT-L/16↑ [27]				307	5172		224	384	76.5	85.15	97.9	86.4	
VT-ResNet18 [43]				11.7	1.569		224	224	76.8	-	-	-	
VT-ResNet34 [43]	TEC	90	256	19.2	3.236		224	224	79.9	-	-	-	●
VT-ResNet50 [43]				21.4	3.412		224	224	80.6	-	-	-	
VT-ResNet101 [43]				41.5	7.129		224	224	82.3	-	-	-	
BoTNet-S1-59-T2 [44]				33.5	7.3		224	224	81.7	-	-	-	
BoTNet-S1-110-T4 [44]	TEC	350	4096	54.7	10.9		224	224	82.8	-	-	-	●
BoTNet-S1-128-T5↑ [44]				75.1	19.3		224	256	83.5	-	-	-	
DeiT-Ti [38]				5.7	1.3		224	224	72.2	74.5Υ	-	-	
DeiT-S [38]	CET	300	1024	22.1	4.6	DeiT [38]	224	224	79.8	81.2Υ	-	-	●
DeiT-B [38]				86.6	17.6		224	224	81.8	83.4Υ	99.1	90.8	
DeiT-B↑ [38]				86.6	52.8		224	384	83.1	84.5Υ	99.2	91.4	
ConViT-Ti [45]				6	1		224	224	73.1	-	-	-	
ConViT-S [45]	CET	300	512	27	5.4	DeiT [38]	224	224	81.3	-	-	-	●
ConViT-B [45]				86	17		224	224	82.4	-	-	-	
LocalViT-T [47]	CET	300	1024	5.9	1.3	DeiT [38]	224	224	74.8	-	-	-	●
LocalViT-S [47]				22.4	4.6		224	224	80.8	-	-	-	
CeiT-T [46]				6.4	1.2		224	224	76.4	-	98.5	88.4	
CeiT-S [46]	CET	300	1024	24.2	4.5	DeiT [38]	224	224	82	-	99	90.8	●
CeiT-T↑ [46]				6.4	3.6		224	384	78.8	-	98.5	88	
CeiT-S↑ [46]				24.2	12.9		224	384	83.3	-	99.1	90.8	
ResT-Small [49]				13.66	1.9		224	224	79.6	-	-	-	
ResT-Base [49]	CET	300	2048	30.28	4.3	DeiT [38]	224	224	81.6	-	-	-	●
ResT-Large [49]				51.63	7.9		224	224	83.6	-	-	-	
ViTC-1GF [55]				2048	4.6		224	224	75.3	-	-	-	
ViTC-4GF [55]	CET	400	2048	17.8	4	DeiT [38], PVT [39]	224	224	81.4	-	-	-	●
ViTC-18GF [55]				1024	81.6		224	224	83	84.9	-	-	
ViTC-36GF [55]				512	167.8		224	224	84.2	85.8	-	-	
CoAtNet-0 [37]				25	4.2		224	224	81.6	-	-	-	
CoAtNet-1 [37]				42	8.4		224	224	83.3	-	-	-	
CoAtNet-2 [37]	CET	300/90	4096	75	15.7		224	224	84.1	87.1	-	-	●
CoAtNet-3 [37]				168	34.7		224	224	84.5	87.6	-	-	
CoAtNet-4-E150↑ [37]				275	189.5		224	384	-	88.4	-	-	
TNT-S [52]				23.8	5.2		224	224	81.3	-	-	-	
TNT-B [52]	TET	300	1024	65.6	14.1	DeiT [38]	224	224	82.8	-	-	-	●
TNT-S↑ [52]				23.8	-		224	384	83.1	-	98.7	90.1	
TNT-B↑ [52]				65.6	-		224	384	83.9	-	99.1	91.1	
Swin-T [33]				29	4.5		224	224	81.3	-	-	-	
Swin-S [33]	TET	300/60	1024/4096	50	8.7	DeiT [38]	224	224	83	-	-	-	●
Swin-B [33]				88	15.4		224	224	83.3	85.2	-	-	
Swin-B↑ [33]				88	47		224	384	84.2	86.0	-	-	
Swin-L↑ [33]				197	103.9		224	384	-	86.4	-	-	
VOLO-D1 [42]				27	6.8		224	224	84.2	-	-	-	
VOLO-D2 [42]				59	14.1		224	224	85.2	-	-	-	
VOLO-D3 [42]				86	20.6		224	224	85.4	-	-	-	
VOLO-D4 [42]	TET	300	1024	193	43.8	LV-ViT [41]	224	224	85.7	-	-	-	●
VOLO-D5 [42]				296	69		224	224	86.1	-	-	-	
VOLO-D3↑ [42]				86	67.9		224	448	86.3	-	-	-	
VOLO-D4↑ [42]				193	197		224	448	86.8	-	-	-	
VOLO-D5↑ [42]				296	304		224	448	87	-	-	-	
T2T-ViT-14 [56]	TET	310	1024	21.5	5.2		224	224	81.5	-	97.5	88.4	●
T2T-ViT-19 [56]				39.2	8.9		224	224	81.9	-	98.3	89	
PVT-Tiny [39]				13.2	1.9		224	224	75.1	-	-	-	
PVT-Small [39]	HT	300	128	24.5	3.8	DeiT [38]	224	224	79.8	-	-	-	●
PVT-Medium [39]				44.1	6.7		224	224	81.2	-	-	-	
PVT-Large [39]				61.4	9.8		224	224	81.7	-	-	-	
PiT-Ti [57]				4.9	0.7		224	224	73	74.6Υ	-	-	
PiT-XS [57]	HT	300	1024	10.6	1.4	DeiT [38]	224	224	78.1	79.1Υ	-	-	●
PiT-S [57]				23.5	2.9		224	224	80.9	81.9Υ	-	-	
PiT-B [57]				73.8	12.5		224	224	82	84Υ	-	-	
CvT-13 [34]				20	4.5		224	224	81.6	-	-	-	
CvT-21 [34]				32	7.1		224	224	82.5	-	-	-	
CvT-13↑ [34]	HT	300	2048	20	16.3	ViT [27], BiT [110]	224	384	83	83.3	-	-	●
CvT-21↑ [34]				32	24.9		224	384	83.3	84.9	-	-	
CvT-W24↑ [34]				277	193.2		224	384	-	87.7	-	-	
DeepViT-S [59]	DT	300	256	27	6.2	DeiT [38], ResNest [111]	224	224	82.3	-	-	-	●
DeepViT-L [59]				55	12.5		224	224	83.1	-	-	-	
CaiT-XS-24 [40]				26.6	5.4		224	224	81.8	82.0Υ	-	-	
CaiT-S-24 [40]	DT	400	1024	46.9	9.4	DeiT [38]	224	224	82.7	83.5Υ	-	-	
CaiT-S-36 [40]				68.2	13.9		224	224	83.3	84Υ	99.2	92.2	●
CaiT-M-24 [40]				185.9	36		224	224	83.4	84.7Υ	-	-	
CaiT-M-36 [40]				270.9	53.7		224	224	83.8	85.1Υ	99.3	93.3	
DiversePatch-S12 [60]				22	-		224	224	81.2	-	-	-	
DiversePatch-S24 [60]	DT	400	1024	44	-	DeiT [38]	224	224	82.2	-	-	-	
DiversePatch-B12 [60]				86	-		224	224	82.9	-	-	-	
DiversePatch-B24 [60]				172	-		224	224	83.3	-	-	-	
DiversePatch-B12↑ [60]				86	-		224	384	84.2	-	-	-	
Refined-ViT-S [35]				25	7.2		224	224	83.6	-	-	-	
Refined-ViT-M [35]	DT	300	256	55	13.5	DeiT [38]	224	224	84.6	-	-	-	
Refined-ViT-L [35]				81	19.1		224	224	84.9	-	-	-	●
Refined-ViT-M↑ [35]				55	49.2		224	384	85.6	-	-	-	
Refined-ViT-L↑ [35]				512	81		224	384	85.7	-	-	-	
CrossViT-9 [108]				8.6	1.8		224	224	73.9	-	-	-	
CrossViT-15 [108]	M	300	4096	27.4	5.8		224	224	81.5	-	99	90.77	
CrossViT-18 [108]				43.3	9.03	DeiT [38]	224	224	82.5	-	99.11	91.36	●
CrossViT-18* [108]				44.3	9.5		224	224	82.8	-	-	-	
CrossViT-15† [108]				28.5	21.4		224	384	83.5	-	-	-	
CrossViT-18† [108]				44.6	32.4		224	384	83.9	-	-	-	
LV-ViT-S [41]				26	6.6		224	224	83.3	-	-	-	
LV-ViT-M [41]	DAT	300	1024	56	16	LV-ViT [41]	224	224	84	-	-	-	
LV-ViT-L [41]				150	59		288	288	85.3	-	-	-	●
LV-ViT-M† [41]				56	42.2		224	384	85.4	-	-	-	
LV-ViT-L↑ [41]				150	157.2		288	448	85.9	-	-	-	

they perform downsampling differently. CaiT [40], Diverse Patch [60], DeepViT [59], and Refiner [35] focus on the problem in deep Transformer. Moreover, some approaches move on to the internal components to further enhance the image processing capability in previous Transformers, i.e., positional encoding [48], [112], [113], MHSA [26], and MLP [94].

The next wave of Transformers is locality paradigm. Most of them introduce locality into Transformer via introducing local attention mechanism [33], [42], [52], [53] or convolution [45]–[47]. Nowadays, the most recent supervised Transformers are exploring both the structural combination [37], [50] and scaling laws [36], [114]. In addition to supervised Transformers, self-supervised learning accounts for a substantial part of vision Transformers [61]–[66]. However, it is unclear what tasks and structures are more beneficial to self-supervised Transformer in CV.

3) *Brief Discussion on Alternatives*: During the development of visual Transformers, the most common question is whether Transformer can replace convolution. By reviewing the improvement history in the last year, there is no sign of alternative inferiority here. The visual Transformer has returned from a pure structure to a hybrid form, and global information has gradually returned to a mixed stage with local information. Although Transformer can be equivalent to convolution or even has a better modeling ability than convolution, such a simple and effective convolution operation is enough to process the locality and low-level semantic features in the shallow layer. In the future, the spirit of combining both of them shall drive more breakthroughs for image classification.

IV. TRANSFORMER FOR DETECTION

In this section, we detail visual Transformers for object detection, which can be grouped into two categories: *Transformer as the neck* and *Transformer as the backbone*. Neck detectors are mainly based on a new representation specified to the Transformer structure, called object query that a set of learned parameters equally aggregate global features. They try to solve an optimal fusion paradigm in terms of either convergence acceleration or performance improvement. Besides various necks particularly designed for detection tasks, a proportion of backbone detectors also take specific strategies in consideration. In the end, we compare their performance in Table II and Table III, and then analyze some potential improvements of Transformer detectors.

A. Transformer Neck

We first review DETR [28], an *original Transformer detector* that provides a new representation object query, formulates object detection as a set prediction problem. Due to its low accuracy on small objects and slow convergence, there are many efforts to improve such Transformer detector from three aspects: *sparse attention*, *spatial prior* and *structural redesign*. Additionally, we review the self-supervised application [74].

1) *The Original Detector*: **D**Etection with **T**Ransformer (**DETR**) [28] is the first end-to-end Transformer detector that eliminates hand-designed representations [115]–[118] and

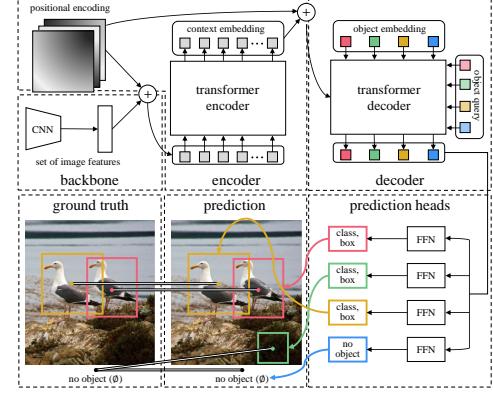


Fig. 11. An overview of DETR. (Modified from [28].)

non-maximum suppression (NMS) post-processing and directly detects all objects by introducing object query and set prediction. Specifically, DETR uses an encoder-decoder Transformer as the neck and a FFN as the prediction head (Fig. 11). The input is extracted by a CNN backbone, flattened into a 1D sequence, attached with positional encodings, and then fed into the Transformer encoder. A small set of learnable positional encodings called object queries are appended to zero inputs and then fed into the Transformer decoder parallelly. In the decoder, a self-attention block processes decoder embedding's relationship, and a cross-attention block aggregates global features into embeddings. Then, a prediction head directly converts decoder outputs into box coordinates and a class score for each object via a simple three-layer FFN. In addition, the class label contains k -class object label and a special class: no object label (\emptyset). During training process, a bipartite matching loss $\mathcal{L}_{\text{match}}$ is applied between prediction $\hat{y}_{\sigma(i)}$ and ground-truth objects y_i to identify one-to-one label assignment $\hat{\sigma}$ as

$$\begin{aligned} \hat{\sigma} = \operatorname{argmin}_{\sigma \in \mathfrak{S}_N} \sum_i^N \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}), \\ \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) = -\mathbb{1}_{\{c_i \neq \emptyset\}} \hat{p}_{\sigma(i)}(c_i) \\ + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}). \end{aligned} \quad (7)$$

Such label assignment bears a resemblance to the method of learning to match anchors (AnchorFree) [119]. In back propagation, the Hungarian loss \mathcal{L}_H includes a negative log-likelihood loss for all label predictions ($i = 1 \dots N$) and a box loss for all matched pairs ($c_i \neq \emptyset$) as

$$\mathcal{L}_H(y_i, \hat{y}_{\sigma(i)}) = \sum_{i=1}^N [-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})]. \quad (8)$$

Overall, DETR provides a new way for the end-to-end object detection. The object query gradually learns instance features during the interaction with image features. The bipartite matching allows a direct set prediction to easily adapt to one-to-one label assignment task, and thus eliminating traditional post-processing. DETR achieves competitive performance on the COCO benchmark but suffers from slow convergence as well as poor performance on small objects.

2) *Transformer with Sparse Attention*: In DETR, the dense interaction between decoder embeddings and global features

costs enormous computation resources and slows down the convergence of DETR. Therefore, some efforts aim to design data-dependent sparse attention to address this problem, such as Deformable DETR [67] and ACT [?].

Deformable DETR: Following their previous work [120], Zhu et al. propose Deformable DETR [67] containing a learnable sparse deformable attention for convergence acceleration and a multi-scale architecture for accuracy improvement. Compared with the original DETR, Deformable DETR achieves higher accuracy (especially on small objects) with $10\times$ less training epochs. Inspired by FPN [121], it cooperates multi-scale features hence the term Multi-Scale Deformable Attention (MSDA) to replace with partial attention blocks, as illustrated in Fig. 12. Given L feature maps $X^l \in \mathbb{R}^{H_l \times W_l \times C}$ and a query sequence $\mathbf{z} \in \mathbb{R}^{N_q \times C}$, MSDA samples offsets $\Delta\mathbf{p} \in \mathbb{R}^2$ of each query for N_k sample keys at each layer's head via two linear layers. While it is sampling the features of key points $V_i \in \mathbb{R}^{L \times N_k \times C_v}$, a linear projection layer is applied to the query to generate an attention map $A_i \in \mathbb{R}^{N_q \times L \times N_k}$ for key samples, where N_q and C are the query's length and dimension, respectively. The process is formulated as

$$\begin{aligned} A_{qik}^l &= \mathbf{z}_q W_{ilk}^A, \quad V_{ik}^l = X^l(\phi_l(\hat{\mathbf{p}}_q) + \Delta\mathbf{p}_{ilqk})W_i^V, \\ \text{MSDAttn}(A_{qik}^l, V_{ik}^l) &= \sum_{i=1}^h (\sum_{l=1}^L \sum_{k=1}^{N_k} A_{qik}^l V_{ik}^l) W_i^O, \end{aligned} \quad (9)$$

where m denotes the attention head, $W_{il}^A \in \mathbb{R}^{C \times N_k}$, $W_i^V \in \mathbb{R}^{C \times C_v}$ and $W_i^O \in \mathbb{R}^{C_v \times C}$ are linear matrices. $\hat{\mathbf{p}}_q \in [0, 1]^2$ is normalized coordinates of each query. After all, MSDA reduces the complexity of computing to $O(2N_qC^2 + \min(HWC^2, N_qKC^2))$ and promotes $1.6\times$ faster inference speed.

ACT: By visualizing the attention map of DETR [28], Zheng et al. observe that the semantically similar and spatially close elements always have a similar attention map in the encoder [68]. To eliminate redundant queries of the encoder, they propose an Adaptive Clustering Transformer (ACT). Based on multi-round Exact Euclidean Locality Sensitivity Hashing (E2LSH) [122], ACT can dynamically cluster queries into different prototypes which are then used to approximate a query-key attention map by broadcasting each prototype to their corresponding queries. Compared with DETR, ACT decreases by 15 GFLOPs and only losses 0.7% AP.

3) *Transformer with Spatial Prior:* Unlike anchor or other representations generated by content and geometry features directly [115], [123], object query implicitly models the spatial information with random initialization, which may be weakly related with the bounding box. Recently, the main methods of the spatial prior application are the one-stage detector with empirical spatial information [69], [70] and the two-stage detector with geometric coordinates initialization [67], [71].

SMCA: To enhance the relation of object query and bounding box with empirical spatial prior, Gao et al. propose a one-stage scheme called Spatially Modulated Cross-Attention (SMCA) [69] to estimate each object queries' spatial prior explicitly. Specifically, SMCA dynamically predicts the initial center and scale of the box corresponding to each object query, generates a Gaussian-like weight map, and then multiplies with the corresponding cross-attention map. Furthermore, SMCA

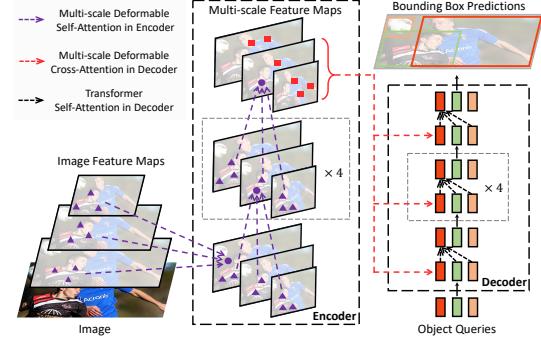


Fig. 12. Illustration of Deformable DETR. A fixed number of key samples in each scale feature interacting with all queries. (from [67]).

leverages the multi-scale feature to augment DETR. In the encoder, an intra-scale self-attention processes feature pixels within each scale, and a multi-scale self-attention equally aggregates visual features at all spatial locations of different scales. For the decoder, the scale-selection attention weights generated by each object query are automatically applied to select scale features for each box. In this way, SMCA obtains an impressive performance (45.6% mAP) on COCO datasets with $5\times$ fewer training epochs than DETR.

Conditional DETR: In the DETR decoder's cross-attention, a spatial attention map between object queries and encoder's positional encodings can be isolated from the query-key attention map. By comparing the spatial attention map in different training epochs, Meng et al. find that the extreme region has larger deviations in shorter-trained models. Given this observation, they propose a new spatial prior, called conditional spatial embedding [70], to explicitly represent the extreme region of an object, thus narrowing down the spatial range for localizing distinct regions. The conditional spatial embedding is learned from reference points and decoder embeddings for cross-attention, where reference points serve as either learned parameters replacing with object queries or the output prediction from object queries. Unlike the original prediction head of DETR, Conditional DETR uses reference points as additional inputs to localize the candidate box. These two improvements enhance DETR with $8\times$ faster convergence time and by 1.8% box mAP on COCO.

Two-Stage Deformable DETR: In [67], Zhe et al. also empower a two-stage scheme to augment spatial prior. Instead of learned object query, it generates Top-K region proposals from encoder outputs and inputs them as substitutes into Transformer decoder for further refinement. Similar to previous two-stage detectors [115], it significantly improves the accuracy of Deformable DETR but increases training costs slightly.

Efficient DETR: In Deformable DETR [67], object queries can be visualized as a series of reference points. Interestingly, the different initialized points always tend to distribute similarly after multiple training iterations. Given this observation, Yao et al. propose a two-stage Efficient DETR [71], consisting of dense proposal generation and sparse set prediction parts. Its overall structure is similar to Two-Stage Deformable DETR [67], but both dense and sparse parts share the same detection head. With a single decoder layer, Efficient DETR

TABLE II

COMPARISON BETWEEN TRANSFORMER NECKS AND REPRESENTATIVE CNNs ON COCO 2017 VAL SET. “GPUs” DENOTES THE TRAINING TIME WITH THEIR GPU ENVIRONMENTS; “TYPE” REFERS TO THE PERSPECTIVE REVIEWED IN THIS PAPER; “MULTI-SCALE” REFERS TO MULTI-SCALE INPUTS

Method	Backbone	Type	Training Scheme	GPUs	Epochs	FLOPs (G)	#Params (M)	FPS	Multi Scale	AP	AP ₅₀	AP ₇₅	Ap _S	Ap _M	Ap _L
<i>CNN Backbone with Other Representations</i>															
FCOS [72], [123]	ResNet50	-	-	-	36	177	-	17	✓	41.0	59.8	44.1	26.2	44.6	52.2
Faster R-CNN [115]		-	-	-	36	180	42	26	✓	40.2	61.0	43.8	24.2	43.5	52
Faster R-CNN+ [115]		-	-	-	108	180	42	26	✓	42	62.1	45.5	26.6	45.4	53.4
Mask R-CNN [124]		-	-	-	36	260	44	-	✓	41	61.7	44.9	-	-	-
Cascade Mask R-CNN [125]		-	-	-	36	739	82	18	✓	46.3	64.3	50.5	-	-	-
<i>Transformer Model as Neck</i>															
DETR-R50 [28]	ResNet50	OTD	DETR	8 V100*240h 8 V100*240h	500 500	86 187	41 41	28 12	-	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5-R50 [28]		SA	DETR	Without Training Without Training	- -	156 169	- -	14 16	-	43.3	63.1	45.9	22.5	47.3	61.1
ACT-MTKD (L=16) [68]	ReNet50	SA	DETR	Without Training Without Training	- -	156 169	- -	14 16	-	40.6	-	-	18.5	44.3	59.7
ACT-MTKD (L=32) [68]		SA	DETR	8 V100*20h 8 V100*27h 8 V100*40h 8 V100*43h	50 50 50 50	78 128 173 173	34 34 40 40	27 22 19 19	-	39.7	60.1	42.4	21.2	44.3	56.0
Deformable DETR [67]	ReNet50	SA	DETR	8 V100*33h 8 V100*70h 8 V100*75h 8 V100*160h	50 108 50 108	152 152 152 152	40 40 40 40	22 22 10 10	-	41.0	-	-	21.9	44.3	59.1
Deformable DETR-DC5 [67]		SA+SP	DETR	8 V100*33h 8 V100*70h 8 V100*75h 8 V100*160h	50 108 50 108	152 152 152 152	40 40 40 40	22 22 10 10	-	42.7	-	-	22.8	46.1	60.0
Deformable DETR [67]	ResNet50	SP	DETR	8 V100*33h 8 V100*70h 8 V100*75h 8 V100*160h	50 108 50 108	152 152 152 152	40 40 40 40	22 22 10 10	✓	43.7	63.6	47.2	24.2	47.0	60.4
Two-Stage Deformable DETR [67]		SP	DETR	8 V100*33h 8 V100*70h 8 V100*75h 8 V100*160h	50 108 50 108	152 152 152 152	40 40 40 40	22 22 10 10	✓	45.6	65.5	49.1	25.9	49.3	62.6
Efficient DETR [71]	ReNet50	SP	DETR	-	36 36	159 210	32 35	-	✓	44.2	62.2	48	28.4	47.5	56.6
Efficient DETR* [71]		SP	DETR	-	36 36	159 210	32 35	-	✓	45.1	63.1	49.1	28.3	48.4	59.0
Conditional DETR [70]	ReNet50	SP	DETR	8 V100*30h 8 V100*30h	108 108	90 195	44 44	-	-	43.0	64.0	45.7	22.7	46.7	61.5
Conditional DFETR-DC5 [70]		SP	DETR	8 V100*30h 8 V100*30h	108 108	90 195	44 44	-	-	45.1	65.4	48.5	25.3	49	62.2
UP-DETR [74]	ReNet50	VA	DETR	8 V100*72h 8 V100*144h	150 300	86 86	41 41	28 28	-	40.5	60.8	42.6	19.0	44.4	60.0
UP-DETR+ [74]		VA	DETR	8 V100*72h 8 V100*144h	150 300	86 86	41 41	28 28	-	42.8	63.0	45.3	20.8	47.1	61.7
TSP-FCOS [72]	ReNet50	ES	Detreron2	8 V100*15h 8 V100*15h 8 V100*40h	36 36 96	189 188 188	51.5 64 64	15 11 11	✓	43.1	62.3	47	26.6	46.8	55.9
TSP-RCNN [72]		ES	Detreron2	8 V100*15h 8 V100*15h 8 V100*40h	36 36 96	189 188 188	51.5 64 64	15 11 11	✓	43.8	63.3	48.3	28.6	46.9	55.7
TSP-RCNN+ [72]		ES	Detreron2	8 V100*15h 8 V100*15h 8 V100*40h	36 36 96	189 188 188	51.5 64 64	15 11 11	✓	45	64.5	49.6	29.7	47.7	58.0
YOLOS-S [73]	DeiT-S	ES	DeiT, DETR	8 2080Ti*240h	150	200	30.7	7	-	36.1	56.4	37.1	15.3	38.5	56.1
YOLOS-S [73]		DeiT-B	DeiT, DETR	8 2080Ti*480h	150	179	27.9	5	✓	37.6	57.6	39.2	15.9	40.2	57.3
YOLOS-B [73]	DeiT-B	ES	DeiT, DETR	8 2080Ti*480h	150	537	127	-	-	42.0	62.2	44.5	19.5	45.3	62.1

achieves a comparable results of DETR with $14\times$ fewer training epochs. More interestingly, it is observed that two stacking decoder layers brings slight improvement, but more stacks yield even worse results. How to improve the deeper stacking decoder for Efficient DETR deserves further investigation.

4) *Transformer with Redesigned Structure*: Besides the modifications focusing on cross-attention, some works redesign an encoder-only structure to avoid decoder’s problem directly. For example, TSP [72] inherits the idea of set prediction [28] and yields decoder and object query. YOLOS [73] combines the encoder-decoder neck of DETR and encoder-only backbone of ViT to redesign an encoder-only detector.

TSP: By monitoring the sparsity (negative entropy) of the attention map in each Transformer layer, Sun et al. [72] indicate that the long training process is mainly caused by the slow convergence of cross-attention, and present an encoder-only DETR with previous representations [115], [123], called TSP-FCOS and TSP-RCNN, to accelerate model convergence. Specifically, they generate a set of fixed-size Features of Interests (FoI) [123] or proposals [115] that are subsequently fed into input Transformer encoder. In addition, a matching distillation is applied to resolve the instability of the bipartite matching, especially in the early training stage. Compared with the original DETR, TSP achieves a better performance with $5-10\times$ less training cost and $1.5-2\times$ faster inference speed.

YOLOS: Inspired by ViT [27] and DETR [28], Fang et al. propose YOLOS [73], a pure sequence-to-sequence Transformer to unify classification and detection tasks. It inherits ViT’s structure and replaces the single class token with fixed

size learned detection tokens, similar to the input of DETR decoder. These object tokens and patches are concatenated into one sequence and inputs into Transformer encoder. It first pre-trains the transfer ability of object tokens in classification tasks and then fine-tunes the pre-trained object tokens on detection benchmark. YOLOS not only improves DETR performance but also provides a generalized framework for visual tasks.

5) *Transformer Detector with Self-Supervised Learning*: Inspired by the successful pre-trained NLP Transformer [3], [5], Dai et al. propose an Unsupervised Pre-training **DETR (UP-DETR)** [74] to assist supervised training from three aspects. 1) One single randomly cropped patch from the given image is assigned to all object queries. The objective of the decoder is to localize the patch position. 2) To avoid over-bias towards localization in pre-training, an auxiliary reconstruction task is proposed to preserve the feature discrimination. 3) Based on the single-query patch, multi-query localization assigns multiple patches to different object queries to mimic multi-target detection tasks and accelerate convergence. Each patch query is predicted independently via the mask attention and object query shuffle. UP-DETR obtains higher accuracy with faster convergence speed on small datasets than DETR, and even a better performance with sufficient training data.

B. Transformer Backbone

We reviewed numerous Transformer-based backbones for image classification [27], [38] in Section III. These backbones can be easily incorporated into various frameworks (e.g., Mask R-CNN [124], RetinaNet [117], DETR [28], etc.) to perform

TABLE III

DENSE PREDICTION RESULTS OF COCO 2017 VAL. SET BASED ON RETINANET [117] AND MASK R-CNN [124], WHEN TRAINED WITH 3 \times SCHEDULE AND MULTI-SCALE INPUTS (MS). THE NUMBERS BEFORE AND AFTER “/” CORRESPOND TO THE PARAMETER OF RETINANET AND MASK R-CNN, RESPECTIVELY. (MOST OF DATA FROM [54].)

Backbone	#Params (M)	FLOPs (G)	RetinaNet 3 \times schedule + MS						Mask R-CNN 3 \times schedule + MS					
			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 [11]	38 / 44	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
PVTv1-Small [39]	34 / 44	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 [55]	36 / 45	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 [33]	39 / 48	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
PVTv2-B2-Li [58]	32 / 42	- / -	-	-	-	-	-	-	46.8	68.7	51.4	42.3	65.7	45.4
Focal-Tiny [54]	39 / 49	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
PVTv2-B2 [58]	35 / 45	- / -	-	-	-	-	-	-	47.8	69.7	52.6	43.1	66.8	46.7
ResNet101 [11]	57 / 63	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 [126]	56 / 63	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
PVTv1-Medium [39]	54 / 64	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 [55]	51 / 60	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 [33]	60 / 69	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 [54]	62 / 71	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 [126]	96 / 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
PVTv1-Large [39]	71 / 81	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 [55]	67 / 76	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 [33]	98 / 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 [54]	101 / 110	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

dense prediction tasks. In addition to general improvements, proportion of them also benefit to dense prediction tasks. The hierarchical structure, like PVT [39], [58], constructs Transformer as a high-to-low resolution process to learn multi-scale features. The local enhanced structure constructs the backbone as a local-to-global combination to efficiently extract both short- and long-range visual dependencies and avoid the quadratic computational overhead, such as Swin-Transformer [33], ViL [55], and Focal Transformer [54]. Table III compares these models based on different frameworks in dense prediction tasks. Transformer-based backbones outperform the modern CNN models by 2-6.8%, which demonstrates the effectiveness of Transformer for dense prediction.

Similar to FPN [121], Zhang et al. propose a **Feature Pyramid Transformer (FPT)** [75] dedicated to dense prediction tasks, by combining the characteristics of non-local [14] and multi-scale features. It leverages three attention components to model interactions across both space and scales, including self-attention, top-down cross-attention, and bottom-up cross channel attention. FPT serves as a general-purpose backbone for dense prediction tasks attaining further promotion over many SOTA models.

C. Discussion

This section briefly compares and analyzes Transformer detectors according to Table II and III. For Transformer necks, we only focus on their FLOPs within the single-scale feature structure rather than multi-scale features because they apply different number of layers. From the perspective of Sparse Attention (SA), Deformable DETR [67] reduces 8 GFLOPs and accelerates 12 \times training times, while ACT-DC5 [68] decreases computation costs from 187 to 156 GFLOPs, with only slight performance loss. From the perspective of Spatial Prior (SP), the one-stage detector explicitly separates the spatial prior from object query, bringing fast convergence and high accuracy. SMCA [69] and Conditional DETR [70] achieve 42.7% and 43% mAP with 108 training epochs, respectively.

Both the two-stage detectors and TSP-RCNN [72] replace learned object queries with proposals. Such essentially the same but structurally different methods significantly promote detector accuracy. From the perspective of Multi-Scale (MS) features, it can compensate for Transformer performance on small object detection. For example, Deformable DETR [67] and SMCA [69] boost DETR by 5.2% and 3.1% AP_S. The encoder-only structure reduces the number of Transformer layers but increases FLOPs excessively, such as YOLOS-B with 537 GFLOPs. In contrast, the encoder-decoder structure is a good trade-off between GFLOPs and layer number, but deeper decoder layers may cause long training processes and over-smooth problems. Therefore, an integration of SA into the deep decoder with MS and SP deserves further investigation.

There are many backbone improvements for classification but few works attend to dense prediction tasks. Based on the proposed taxonomy in Section III, it is easy to categorize the existing methods into two parts: hierarchical Transformer and locality enhanced Transformer. In the future, we anticipate that Transformer backbone would join with the deep high-resolution network [127] to solve dense prediction tasks.

V. TRANSFORMER FOR SEGMENTATION

Transformer has been widely applied for segmentation in two ways: *patch-based Transformer* and *query-based Transformer*. The latter can be further decomposed into *Transformer with object query* and *Transformer with mask embedding*.

A. Patch-Based Transformer

As the receptive field expansion strategy, CNN needs lots of decoder stacks to map high-level features into original spatial resolution. In contrast, relying on the global modeling capability, patch-based Transformer treats the input image as a patch sequence and fed them into a columnar Transformer encoder. This resolution invariance strategy empowers Transformer to incorporate just a relatively simple decoder

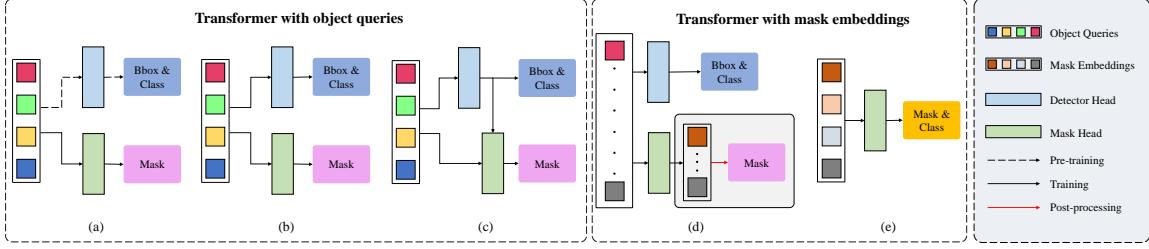


Fig. 13. Query-based frameworks for segmentation tasks. (a) The model is firstly trained on detection task and then fine-tuned on segmentation task following a serial training process. (b) The model is supervised on two dependent task simultaneously. (c) The cascade hybrid task model generates a fine-grained mask based on the coarse region predicted by detector head. (d) The query embeddings are dependently supervised by mask embeddings and boxes. (e) The box-free model directly predicts masks without box branch and views segmentation task as a mask prediction problem.

and obtains desirable performance for segmentation tasks. Furthermore, some works [76]–[78] attempt to investigate an optimal combination between patch-based Transformer and different segmentation frameworks [124], [128].

SETR: Inspired by ViT [27], Zheng et al. extend visual Transformer to semantic segmentation tasks and present SEgmentation TRansformer (SETR) [76]. SETR substitutes CNN backbone with Transformer encoder and follows ViT’s input-output structure excepting the class token. Moreover, it employs three fashions of the decoder to perform per-pixel classification: naive up-sampling (Naive), progressive up-sampling (PUP), and multi-level feature aggregation (MLA). SETR demonstrates great feasibility of Transformer encoder in the segmentation task, but relies on expensive GPU clusters and extra RAMs attributed to numbers of stack layers and quadratic computational costs.

TransUNet: TransUNet [77] is the first visual Transformer for medical image segmentation. It can be viewed as either a variant of SETR with MLA decoder [76], or a hybrid model of U-Net [128] and Transformer. Although this structure is simple in concept, it attains excellent results and proves the effectiveness of visual Transformer in this field.

Segformer: Segformer [78] utilizes a series of simple and practical measures to improve Transformer for semantic segmentation tasks, such as hierarchical structure [39], overlap patch projection [34], efficient attention mechanism [34], [39], and convolutional position embeddings [48], [53]. In contrast to CNN decoder consisting of heavy stacks for expanding receptive field, Segformer redesigns a lightweight decoder with only four MLP layers as the powerful global integration capability of Transformer encoder. Experimentally, Segformer-B5 attains a new SOTA result with 51.8% mIoU on ADE20K with 4× smaller than SETR. Moreover, Segformer shows the stronger robustness than DeepLabv3 when testing on the Cityscapes dataset with multiple types of corruption.

B. Query-Based Transformer

Query is a learnable embedding at the input and output of the Transformer decoder. In comparison with patch embedding, query embedding can more “fairly” integrate the information of each patch. The query-based Transformer with set prediction loss is able to remove other hand-crafted representations and post-processings. Recently, many efforts attempt to generalize this representation to segmentation tasks

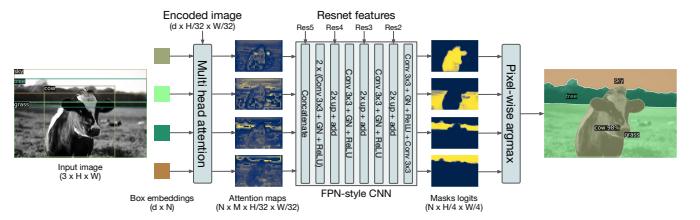


Fig. 14. An overview of mask head in Panoptic DETR. (from [28].)

which can be grouped into two categories. One type of framework is driven by *object query* that is also supervised by detection task. The other type of query is only supervised by segmentation task, referred to as *mask embedding*.

1) *Transformer with Object Queries*: There are three training manners for object query based methods. Based on the pre-trained object query of DETR [28], a mask head with the query is further refined by segmentation task (Fig. 13(a)). Instead of the multi-stage training process, the object query is concurrently modeled by detection and segmentation tasks in some end-to-end frameworks [79] (Fig. 13(b)). Another type [81] attempts to bridge the gap between different task branches building with a hybrid cascaded network, where the box output serves as the input of the mask head (Fig. 13(c)).

Panoptic DETR: Query-based Transformer is firstly introduced by DETR [28] for detection tasks. It also extends this form to panoptic segmentation by training a mask head based on the pre-trained object query. In detail, a cross-attention block between object query and encoded feature is applied to generate an attention map for each object. After an upsampling FPN-style CNN, a spatial argmax operation fuses the resulting binary masks to a non-overlapping prediction. Empirically, it achieves 45.1% Panoptic Quality (PQ) [129] on COCO panoptic benchmark.

Cell-DETR: Similar to the expansion from Faster R-CNN [115] to Mask R-CNN [124], Prangemeier et al. leverage DETR [28] to perform end-to-end cell instance segmentation (term Cell-DETR) [79]. It inherits the cross-attention of original mask head in DETR and substitutes the addition operation with a pixel-adaptive convolution [130] for feature fusion. Unlike the serial training process of DETR, it assigns different heads/branches to each task concurrently, where a non-overlapping mask is generated by a U-Net [128] branch, and a class label serves as the output from each query. Cell-

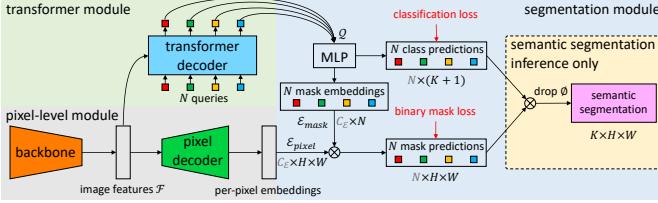


Fig. 15. Illustration of Maskformer. (from [85].)

DETR achieves SOTA on live-cell microscopy dataset.

VisTR: Another extension is Video instance segmentation Transformer (VisTR) [80] that formulates video instance segmentation (VIS) task as a direct parallel sequence prediction problem. The core idea of VisTR is a bipartite matching loss based on instance sequence level to maintain the order of outputs, which can force one-to-one predictions directly. In detail, a 2D CNN backbone extracts features for each frame, and an encoder-decoder Transformer with 3D position encoding is applied to model the similarity of pixel-level and instance-level features. Similar to Cell-DETR [79], two parallel heads are used to predict box and mask sequence for each instance of different frames simultaneously. Experimentally, VisTR achieves 35.3% accuracy on the YouTube VIS dataset, the best performance among single model approaches.

QueryInst: The other extension is QueryInst [81] that enables the hybrid task cascade structure to bridge the gap between two relatively independent branches. The goal of QueryInst is to fore a one-to-one correspondence between mask ROI features and object queries via a series of parallel dynamic mask heads with shared queries. Based on the update strategy of Sparse R-CNN [131] and Cascade Mask R-CNN [132], QueryInst predicts a mask according to the previous stage queries and the current stage bounding box prediction. The dynamic mask heads can be parallelly trained to facilitate the communication between the detection and segmentation tasks. In addition, the query embedding and MSA are shared to encourage the two sub-tasks to benefit each other. QueryInst obtains the latest SOTA results based on Swin-Transformer on COCO instance segmentation benchmarks.

2) *Transformer with Mask Embeddings:* The other framework makes efforts to use query to predict mask directly, and we refer to this learned mask-based query as mask embedding. Unlike object query, mask embedding is only supervised by segmentation tasks. As shown in Fig. 13(d), two disjoint sets of queries are employed parallelly for different tasks. For semantic segmentation and box-free frameworks, some studies remove object query from query-based Transformer and predict mask by mask embedding directly (Fig. 13(e)).

ISTR & SOLQ: To make 1D sequence output supervised by 2D mask label directly, ISTR [82] empowers a mask pre-coding method to encode ground-truth spatial mask into low-dimensional mask embedding for instance segmentation. Similar to ISTR, Dong et al. propose a more simple pipeline SOLQ [83] and explore three reversible compression encoding methods (Sparse Coding, PCA, and DCT) for the mask embedding. In detail, a set of unified queries is applied to parallelly perform multiple representation learning: classification, box

TABLE IV
COMPARISON BETWEEN UPERNET [133] AND TRANSFORMER-BASED MODEL ON ADE20K VAL. SET FOR SEMANTIC SEGMENTATION. ALL MODELS ARE TRAINED WITH 160K ITERATION AND 16 BATCHSIZE.
“+MS” DENOTES THE PERFORMANCE WHEN TRAINED WITH
MULTI-SCALE INPUTS; “PRETRAIN” REFERS TO THE PRETRAINED
DATASET OF BACKBONE; “1K” AND “22K” DENOTE IMAGENET-1K AND
IMAGNET-21K WITH THE FORMER, RESPECTIVELY.

Method	Backbone	Pretrain	image size	#Params. (M)	FLOPs (G)	FPS	mIoU	+MS
Upernet [133]	R-50	1k	512 ²	-	-	23.4	42.1	42.8
	R-101	1k	512 ²	86	1029	20.3	43.8	44.9
	Swin-T	1k	512 ²	60	236	18.5	44.5	46.1
	Swin-S	1k	512 ²	81	259	15.2	47.6	49.3
	Swin-B	22k	640 ²	121	471	8.7	50.0	51.6
	Swin-L	22k	640 ²	234	647	6.2	52.0	53.5
Segformer [78]	MiT-B3	1k	512 ²	47.3	79	-	49.4	50.0
	MiT-B4	1k	512 ²	64.1	95.7	15.4	50.3	51.1
	MiT-B5	1k	512 ²	84.7	183.3	9.8	51.0	51.8
Segmenter [84]	ViT-S/16	1k	512 ²	37	-	34.8	45.3	46.9
	ViT-B/16	1k	512 ²	106	-	24.1	48.5	50.0
	ViT-L/16	22k	640 ²	334	-	-	51.8	53.6
MaskFormer [85]	R-50	1k	512 ²	41	53	24.5	44.5	46.7
	R-101	1k	512 ²	60	73	19.5	45.5	47.2
	Swin-T	1k	512 ²	42	55	22.1	46.7	48.8
	Swin-S	1k	512 ²	63	79	19.6	49.8	51.0
	Swin-B	22k	640 ²	102	195	12.6	52.7	53.9
	Swin-L	22k	640 ²	212	375	7.9	54.1	55.6

regression, and mask encoding. Based on the original detector head of DETR [134], SOLQ adds a mask branch via MLP to produce mask embedding loss. Both ISTR and SOLQ obtain similar results and outperform previous methods even with approximation-based suboptimal embeddings. However, they lead to a huge gap between the performance of AP^{box} and AP^{seg} , which is worthy of further exploration.

Max-DeepLab: Inspired by the end-to-end Transformer detector [28], Wang et al. propose Max-DeepLab [29], the first query-based end-to-end Transformer for panoptic segmentation. Max-DeepLab directly forces a set of unique labeled-mask predictions via a PQ-style bipartite matching loss and a dual-path Transformer structure. Given a set of mask embeddings and an image input, Max-DeepLab processes them separately in both memory and CNN path, and then predicts non-labeled masks and corresponding labels by using FFN and dot-product cross multiplication, respectively. This end-to-end box-free Transformer achieves new SOTA with 51.3% PQ on COCO test-dev set, but leads to heavy computational costs because of its excessive auxiliary losses and dual-path structure processing on high-resolution features.

Segmenter: Rather than processing the 2D lattice and memory sequence in a dual-path structure [29], Strudel et al. present a simple convolution-free model Segmenter [84] that views semantic segmentation task as a sequence-to-sequence problem. In Segmenter, a pre-trained Transformer backbone aggregates the relations of patches, a set of mask embeddings assigned to each class are fed into a Transformer encoder together with the patches, and then a scalar product between embeddings and patches generates a labeled-mask prediction for each patch. As a result, Segmenter surpasses the latest SOTA CNN on mainstream semantic segmentation benchmarks (i.e., ADE20K, Pascal Context datasets and Cityscapes).

Maskformer: Unlike mainstream semantic segmentation methods that predict mask in pixel-level, Cheng et al. view

TABLE V
COMPARISON BETWEEN TRANSFORMERS AND REPRESENTATIVE CNNS ON COCO TEST-DEV FOR INSTANCE SEGMENTATION.

Method	Backbone	Epochs	Ap^m	AP_S^m	Ap_M^m	Ap_L^m	Ap^b	FPS
Mask R-CNN [124]	R-50-FPN	36	37.5	21.1	39.6	48.3	41.3	15.3
	R-101-FPN	36	38.8	21.8	41.4	50.5	43.1	11.8
Blend Mask [135]	R-50-FPN	36	37.8	18.8	40.9	53.6	43	15
	R-101-FPN	36	39.6	22.4	42.2	51.4	44.7	11.5
SOLO v2 [136]	R-50-FPN	36	38.2	16	41.2	55.4	40.7	10.5
	R-101-FPN	36	39.7	17.3	42.9	57.4	42.6	9
ISTR [82]	R-50-FPN	36	38.6	22.1	40.4	50.6	46.8	13.8
	R-101-FPN	36	39.9	22.8	41.9	52.3	48.1	11
SOLQ [83]	R-50	50	39.7	21.5	42.5	53.1	47.8	-
	R-101	50	40.9	22.5	43.8	54.6	48.7	-
	Swin-L	50	45.9	27.8	49.3	60.5	55.4	-
QueryInst [81]	R-50-FPN	36	40.1	23.3	42.1	52	44.8	10.5
	R-50-FPN	36	40.6	23.4	42.5	52.8	45.6	7
	R-101-FPN	36	41.7	24.2	43.9	53.9	47	6.1
	Swin-L	50	49.1	31.5	51.8	63.2	56.1	3.3

semantic segmentation task as a mask prediction problem and enable this output format to query-based Transformer hence the term Maskformer [85]. In detail, a parallel Transformer-CNN decoder separately processes mask embeddings and per-pixel features. Then, the model predicts a set of overlapping binary masks via a dot product with sigmoid activation. At the semantic inference time, a matrix multiplication combines them to generate the final prediction (Fig. 15). MaskFormer not only outperforms the current per-pixel classification SOTA on large-classes semantic segmentation datasets (55.6 mIoU on ADE20K), but also is generalized to panoptic segmentation task with a new SOTA (52.7 PQ on COCO).

C. Discussion

As a fundamental yet still challenging task, segmentation also benefits from evolving visual Transformers. We summarize these Transformers according to three different segmentation sub-tasks. Table IV focuses on the ADE20K validation set (170 classes) for semantic segmentation task. We find that Transformer shows great performance improvements trained on the datasets with large numbers of classes rather than smaller ones. Table V focuses on the COCO test-dev 2017 dataset for instance segmentation task. Clearly, Transformer with mask embedding surpasses the previous prevailing models in both segmentation and detection tasks. These methods significantly improve the accuracy of the box but only have slight improvement for segmentation, thus leading to a huge gap between the performance of AP_{box} and AP_{seg} . Based on the cascaded framework, QueryInst [81] attains the SOTA performance among Transformer models. The combination of Transformer and hybrid task cascade structure is worthy of further research. Table VI focuses on panoptic segmentation task. Max-DeepLab [29] is general to solve both foreground and background in panoptic segmentation tasks via the mask prediction format, while Maskformer [85] successfully employs this format to semantic segmentation and unifies both semantic- and instance-level segmentation tasks. Based on their performance in the panoptic segmentation field, it is concluded that Transformer could unify multiple segmentation tasks into one box-free framework with mask prediction.

TABLE VI
COMPARISON OF THREE REPRESENTATIVE TRANSFORMERS ON COCO PANOPTIC MINIVAL. FOR PANOPTIC SEGMENTATION.

Method	Backbone	Epochs	PQ	PQ^{Th}	PQ^{St}	#Params.	FLOPs
DETR [28]	ResNet-50	150+25	43.4	48.2	36.3	42.8	137
	ResNet-101		45.1	50.5	37	61.8	157
MaxDeepLab [29]	Max-S	54	48.4	53	41.5	61.9	162
	Max-L		51.1	57	42.2	451	1846
MaskFormer [137]	RseNet-50	300	46.5	51	39.8	45	181
	ResNet-101		47.6	52.5	40.3	64	248
MaskFormer [137]	Swin-T	300	47.7	51.7	41.7	42	179
	Swin-S		49.7	54.4	42.6	63	259
	Swin-B		51.1	56.3	43.2	102	411
	Swin-L		52.7	58.5	44	212	792

VI. DISCUSSION AND CONCLUSION

This section briefly conducts this paper with a summary of the reviewed improvements in Sec. VI-A, an overall discussion of critical issues in Sec. VI-B, a suggestion about future work directions in Sec. VI-C, and a final conclusion in Sec. VI-D.

A. Summary of Recent Improvements

Based on the afore-mentioned comparison and discussion, we would now like to provide a brief summary of recent improvements for the three basic tasks as follows.

- For classification, a deep hierarchical Transformer backbone can be valid for decreasing computational complexity [39] and avoiding feature over-smoothing [35], [40], [59], [60] in the deep layer. Meanwhile, the early-stage convolution [37] is enough to capture low-level features, which can significantly enhance the robustness and reduce the computational complexity in the shallow layer. Moreover, both convolutional projection [46], [47] and local attention mechanism [33], [42] can improve the locality of Transformer. The former [48], [49] may also be a new approach to replace with positional encoding.
- For detection, the Transformer necks benefit from the encoder-decoder structure with less computation than the encoder-only Transformer detector [73]. Thus, the decoder is necessary but requires few stacks [71] owing to its slow convergence [72]. Furthermore, sparse attention [67] is conducive to reduce computational complexity and accelerate the convergence of Transformer, while spatial prior [67], [69], [70] benefits to the performance of Transformer with a little fast convergence.
- For segmentation, the encoder-decoder Transformer models may unify three segmentation sub-tasks into a mask prediction problem by a series of learnable mask embeddings [29], [84], [137]. This box-free approach has achieved the latest SOTA on multiple benchmarks [137]. Moreover, the specific hybrid tasks cascaded model [81] of box-based Transformer is demonstrated to obtain a higher performance in the instance segmentation task.

B. Discussion on Visual Transformer

Despite large amounts of visual Transformer models and applications, the “essential” understanding of visual Transformer remains inefficient. Therefore, we will focus on some key issues to help resolve the confusion of readers.

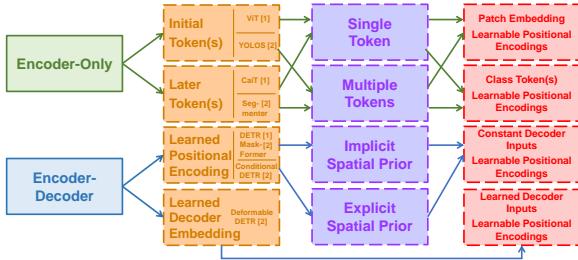


Fig. 16. Taxonomy of the learnable embedding.

1) *How Transformer Bridge The Gap Between Language and Vision:* Transformer is initially designed for machine translation tasks [1]. In the language model, each word of a sentence is taken as a basic unit that represents high-level, high-dimensional semantic information. These words can be embedded into low-dimensional vector space representations hence the term word embeddings. In the visual task, each pixel of an image is low-level, low-dimensional semantic information, which is not matched with the embedding features. Therefore, the key to transferring into visual tasks is *building an image-to-vector transformation and maintaining the image's characteristics*. For example, ViT [27] transforms an image into patch embeddings with multiple low-level information by strong slackness conditions, while Early Conv. [50] and CoAtNet [37] leverage convolution to extract high-level information and reduce the redundant features from patches.

2) *The Relationship Between Transformer, Self-Attention and CNN:* From the perspective of convolution, as mentioned in Sec. III-C, its inductive bias is mainly shown as locality, translation invariance, weight sharing, and sparse connection. Such simple convolutional kernels can perform template matching efficiently but its upper-bound is lower than Transformer because of the strong inductive bias.

From the perspective of self-attention mechanism, as illustrated in Sec. III-B and Sec. III-D, *it can theoretically express any convolutional layer when given a sufficient number of heads* [26]. Such fully-attentional operation can combine local and global level attention alternatively, and generate attention weights dynamically according to the relationship of features. Even so, its practicality is inferior, with lower accuracy and higher computational complexity than SOTA CNN.

From the perspective of Transformer, Dong et al. demonstrate that the self-attention layer manifests strong inductive bias towards “token uniformity” when trained on deep layers without short connection or FFNs [94]. It is concluded that *Transformer consists of two key components: a self-attention layer aggregates the relationship of tokens, and a position-wise FFN extracts the feature of inputs*. Although Transformer has a powerful global modeling capability, as illustrated in Sec. III-C and VI-B1, convolution can effectively process low-level features [37], [50], enhance the locality of Transformer [45], [70], and append positional features via padding [48], [49], [102].

3) *Learnable Embeddings in Different Visual Tasks:* Transformer models employ learnable embeddings to perform different visual tasks. From the supervision task point of view, these embeddings can be categorized into class token, object

query, and mask embedding. From the structural point of view, there is internal relationship between them. The recent Transformer methods mainly adopt two different patterns, encoder-only and encoder-decoder structure. Each structure consists of three levels of embedding applications, as illustrated in Fig. 16. From the position level, the application of learned embedding in the encoder-only Transformer is decomposed into the initial token(s) [27], [73] and later token(s) [40], [84], while learned positional encoding [28], [70], [137] and learned decoder input embedding [67] are applied to the encoder-decoder structure. From the quantity level, encoder-only design applies different number of tokens. For example, the ViT [27], [38] family and YOLOS [73] append different number tokens into the initial layer, while CaiT [40] and Segmenter [84] leverage the ones to represent the last few layers’ features in different tasks. In the encoder-decoder structure, the learned positional encoding of decoder (object query [28], [70] or mask embedding [137]) is attached to the decoder inputs in explicitly [28], [137] or implicitly [69], [70]. Different from the constant inputs, Deformable DETR [67] employs a learned embedding as input and attends to the encoder output.

Inspired by the multi-head attention design, the multiple initial tokens strategy is supposed to further improve the classification performance. However, DeiT [38] indicates that these additional tokens would converge towards the same results, which does not benefit to ViT. From the other perspective, YOLOS [73] provides a paradigm to unify classification and detection by using multiple initial tokens, but this encoder-only design would cause too much computational complexity. Based on the observation of CaiT [40], the later class token can reduce a few FLOPs of Transformer and improve a slight performance (from 79.9% to 80.5%). Segmenter [84] also shows the efficiency of this strategy in segmentation tasks.

In contrast to the multiple later tokens with encoder-only Transformer, the encoder-decoder structure saves more calculations. It standardizes Transformer methods in the detection [28] and segmentation [137] field by using a small set of object queries (mask embeddings). By combining the form of multiple later tokens and object queries (mask embeddings), the structure like Deformable DETR [67], which takes object query and learnable decoder embeddings (equivalent to multiple later tokens) as inputs, may unify the learnable embeddings based on different tasks into Transformer encoder-decoder.

C. Future Research

Visual Transformer methods achieve tremendous progresses and show promising results that approach to or surpass the records of SOTA CNN methods on multiple benchmarks. However, the technology is too immature to upset the dominant of convolution in the field of CV. Based on the analysis in Sec. VI-B, we point out some promising future directions of visual Transformer for further overall concatenation.

1) *Set Prediction:* As mentioned in Sec. VI-B3, the additional class token would converge consistently [38] because of the same gradient from the loss function. Set prediction strategy with bipartite loss function has been widely applied to visual Transformer in many dense prediction tasks [28],

[137]. As mentioned above, it is natural to consider set prediction design for classification tasks, such as multiple class token Transformer predicts mix-patches image by set prediction, which likes the data augmentation strategy of LV-ViT [41]. Furthermore, the one-to-one label assignment in the set prediction strategy leads to training instability during the early process, which may degrade accuracy in the final results. Improving set prediction with other label assignments and losses may be helpful for new detection frameworks.

2) *Self-Supervised Learning*: Self-supervised Transformer pre-training has standardized the field of NLP and obtained tremendous successes in various applications [2], [5]. As the self-supervision paradigms in CV, the convolutional siamese networks employ contrastive learning to perform self-supervised pre-training, which differs from the masked auto-encoders in NLP. Recently, some studies have tried designing a self-supervised visual Transformer to bridge the gap of pre-training methodology between vision and language. Most of them inherit the masked auto-encoders in NLP or contrastive learning schemes in CV. But, there is no specific supervised method for visual Transformer as revolutionized as GPT-3 in NLP. As described in Sec. VI-B3, the encoder-decoder structure may unify visual tasks by learned decoder embedding and positional encoding. It is worth further investigating the encoder-decoder Transformer for self-supervised learning.

D. Conclusion

After ViT demonstrated its effectiveness in CV tasks, visual Transformers have received considerable attention and undermined the dominant of CNN. In this paper, we have comprehensively reviewed more than a hundred of Transformer models which have been successively applied to various vision tasks, including classification, detection, and segmentation. For each task, a specific taxonomy is proposed to organize the recently-developed Transformer methods and their performances are evaluated over various prevailing benchmarks. From our integrative analysis and systematic comparison of these methods, a summary for remarkable improvements is provided in this paper, three essential issues for visual Transformers are discussed, and several potential research directions are further suggested for future investment. We do expect that this review paper can help readers understand visual Transformers better before deep exploration.

REFERENCES

- [1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Proc. NIPS*, vol. 30, 2017, pp. 5998–6008.
- [2] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving language understanding by generative pre-training,” 2018.
- [3] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
- [4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” in *Proc. NIPS*, vol. 33, 2020, pp. 1877–1901.
- [5] J. Devlin, M.-W. Chang, K. Lee, and K. N. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. NAACL*, 2018, pp. 4171–4186.
- [6] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” *arXiv:1907.11692*, 2019.
- [7] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “Albert: A lite bert for self-supervised learning of language representations,” in *Proc. ICLR*, 2020.
- [8] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” in *Proc. NIPS*, vol. 32, 2019, pp. 5753–5763.
- [9] D. W. Otter, J. R. Medina, and J. K. Kalita, “A survey of the usages of deep learning for natural language processing,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 2, pp. 604–624, 2020.
- [10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Proc. NIPS*, vol. 25, pp. 1097–1105, 2012.
- [11] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016, pp. 770–778.
- [12] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *Proc. ICML*. PMLR, 2019, pp. 6105–6114.
- [13] A. Galassi, M. Lippi, and P. Torroni, “Attention in natural language processing,” *IEEE Trans. Neural Netw. Learn. Syst.*, 2020.
- [14] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in *Proc. CVPR*, 2018, pp. 7794–7803.
- [15] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, “Cnnet: Criss-cross attention for semantic segmentation,” in *Proc. ICCV*, 2019, pp. 603–612.
- [16] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, “Gcnet: Non-local networks meet squeeze-excitation networks and beyond,” in *ICCV Workshop*, 2019, pp. 0–0.
- [17] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proc. CVPR*, 2018, pp. 7132–7141.
- [18] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, “Cbam: Convolutional block attention module,” in *Proc. ECCV*, 2018, pp. 3–19.
- [19] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, “Eca-net: Efficient channel attention for deep convolutional neural networks,” in *Proc. CVPR*, 2020, pp. 11534–11542.
- [20] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran, “Image transformer,” in *Proc. ICML*. PMLR, 2018, pp. 4055–4064.
- [21] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, “Relation networks for object detection,” in *Proc. CVPR*, 2018, pp. 3588–3597.
- [22] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, “Attention augmented convolutional networks,” in *Proc. ICCV*, 2019, pp. 3286–3295.
- [23] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, “Stand-alone self-attention in vision models,” in *Proc. NIPS*, vol. 32, 2019, pp. 68–80.
- [24] H. Zhao, J. Jia, and V. Koltun, “Exploring self-attention for image recognition,” in *Proc. CVPR*, 2020, pp. 10 076–10 085.
- [25] Z. Zheng, G. An, D. Wu, and Q. Ruan, “Global and local knowledge-aware attention network for action recognition,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 334–347, 2020.
- [26] J.-B. Cordonnier, A. Loukas, and M. Jaggi, “On the relationship between self-attention and convolutional layers,” in *Proc. ICLR*, 2020.
- [27] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” in *Proc. ICLR*, 2021.
- [28] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in *Proc. ECCV*. Springer, 2020, pp. 213–229.
- [29] H. Wang, Y. Zhu, H. Adam, A. L. Yuille, and L.-C. Chen, “Maxdeeplab: End-to-end panoptic segmentation with mask transformers,” in *Proc. CVPR*, 2021, pp. 5463–5474.
- [30] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, “Transformer tracking,” in *Proc. CVPR*, 2021, pp. 8126–8135.
- [31] Y. Jiang, S. Chang, and Z. Wang, “Transgan: Two pure transformers can make one strong gan, and that can scale up,” *arXiv preprint arXiv:2102.07074*, 2021.
- [32] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao, “Pre-trained image processing transformer,” in *Proc. CVPR*, 2021, pp. 12 299–12 310.

- [33] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proc. ICCV*, 2021, pp. 10 012–10 022.
- [34] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, “Cvt: Introducing convolutions to vision transformers,” in *Proc. ICCV*, 2021, pp. 22–31.
- [35] D. Zhou, Y. Shi, B. Kang, W. Yu, Z. Jiang, Y. Li, X. Jin, Q. Hou, and J. Feng, “Refiner: Refining self-attention for vision transformers,” *arXiv:2106.03714*, 2021.
- [36] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, “Scaling vision transformers,” *arXiv preprint arXiv:2106.04560*, 2021.
- [37] Z. Dai, H. Liu, Q. V. Le, and M. Tan, “Coatnet: Marrying convolution and attention for all data sizes,” *arXiv preprint arXiv:2106.04803*, 2021.
- [38] H. Touvron, M. Cord, D. Matthijs, F. Massa, A. Sablayrolles, and H. Jegou, “Training data-efficient image transformers & distillation through attention,” in *Proc. ICLR*, 2021, pp. 10 347–10 357.
- [39] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions,” in *Proc. ICCV*, 2021, pp. 568–578.
- [40] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou, “Going deeper with image transformers,” in *Proc. ICCV*, 2021, pp. 32–42.
- [41] Z. Jiang, Q. Hou, L. Yuan, D. Zhou, X. Jin, A. Wang, and J. Feng, “Token labeling: Training a 85.5% top-1 accuracy vision transformer with 56m parameters on imagenet,” *arXiv:2104.10858*, 2021.
- [42] L. Yuan, Q. Hou, Z. Jiang, J. Feng, and S. Yan, “Volo: Vision outlooker for visual recognition,” *arXiv preprint arXiv:2106.13112*, 2021.
- [43] B. Wu, C. Xu, X. Dai, A. Wan, P. Zhang, M. Tomizuka, K. Keutzer, and P. Vajda, “Visual transformers: Token-based image representation and processing for computer vision,” *arXiv preprint arXiv:2006.03677*, 2020.
- [44] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, “Bottleneck transformers for visual recognition,” in *Proc. CVPR*, 2021, pp. 16 519–16 529.
- [45] S. d’Ascoli, H. Touvron, M. Leavitt, A. Morcos, G. Biroli, and L. Sagun, “Convit: Improving vision transformers with soft convolutional inductive biases,” in *Proc. ICLR*, 2021, pp. 2286–2296.
- [46] K. Yuan, S. Guo, Z. Liu, A. Zhou, F. Yu, and W. Wu, “Incorporating convolution designs into visual transformers,” in *Proc. ICCV*, 2021, pp. 579–588.
- [47] Y. Li, K. Zhang, J. Cao, R. Timofte, and L. Van Gool, “Localvit: Bringing locality to vision transformers,” *arXiv:2104.05707*, 2021.
- [48] X. Chu, Z. Tian, B. Zhang, X. Wang, X. Wei, H. Xia, and C. Shen, “Conditional positional encodings for vision transformers,” *arXiv:2102.10882*, 2021.
- [49] Q. Zhang and Y. Yang, “Rest: An efficient transformer for visual recognition,” *arXiv:2105.13677*, 2021.
- [50] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Dollár, and R. B. Girshick, “Early convolutions help transformers see better,” *ArXiv*, vol. abs/2106.14881, 2021.
- [51] A. Vaswani, P. Ramachandran, A. Srinivas, N. Parmar, B. A. Hechtman, and J. Shlens, “Scaling local self-attention for parameter efficient visual backbones,” in *Proc. CVPR*, 2021, pp. 12 894–12 904.
- [52] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, “Transformer in transformer,” *arXiv:2103.00112*, 2021.
- [53] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen, “Twins: Revisiting the design of spatial attention in vision transformers,” *arXiv:2104.13840*, 2021.
- [54] J. Yang, C. Li, P. Zhang, X. Dai, B. Xiao, L. Yuan, and J. Gao, “Focal self-attention for local-global interactions in vision transformers,” *arXiv:2107.00641*, 2021.
- [55] P. Zhang, X. Dai, J. Yang, B. Xiao, L. Yuan, L. Zhang, and J. Gao, “Multi-scale vision longformer: A new vision transformer for high-resolution image encoding,” in *Proc. ICCV*, 2021, pp. 2998–3008.
- [56] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, F. E. H. Tay, J. Feng, and S. Yan, “Tokens-to-token vit: Training vision transformers from scratch on imagenet,” in *Proc. ICCV*, 2021, pp. 558–567.
- [57] B. Heo, S. Yun, D. Han, S. Chun, J. Choe, and S. J. Oh, “Rethinking spatial dimensions of vision transformers,” in *Proc. ICCV*, 2021, pp. 11 936–11 945.
- [58] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, “Pvtv2: Improved baselines with pyramid vision transformer,” *arXiv:2106.13797*, 2021.
- [59] D. Zhou, B. Kang, X. Jin, L. Yang, X. Lian, Q. Hou, and J. Feng, “Deepvit: Towards deeper vision transformer,” *arXiv:2103.11886*, 2021.
- [60] C. Gong, D. Wang, M. Li, V. Chandra, and Q. Liu, “Vision transformers with patch diversification,” *arXiv:2104.12753*, 2021.
- [61] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever, “Generative pretraining from pixels,” in *Proc. ICML*. PMLR, 2020, pp. 1691–1703.
- [62] Z. Li, Z. Chen, F. Yang, W. Li, Y. Zhu, C. Zhao, R. Deng, L. Wu, R. Zhao, M. Tang *et al.*, “Mst: Masked self-supervised transformer for visual representation,” *arXiv:2106.05656*, 2021.
- [63] H. Bao, L. Dong, and F. Wei, “Beit: Bert pre-training of image transformers,” *arXiv:2106.08254*, 2021.
- [64] X. Chen, S. Xie, and K. He, “An empirical study of training self-supervised vision transformers,” in *Proc. ICCV*, 2021, pp. 9640–9649.
- [65] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, “Emerging properties in self-supervised vision transformers,” in *Proc. ICCV*, 2021, pp. 9650–9660.
- [66] Z. Xie, Y. Lin, Z. Yao, Z. Zhang, Q. Dai, Y. Cao, and H. Hu, “Self-supervised learning with swin transformers,” *arXiv:2105.04553*, 2021.
- [67] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, “Deformable detr: Deformable transformers for end-to-end object detection,” in *Proc. ICLR*, 2021.
- [68] M. Zheng, P. Gao, X. Wang, H. Li, and H. Dong, “End-to-end object detection with adaptive clustering transformer,” *arXiv:2011.09315*, 2020.
- [69] P. Gao, M. Zheng, X. Wang, J. Dai, and H. Li, “Fast convergence of detr with spatially modulated co-attention,” in *Proc. ICCV*, 2021, pp. 3621–3630.
- [70] D. Meng, X. Chen, Z. Fan, G. Zeng, H. Li, Y. Yuan, L. Sun, and J. Wang, “Conditional detr for fast training convergence,” in *Proc. ICCV*, 2021, pp. 3651–3660.
- [71] Z. Yao, J. Ai, B. Li, and C. Zhang, “Efficient detr: Improving end-to-end object detector with dense prior,” *arXiv:2104.01318*, 2021.
- [72] Z. Sun, S. Cao, Y. Yang, and K. M. Kitani, “Rethinking transformer-based set prediction for object detection,” in *Proc. ICCV*, 2021, pp. 3611–3620.
- [73] Y. Fang, B. Liao, X. Wang, J. Fang, J. Qi, R. Wu, J. Niu, and W. Liu, “You only look at one sequence: Rethinking transformer in vision through object detection,” *arXiv:2106.00666*, 2021.
- [74] Z. Dai, B. Cai, Y. Lin, and J. Chen, “Up-detr: Unsupervised pre-training for object detection with transformers,” in *Proc. CVPR*, 2021, pp. 1601–1610.
- [75] D. Zhang, H. Zhang, J. Tang, M. Wang, X. Hua, and Q. Sun, “Feature pyramid transformer,” in *Proc. ECCV*. Springer, 2020, pp. 323–339.
- [76] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, and L. Zhang, “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers,” in *Proc. CVPR*, 2021, pp. 6881–6890.
- [77] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, and Y. Zhou, “Transunet: Transformers make strong encoders for medical image segmentation,” *arXiv:2102.04306*, 2021.
- [78] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, “Segformer: Simple and efficient design for semantic segmentation with transformers,” *arXiv:2105.15203*, 2021.
- [79] T. Prangemeier, C. Reich, and H. Koepll, “Attention-based transformers for instance segmentation of cells in microstructures,” in *BIBM*. IEEE, 2020, pp. 700–707.
- [80] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia, “End-to-end video instance segmentation with transformers,” in *Proc. CVPR*, 2021, pp. 8741–8750.
- [81] Y. Fang, S. Yang, X. Wang, Y. Li, C. Fang, Y. Shan, B. Feng, and W. Liu, “Queryinst: Parallelly supervised mask query for instance segmentation,” *arXiv:2105.01928*, 2021.
- [82] J. Hu, L. Cao, Y. Lu, S. Zhang, Y. Wang, K. Li, F. Huang, L. Shao, and R. Ji, “Istr: End-to-end instance segmentation with transformers,” *arXiv:2105.00637*, 2021.
- [83] B. Dong, F. Zeng, T. Wang, X. Zhang, and Y. Wei, “Solq: Segmenting objects by learning queries,” *arXiv:2106.02351*, 2021.
- [84] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, “Segmenter: Transformer for semantic segmentation,” in *Proc. ICCV*, 2021, pp. 7262–7272.
- [85] B. Cheng, A. G. Schwing, and A. Kirillov, “Per-pixel classification is not all you need for semantic segmentation,” *arXiv:2107.06278*, 2021.
- [86] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, “Efficient transformers: A survey,” *arXiv:2009.06732*, 2020.
- [87] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, “Transformers in vision: A survey,” *arXiv:2101.01169*, 2021.

- [88] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, “A survey on visual transformer,” *arXiv:2012.12556*, 2020.
- [89] T. Lin, Y. Wang, X. Liu, and X. Qiu, “A survey of transformers,” *arXiv:2106.04554*, 2021.
- [90] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Proc. NIPS*, vol. 27, 2014, pp. 3104–3112.
- [91] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “Lstm: A search space odyssey,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, 2016.
- [92] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv:1607.06450*, 2016.
- [93] P. P. Brahma, D. Wu, and Y. She, “Why deep learning works: A manifold disentanglement perspective,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 10, pp. 1997–2008, 2015.
- [94] Y. Dong, J.-B. Cordonnier, and A. Loukas, “Attention is not all you need: pure attention loses rank doubly exponentially with depth,” in *Proc. ICLR*, 2021, pp. 2793–2803.
- [95] H. Hu, Z. Zhang, Z. Xie, and S. Lin, “Local relation networks for image recognition,” in *Proc. ICCV*, 2019, pp. 3464–3473.
- [96] P. Shaw, J. Uszkoreit, and A. Vaswani, “Self-attention with relative position representations,” in *Proc. NAACL*, vol. 2, 2018, pp. 464–468.
- [97] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner *et al.*, “Relational inductive biases, deep learning, and graph networks,” *arXiv:1806.01261*, 2018.
- [98] S. Abnar, M. Dehghani, and W. Zuidema, “Transferring inductive biases through knowledge distillation,” *arXiv:2006.00555*, 2020.
- [99] B. Frénay and M. Verleysen, “Classification in the presence of label noise: a survey,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 845–869, 2013.
- [100] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng, “ a^2 -nets: Double attention networks,” in *Proc. NIPS*, vol. 31, 2018, pp. 352–361.
- [101] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proc. CVPR*, 2018, pp. 4510–4520.
- [102] A. Islam, S. Jia, and N. D. B. Bruce, “How much position information do convolutional neural networks encode,” in *Proc. ICLR*, 2020.
- [103] Y. Pang, M. Sun, X. Jiang, and X. Li, “Convolution in convolution for network in network,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1587–1597, 2017.
- [104] J. Lin, C. Gan, and S. Han, “Tsm: Temporal shift module for efficient video understanding,” in *Proc. ICCV*, 2019, pp. 7083–7093.
- [105] J. Gao, D. He, X. Tan, T. Qin, L. Wang, and T.-Y. Liu, “Representation degeneration problem in training natural language generation models,” in *Proc. ICLR*, 2019.
- [106] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, “Cutmix: Regularization strategy to train strong classifiers with localizable features,” in *Proc. ICCV*, 2019, pp. 6023–6032.
- [107] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, “Zero-shot text-to-image generation,” in *Proc. ICLR*, 2021, pp. 8821–8831.
- [108] C.-F. Chen, Q. Fan, and R. Panda, “Crossvit: Cross-attention multi-scale vision transformer for image classification,” in *Proc. ICCV*, 2021, pp. 357–366.
- [109] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, “Unsupervised learning of visual features by contrasting cluster assignments,” in *Proc. NIPS*, vol. 33, 2020, pp. 9912–9924.
- [110] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby, “Big transfer (bit): General visual representation learning,” in *Proc. ECCV*, 2020, pp. 491–507.
- [111] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, H. Lin, Z. Zhang, Y. Sun, T. He, J. Mueller, R. Manmatha *et al.*, “Resnest: Split-attention networks,” *arXiv:2004.08955*, 2020.
- [112] K. Wu, H. Peng, M. Chen, J. Fu, and H. Chao, “Rethinking and improving relative position encoding for vision transformer,” in *Proc. ICCV*, 2021, pp. 10 033–10 041.
- [113] M. A. Islam, M. Kowal, S. Jia, K. G. Derpanis, and N. D. Bruce, “Position, padding and predictions: A deeper look at position information in cnns,” *arXiv:2101.12322*, 2021.
- [114] C. Riquelme, J. Puigcerver, B. Mustafa, M. Neumann, R. Jenatton, A. S. Pinto, D. Keysers, and N. Houlsby, “Scaling vision with sparse mixture of experts,” *arXiv:2106.05974*, 2021.
- [115] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017.
- [116] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proc. CVPR*, 2016, pp. 779–788.
- [117] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proc. ICCV*, 2017, pp. 2980–2988.
- [118] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, “Object detection with deep learning: A review,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, 2019.
- [119] X. Zhang, F. Wan, C. Liu, X. Ji, and Q. Ye, “Learning to match anchors for visual object detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021.
- [120] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, “Deformable convolutional networks,” in *Proc. ICCV*, 2017, pp. 764–773.
- [121] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proc. CVPR*, 2017, pp. 2117–2125.
- [122] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, “Locality-sensitive hashing scheme based on p-stable distributions,” in *twentieth annual symposium on Computational geometry*, 2004, pp. 253–262.
- [123] Z. Tian, C. Shen, H. Chen, and T. He, “Fcos: Fully convolutional one-stage object detection,” in *Proc. ICCV*, 2019, pp. 9627–9636.
- [124] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proc. ICCV*, 2017, pp. 2961–2969.
- [125] Z. Cai and N. Vasconcelos, “Cascade r-cnn: Delving into high quality object detection,” in *Proc. CVPR*, 2018, pp. 6154–6162.
- [126] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proc. CVPR*, 2017, pp. 1492–1500.
- [127] K. Sun, B. Xiao, D. Liu, and J. Wang, “Deep high-resolution representation learning for human pose estimation,” in *Proc. CVPR*, 2019, pp. 5693–5703.
- [128] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *MICCAI*. Springer, 2015, pp. 234–241.
- [129] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, “Panoptic segmentation,” in *Proc. CVPR*, 2019, pp. 9404–9413.
- [130] H. Su, V. Jampani, D. Sun, O. Gallo, E. Learned-Miller, and J. Kautz, “Pixel-adaptive convolutional neural networks,” in *Proc. CVPR*, 2019, pp. 11 166–11 175.
- [131] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, C. Wang *et al.*, “Sparse r-cnn: End-to-end object detection with learnable proposals,” in *Proc. CVPR*, 2021, pp. 14 454–14 463.
- [132] Z. Cai and N. Vasconcelos, “Cascade r-cnn: high quality object detection and instance segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.
- [133] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, “Unified perceptual parsing for scene understanding,” in *Proc. ECCV*, 2018, pp. 418–434.
- [134] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2017.
- [135] H. Chen, K. Sun, Z. Tian, C. Shen, Y. Huang, and Y. Yan, “Blendmask: Top-down meets bottom-up for instance segmentation,” in *Proc. CVPR*, 2020, pp. 8573–8581.
- [136] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, “Solov2: Dynamic and fast instance segmentation,” in *Proc. NIPS*, vol. 33, 2020, pp. 17 721–17 732.
- [137] Y. Wang, R. Huang, S. Song, Z. Huang, and G. Huang, “Not all images are worth 16x16 words: Dynamic vision transformers with adaptive sequence length,” *arXiv:2105.15075*, 2021.