A Survey on Visual Transformer

Kai Han[1], Yunhe Wang[1*], Hanting Chen[1,2], Xinghao Chen[1], Jianyuan Guo[1], Zhenhua Liu[1,2], Yehui Tang[1,2],
An Xiao[1], Chunjing Xu[1], Yixing Xu[1], Zhaohui Yang[1,2], Yiman Zhang[1], Dacheng Tao[3*]
[1]Noah's Ark Lab, Huawei Technologies
[2]Peking University [3]The University of Sydney
{kai.han,yunhe.wang}@huawei.com, dacheng.tao@sydney.edu.au

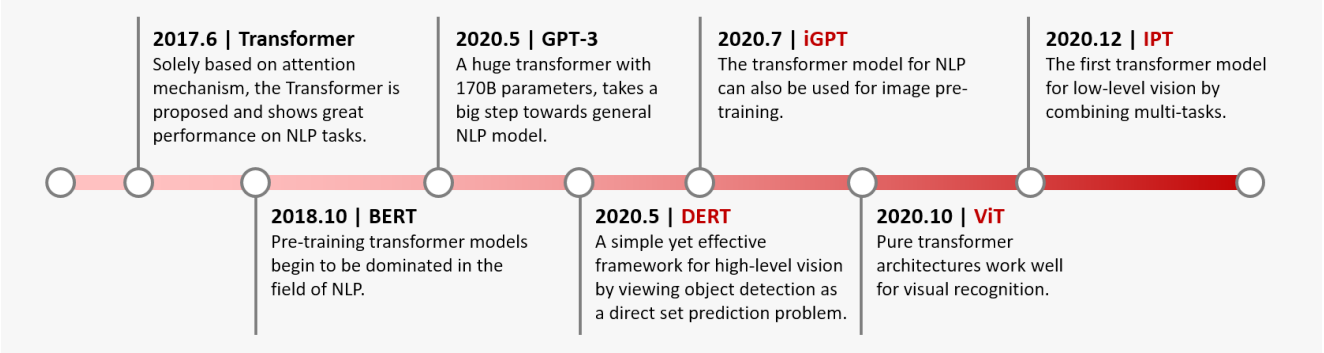

Figure 1: Key milestones in the development of transformer. The visual transformer models are marked in red.

Abstract

Transformer, first applied to the field of natural language processing, is a type of deep neural network mainly based on the self-attention mechanism. Thanks to its strong representation capabilities, researchers are looking at ways to apply transformer to computer vision tasks. In a variety of visual benchmarks, transformer-based models perform similar to or better than other types of networks such as convolutional and recurrent networks. Given its high performance and no need for human-defined inductive bias, transformer is receiving more and more attention from the computer vision community. In this paper, we review these visual transformer models by categorizing them in different tasks and analyzing their advantages and disadvantages. The main categories we explore include the backbone network, high/mid-level vision, low-level vision, and video processing. We also take a brief look at the self-attention mechanism in computer vision, as it is the base component in transformer. Furthermore, we include efficient transformer methods for pushing transformer into real device-based applications. Toward the end of this paper, we discuss the challenges and provide several further research directions for visual transformers.

*Corresponding authors. All authors are listed in alphabetical order of last name (except the primary and corresponding authors).

1. Introduction

Deep neural networks (DNNs) have become the fundamental infrastructure in today's artificial intelligence (AI) systems. Different types of tasks have typically involved different types of networks. For example, multi-layer perception (MLP) or the fully connected (FC) network is the classical type of neural network, which is composed of multiple linear layers and nonlinear activations stacked together [128, 129]. Convolutional neural networks (CNNs) introduce convolutional layers and pooling layers for processing shift-invariant data such as images [83, 80]. And recurrent neural networks (RNNs) utilize recurrent cells to process sequential data or time series data [130, 60]. Transformer is a new type of neural network. It mainly utilizes the self-attention mechanism [6, 111] to extract intrinsic features [152] and shows great potential for extensive use in AI applications

Transformer was first applied to natural language processing (NLP) tasks where it achieved significant improvements [152, 34, 11]. For example, Vaswani *et al*. [152] first proposed transformer based solely on attention mechanisms for machine translation and English constituency parsing tasks. Devlin *et al*. [34] introduced a new language representation model called BERT (short for Bidirectional Encoder Representations from Transformers), which pretrains a transformer on unlabeled text taking into account

Table 1: Representative works of visual transformers.

Category	Sub-category	Method	Highlights	Publication
Bakcbone	Image classification	iGPT [21]	Pixel prediction self-supervised learning, GPT model	ICML 2020
		ViT [36]	Image patches, standard transformer	ICLR 2021
High/Mid-level vision	Object detection	DETR [15]	Set-based prediction, bipartite matching, transformer	ECCV 2020
		Deformable DETR [193]	DETR, deformable attention module	ICLR 2021
		ACT [189]	Adaptive clustering transformer	arXiv 2020
		UP-DETR [33]	Unsupervised pre-training, random query patch detection	arXiv 2020
		TSP [143]	New bipartite matching, encoder-only transformer	arXiv 2020
	Segmentation	Max-DeepLab [155]	PQ-style bipartite matching, dual-path transformer	arXiv 2020
		VisTR [159]	Instance sequence matching and segmentation	arXiv 2020
		SETR [190]	sequence-to-sequence prediction, standard transformer	arXiv 2020
	Pose Estimation	Hand-Transformer [67]	Non-autoregressive transformer, 3D point set	ECCV 2020
		HOT-Net [68]	Structured-reference extractor	MM 2020
		METRO [92]	Progressive dimensionality reduction	arXiv 2020
Low-level vision	Image enhancement	IPT [20]	Multi-task, ImageNet pre-training, transformer model	arXiv 2020
		TTSR [167]	Texture transformer, RefSR	CVPR 2020
	Image generation	Image Transformer [113]	Pixel generation using transformer	ICML 2018
Video processing	Video inpainting	STTN [178]	Spatial-temporal adversarial loss	ECCV 2020
	Video captioning	Masked Transformer [191]	Masking network, event proposal	CVPR 2018
Efficient transformer	Decomposition	ASH [103]	Number of heads, importance estimation	NeurIPS 2019
	Distillation	TinyBert [73]	Various losses for different modules	EMNLP Findings 2020
	Quantization	FullyQT [118]	Fully quantized transformer	EMNLP Findings 2020
	Architecture design	ConvBert [72]	Local dependence, dynamic convolution	NeurIPS 2020

the context of each word (it is bidirectional). When BERT was published, it obtained state-of-the-art performance on 11 NLP tasks. Brown *et al*. [11] pre-trained a massive transformer-based model called GPT-3 (short for Generative Pre-trained Transformer 3) on 45 TB of compressed plaintext data using 175 billion parameters. It achieved strong performance on different types of downstream natural language tasks without requiring any fine-tuning. These transformer-based models, with their strong representation capacity, have achieved significant breakthroughs in NLP.

Inspired by the major success of transformer architectures in the field of NLP, researchers have recently applied transformer to computer vision (CV) tasks. In vision applications, CNNs were once considered the fundamental component [57, 126], but nowadays transformer is showing it is a viable alternative to CNN. Chen *et al*. [21] trained a sequence transformer to auto-regressively predict pixels, achieving results comparable to CNNs on image classification tasks. Another vision transformer model is ViT, which applies a pure transformer directly to sequences of image patches. Recently proposed by Dosovitskiy *et al*. [36], it has achieved state-of-the-art performance on multiple image recognition benchmarks. In addition to basic image classification, transformer has been utilized to address a variety of other computer vision problems, including object detection [15, 193], semantic segmentation, image processing, and video understanding. Thanks to its exceptional performance, more and more researchers are proposing transformer-based models for improving a wide range of visual tasks.

Due to the rapid increase in the number of transformer-based vision models, keeping pace with the rate of new progress is becoming increasingly difficult. As such, a survey of the existing works is urgent and would be beneficial for the community. In this paper, we focus on providing a comprehensive overview of the recent advances in visual transformers and discuss the potential directions for further improvement. To facilitate future research on different topics, we categorize the transformer models by their application scenarios, as listed in Table 1. The main categories include backbone network, high/mid-level vision, low-level vision, and video processing. High-level vision deals with the interpretation and use of what is seen in the image [150], whereas mid-level vision deals with how this information is organized into what we experience as objects and surfaces [77]. Given the gap between high- and mid-level vision is becoming more obscure in DNN-based vision systems [192, 102], we treat them as a single category here. A few examples of transformer models that address these high/mid-level vision tasks include DETR [15], deformable DETR [193] for object detection, and Max-DeepLab [155] for segmentation. Low-level image processing mainly deals with extracting descriptions from images (such descriptions are usually represented as images themselves) [43]. Typical applications of low-level image processing include super-resolution, image denoising, and style transfer. At present, only a few works [20, 113] in low-level vision use transformers, creating the need for further investigation. Another

category is video processing, which is an import part in both computer vision and image-based tasks. Due to the sequential property of video, transformer is inherently well suited for use on video tasks [191, 178], in which it is beginning to perform on par with conventional CNNs and RNNs. Here, we survey the works associated with transformer-based visual models in order to track the progress in this field. Figure 1 shows the development timeline of visual transformer — undoubtedly, there will be many more milestones in the future.

The rest of the paper is organized as follows. Section 2 discusses the formulation of the standard transformer and the self-attention mechanism. In Section 3, we describe the methods of transformer in NLP, as the research experience may be beneficial for vision tasks. Section 4 is the main part of the paper, in which we summarize the visual transformer models on backbone, high/mid-level vision, low-level vision, and video tasks. We also briefly describe the self-attention mechanism for CV and efficient transformer methods, as they are closely related to our main topic. In the final section, we give our conclusion and discuss several research directions and challenges.

2. Formulation of Transformer

Transformer [152] was first used in the field of neural language processing (NLP) on machine translation tasks. As shown in Figure 2, it consists of an encoder module and a decoder module with several encoders/decoders of the same architecture. Each encoder and decoder is composed of a self-attention layer and a feed-forward neural network, while each decoder also contains an encoder-decoder attention layer. Before transformer can be used to translate sentences, each word in the sentence needs to be embedded into a vector with $d_{model} = 512$ dimensions.

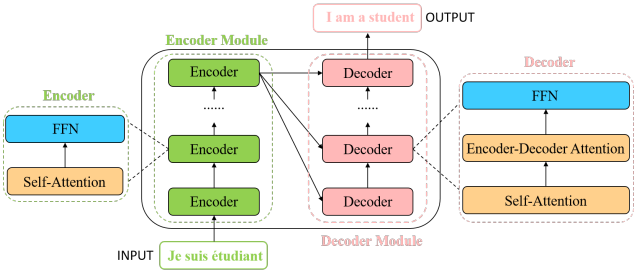

Figure 2: Pipeline of vanilla transformer.

2.1. Self-Attention Layer

In the self-attention layer, the input vector is first transformed into three different vectors: the query vector q, the key vector k and the value vector v with dimension $d_q = d_k = d_v = d_{model} = 512$. Vectors derived from different inputs are then packed together into three different matrices, namely, Q, K and V. Subsequently, the attention function between different input vectors is calculated as follows (and shown in Figure 3 left):

- **Step 1:** Compute scores between different input vectors with $S = Q \cdot K^\top$;
- **Step 2:** Normalize the scores for the stability of gradient with $S_n = S/\sqrt{d_k}$;
- **Step 3:** Translate the scores into probabilities with softmax function $P = \mathrm{softmax}(S_n)$;
- **Step 4:** Obtain the weighted value matrix with $Z = V \cdot P$.

The process can be unified into a single function:

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{Q \cdot K^\top}{\sqrt{d_k}}) \cdot V. \quad (1)$$

The logic behind Eq. 1 is simple. Step 1 computes scores between two different vectors, and these scores determine the degree of attention that we give other words when encoding the word at the current position. Step 2 normalizes the scores to enhance gradient stability for improved training, and step 3 translates the scores into probabilities. Finally, each value vector is multiplied by the sum of the probabilities. Vectors with larger probabilities receive additional focus from the following layers.

The encoder-decoder attention layer in the decoder module is similar to the self-attention layer in the encoder module with the following exceptions: The key matrix K and value matrix V are derived from the encoder module, and the query matrix Q is derived from the previous layer.

Note that the preceding process is irrelevant to the position of each word, meaning that the self-attention layer lacks the ability to capture the positional information of words in a sentence. To address this issue and allow the final input vector of the word to be obtained, a positional encoding with dimension d_{model} is added to the original input embedding. Specifically, the position is encoded with the following equations:

$$PE(pos, 2i) = sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}}); \quad (2)$$

$$PE(pos, 2i+1) = cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}}), \quad (3)$$

in which pos denotes the position of the word in a sentence, and i represents the current dimension of the positional encoding.

2.2. Multi-Head Attention

Multi-head attention is a mechanism that can be used to boost the performance of the vanilla self-attention layer.

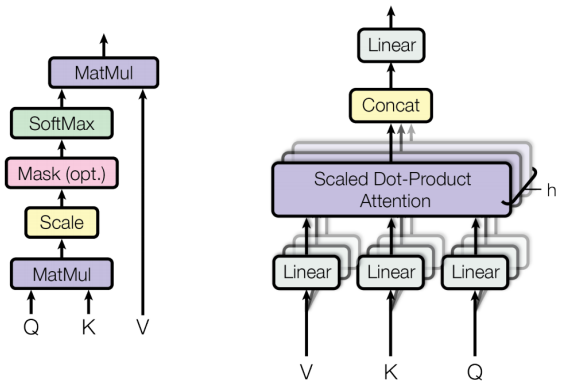

Figure 3: (Left) Self-attention process. (Right) Multi-head attention. The image is from [152].

Note that for a given reference word, we often want to focus on several other words when going through the sentence. A single-head self-attention layer limits our ability to focus on one or more specific positions without influencing the attention on other equally important positions at the same time. This is achieved by giving attention layers different representation subspace. Specifically, different query, key and value matrices are used for different heads, and these matrices can project the input vectors into different representation subspace after training due to random initialization.

To elaborate on this in greater detail, given an input vector and the number of heads h, the input vector is first transformed into three different groups of vectors: the query group, the key group and the value group. In each group, there are h vectors with dimension $d_{q'} = d_{k'} = d_{v'} = d_{model}/h = 64$. The vectors derived from different inputs are then packed together into three different groups of matrices: $\{Q_i\}_{i=1}^h$, $\{K_i\}_{i=1}^h$ and $\{V_i\}_{i=1}^h$. The multi-head attention process is shown as follows:

$$\begin{aligned}\text{MultiHead}(Q', K', V') &= \text{Concat}(\text{head}_1, \cdots, \text{head}_h)W^o, \\ \text{where head}_i &= \text{Attention}(Q_i, K_i, V_i). \end{aligned} \quad (4)$$

Here, Q' (and similarly K' and V') is the concatenation of $\{Q_i\}_{i=1}^h$, and $W^o \in \mathbb{R}^{d_{model} \times d_{model}}$ is the linear projection matrix.

2.3. Other Key Concepts in Transformer

Residual Connection in the Encoder and Decoder. As shown in Figure 4, a residual connection is added to each sub-layer in the encoder and decoder. This strengthens the flow of information in order to achieve higher performance. A layer-normalization [5] is followed after the residual connection. The output of these operations can be described as:

$$\text{LayerNorm}(X + \text{Attention}(X)). \quad (5)$$

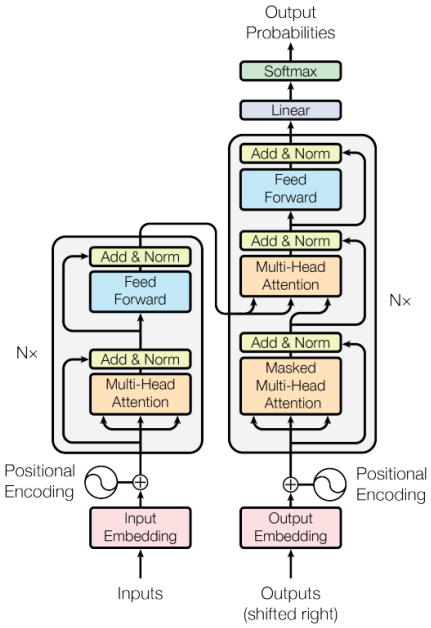

Figure 4: Structure of transformer. The image is from [152].

Here, X is used as the input of self-attention layer. This is because the query, key and value matrices Q, K and V are all derived from the same input matrix X.

Feed-Forward Network. A feed-forward network (FFN) is applied after the self-attention layers in each encoder and decoder. It consists of two linear transformation layers and a nonlinear activation function within them, and can be denoted as the following function:

$$\text{FFN}(X) = W_2\sigma(W_1X), \quad (6)$$

where W_1 and W_2 are the two parameter matrices of the two linear transformation layers, and σ represents the nonlinear activation function, such as GELU [58]. The dimensionality of the hidden layer is $d_h = 2048$.

Final Layer in the Decoder. The final layer in the decoder is used to turn the stack of vectors back into a word. This is achieved by a linear layer followed by a softmax layer. The linear layer projects the vector into a logits vector with d_{word} dimensions, in which d_{word} is the number of words in the vocabulary. The softmax layer is then used to transform the logits vector into probabilities.

When used for CV tasks, most transformers adopt the original transformer's encoder module. Such transformers can be treated as a new feature selector. Compared with CNNs, which focus only on local characteristics, transformer can capture long-distance characteristics, meaning that it can easily derive global information. And in contrast to RNNs, whose hidden state must be computed sequentially, transformer is more efficient because the output of the self-attention layer and the fully connected layers can

be computed in parallel and easily accelerated. From this, we can conclude that further study into using transformer in computer vision as well as NLP would yield beneficial results.

3. Revisiting Transformers for NLP

Before transformer was developed, RNNs (*e.g.*, GRU [31] and LSTM [61]) with added attention empowered most the state-of-the-art language models. However,RNNs require the information flow to be processed sequentially from the previous hidden states to the next one. This rules out the possibility of using acceleration and parallelization during training, and consequently hinders the potential of RNNs to process longer sequences or build larger models. In 2017, Vaswani *et al*. [152] proposed transformer, a novel encoder-decoder architecture built solely on multi-head self-attention mechanisms and feed-forward neural networks. Its purpose was to solve seq-to-seq natural language tasks (*e.g.*, machine translation) easily by acquiring global dependencies. The subsequent success of transformer demonstrates that leveraging attention mechanisms alone can achieve performance comparable with attentive RNNs. Furthermore, the architecture of transformer lends itself to massively parallel computing, which enables training on larger datasets. This has given rise to the surge of large pre-trained models (PTMs) for natural language processing.

BERT [34] and its variants (*e.g.*, SpanBERT [75], RoBERTa [98]) are a series of PTMs built on the multi-layer transformer encoder architecture. Two tasks are conducted on BookCorpus [194] and English Wikipedia datasets at the pre-training stage of BERT: 1) Masked language modeling (MLM), which involves first randomly masking out some tokens in the input and then training the model to predict; 2) Next sentence prediction, which uses paired sentences as input and predicts whether the second sentence is the original one in the document. After pre-training, BERT can be fine-tuned by adding an output layer on a wide range of downstream tasks. More specifically, when performing sequence-level tasks (*e.g.*, sentiment analysis), BERT uses the representation of the first token for classification; for token-level tasks (*e.g.*, name entity recognition), all tokens are fed into the softmax layer for classification. At the time of its release, BERT achieved the state-of-the-art performance on 11 NLP tasks, setting a milestone in pre-trained language models. Generative Pre-trained Transformer models (*e.g.*, GPT [122], GPT-2 [123]) are another type of PTMs based on the transformer decoder architecture, which uses masked self-attention mechanisms. The main difference between the GPT series and BERT is the way in which pre-training is performed. Unlike BERT, GPT models are unidirectional language models pre-trained using Left-to-Right (LTR) language modeling. Furthermore, BERT learns the sentence separator ([SEP]) and classifier token ([CLS]) embeddings during pre-training, whereas these embeddings are involved in only the fine-tuning stage of GPT. Due to its unidirectional pre-training strategy, GPT achieves superior performance in many natural language generation tasks. More recently, a massive transformer-based model called GPT-3, which has an astonishing 175 billion parameters, was developed [11]. By pre-training on 45 TB of compressed plaintext data, GPT-3 can directly process different types of downstream natural language tasks without fine-tuning. As a result, it achieves strong performance on many NLP datasets, including both natural language understanding and generation. Since the introduction of transformer, many other models have been proposed in addition to the transformer-based PTMs mentioned earlier. We list a few representative models in Table 2 for interested readers, but this is not the focus of our study.

Table 2: List of representative language models built on transformer. Transformer is the standard encoder-decoder architecture. Transformer Enc. and Dec. represent the encoder and decoder, respectively. Decoder uses mask self-attention to prevent attending to the future tokens. The data of the Table is from [121].

Models	Architecture	# of Params	Fine-tuning
GPT [122]	Transformer Dec.	117M	Yes
GPT-2 [123]	Transformer Dec.	117M-1542M	No
GPT-3 [11]	Transformer Dec.	125M-175B	No
BERT [34]	Transformer Enc.	110M-340M	Yes
RoBERTa [98]	Transformer Enc.	355M	Yes
XLNet [169]	Two-Stream Transformer Enc.	$\approx$ BERT	Yes
ELECTRA [32]	Transformer Enc.	335M	Yes
UniLM [35]	Transformer Enc.	340M	Yes
BART [85]	Transformer	110% of BERT	Yes
T5 [124]	Transfomer	220M-11B	Yes
ERNIE (THU) [185]	Transform Enc.	114M	Yes
KnowBERT [115]	Transformer Enc.	253M-523M	Yes

Apart from the PTMs trained on large corpora for general NLP tasks, transformer-based models have also been applied in many other NLP-related domains and to multi-modal tasks.

BioNLP Domain. Transformer-based models have outperformed many traditional biomedical methods. Some examples of such models include BioBERT [84], which uses a transformer architecture for biomedical text mining tasks, and SciBERT [9], which is developed by training transformer on 114M scientific articles (covering biomedical and computer science fields) with the aim of executing NLP tasks in the scientific domain more precisely. Another example is ClinicalBERT, proposed by Huang *et al*. [66]. It utilizes transformer to develop and evaluate continuous representations of clinical notes. One of the side effects of this

is that the attention map of ClinicalBERT can be used to explain predictions, thereby allowing high-quality connections between different medical contents to be discovered.

Multi-Modal Tasks. Owing to the success of transformer across text-based NLP tasks, many researches are keen to exploit its potential for processing multi-modal tasks (*e.g.*, video-text, image-text and audio-text). One example of this is VideoBERT [141], which uses a CNN-based module to pre-process videos in order to obtain representation tokens. A transformer encoder is then trained on these tokens to learn the video-text representations for downstream tasks, such as video caption. Some other examples include VisualBERT [86] and VL-BERT [140], which adopt a single-stream unified transformer to capture visual elements and image-text relationship for downstream tasks such as visual question answering (VQA) and visual commonsense reasoning (VCR). In addition, several studies such as SpeechBERT [29] explore the possibility of encoding audio and text pairs with a transformer encoder to process auto-text tasks such as speech question answering (SQA).

The rapid development of transformer-based models on a variety of NLP-related tasks demonstrates its structural superiority and versatility, opening up the possibility that it will become a universal module applied in many AI fields other than just NLP. The following part of this survey focuses on the applications of transformer in a wide range of computer vision tasks that have emerged over the past two years.

4. Visual Transformer

In this section, we review the applications of transformer-based models in computer vision, including image classification, high/mid-level vision, low-level vision and video processing. We also briefly summarize the applications of the self-attention mechanism and model compression methods for efficient transformer.

4.1. Backbone for Image Classification

Inspired by the success of that transformer has achieved in the field of NLP, some researchers have explored whether similar models can learn useful representations for images. Given that images involve more dimensions, noise and redundant modality compared with text, they are believed to be more difficult for generative modeling.

Other than CNNs, the transformer can be used as backbone networks for image classification. Wu *et al*. [163] adopted ResNet as a convenient baseline and used visual transformers to replace the last stage of convolutions. Specifically, they apply convolutional layers to extract low-level features that are then fed into the visual transformer. For the visual transformer, they use a *tokenizer* to group pixels into a small number of *visual tokens*, each representing a semantic concept in the image. These *visual tokens* are used directly for image classification, with the transformers being used to model the relationships between tokens The works that purely use transformer for image classification include iGPT [21], ViT [36] and DeiT [148].

4.1.1 iGPT

Generative pre-training methods for images have existed for a long time, Chen *et al*. [21] re-examined this class of methods and combined it with self-supervised methods. This approach consists of a pre-training stage followed by a fine-tuning stage. During the pre-training stage, auto-regressive and BERT objectives are explored.To implement pixel prediction, a sequence transformer architecture is adopted instead of language tokens (as used in NLP). Pre-training can be thought of as a favorable initialization or regularizer when used in combination with early stopping. During the fine-tuning stage, they add a small classification head to the model. This helps optimize a classification objective and adapts all weights.

Given an unlabeled dataset X consisting of high dimensional data $x = (x_1, \cdots, x_n)$, they train the model by minimizing the negative log-likelihood of the data:

$$L_{AR} = \mathop{\mathbb{E}}_{x \sim X}[-\log p(x)] \tag{7}$$

where $p(x)$ is the density of the data of images, which can be modeled as:

$$p(x) = \prod_{i=1}^{n} p(x_{\pi_i} | x_{\pi_1}, \cdots, x_{\pi_{i-1}}, \theta) \tag{8}$$

Here, the identity permutation $\pi_i = i$ is adopted for $1 \leqslant i \leqslant n$, which is also known as raster order. Chen *et al*. also considered the BERT objective, which samples a subsequence $M \subset [1, n]$ such that each index i independently has probability 0.15 of appearing in M. M is called the BERT mask, and the model is trained by minimizing the negative log-likelihood of the "masked" elements x_M conditioned on the "unmasked" ones $x_{[1,n]\setminus M}$:

$$L_{BERT} = \mathop{\mathbb{E}}_{x \sim X} \mathop{\mathbb{E}}_{M} \sum_{i \in M} [-\log p(x_i | x_{[1,n]\setminus M})] \tag{9}$$

During the pre-training stage, they pick either L_{AR} or L_{BERT} and minimize the loss over the pre-training dataset.

GPT-2 [123] formulation of the transformer decoder block is used. In particular, layer norms precede both the attention and Multi-Layer Perceptron (MLP) operations, and all operations are strictly performed on residual paths. The attention operation is the only one that involves mixing across sequence elements. To ensure proper conditioning when training the AR objective, Chen *et al*. apply the standard upper triangular mask to the $n \times n$ matrix of attention logits. No attention logit masking is required when the

BERT objective is used, but once content embeddings are applied to the input sequence, Chen *et al*. zero out the positions.

Following the final transformer layer, they apply a layer norm and learn a projection from the output to logits parameterizing the conditional distributions at each sequence element. When training BERT, they simply ignore the logits at unmasked positions.

During the fine-tuning stage, they average pool the output of the final layer norm n^L across the sequence dimension to extract a d-dimensional vector of features per example:

$$f^L = \langle n_i^L \rangle_i \tag{10}$$

They learn a projection from f_L to class logits and use this projection to minimize a cross entropy loss L_{CLF}. Practical applications offer empirical evidence that the joint objective $L_{GEN}+L_{CLF}$ works even better, where $L_{GEN} \in \{L_{AR}, L_{BERT}\}$.

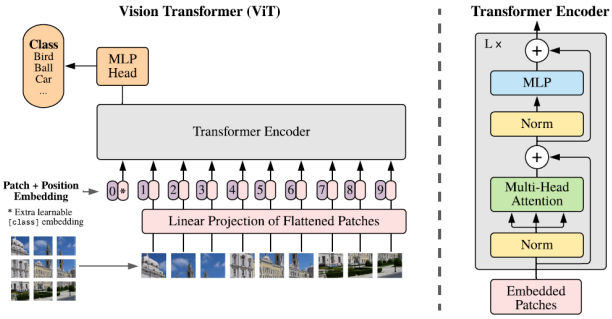

Figure 5: The framework of the Vision Transformer (The image is from [36]).

4.1.2 ViT

Dosovitskiy *et al*. [36] recently proposed Vision Transformer (ViT), which is a pure transformer that performs well on image classification task when applied directly to the sequences of image patches. They follow transformer's original design as much as possible. Figure 5 shows the framework of ViT.

To handle 2D images, the image $x \in \mathbb{R}^{H\times W\times C}$ is reshaped into a sequence of flattened 2D patches $x_p \in \mathbb{R}^{N\times(P^2\cdot C)}$. (H, W) is the resolution of the original image, while (P, P) is the resolution of each image patch. The effective sequence length for the transformer is therefore $N = HW/P^2$. Because the transformer uses constant widths in all of its layers, a trainable linear projection maps each vectorized path to the model dimension D, the output of which is referred to as patch embeddings.

Similar to BERT's $[class]$ token, a learnable embedding is applied to the sequence of embedding patches. The state of this embedding serves as the image representation. During both pre-training and fine-tuning stage, the classification heads are attached to the same size. In addition, 1D position embeddings are added to the patch embeddings in order to retain positional information. Dosovitskiy *et al*. have explored different 2D-aware variants of position embeddings, none of which yield any significant gains over standard 1D position embeddings. The joint embeddings sever as input to the encoder. It is worth noting that ViT utilizes only the standard transformer's encoder, whose output precedes an MLP head.

In most cases, ViT is pre-trained on large datasets, and then fine-tuned for smaller downstream tasks. For this, the pre-trained prediction head is removed and a zero-initialized $D \times K$ feed-forward layer is attached, where K is the number of downstream classes. Using higher resolutions during the fine-tuning stage than during the pre-training stage is often beneficial. For example, a larger effective sequence length can be obtained when feeding higher resolution images, even though the patch size remains the same. Although ViT can handle arbitrary sequence lengths, the pre-trained position embeddings may no longer be meaningful. Dosovitskiy *et al*. therefore perform 2D interpolation of the pre-trained position embeddings according to their location in the original image. Note that an inductive bias about the 2D structure of the images is manually injected into ViT only during resolution adjustment and patch extraction.

ViT yields modest results when trained on mid-sized datasets such as ImageNet, achieving accuracies of a few percentage points below ResNets of comparable size. Because transformers lack some inductive biases inherent to CNNs–such as translation equivariance and locality–they do not generalize well when trained on insufficient amounts of data. However, the authors found that training the models on large datasets (14 million to 300 million images) surpassed inductive bias. When pre-trained at sufficient scale, transformers achieve excellent results on tasks with fewer datapoints. For example, when pre-trained on the JFT-300M dataset, ViT approached or even exceeded state of the art performance on multiple image recognition benchmarks. Specifically, it reached an accuracy of 88.36% on ImageNet, 99.50% on CIFAR-10, 94.55% on CIFAR-100, and 77.16% on the VTAB suite of 19 tasks. The results of both ViT and iGPT are shown in table 3.

Touvron *et al*. [148] proposed a competitive convolution-free transformer, called Data-efficient image transformer (DeiT), by training on only the ImageNet database. DeiT-B, the reference vision transformer, has the same architecture as ViT-B and employs 86 million parameters. With a strong data augmentation, DeiT-B achieves top-1 accuracy of 83.1% (single-crop evaluation) on ImageNet with no external data. In addition, the authors observe that using a

CNN teacher gives better performance than using a transformer. Specifically, DeiT-B can achieve top-1 accuracy 84.40% with the help of a token-based distillation.

In conclusion, iGPT recalls the generative pre-training method and combines it with self-supervised methods, but the results are unsatisfactory. On the other hand, ViT achieves much better results, especially when it utilizes a larger dataset (JFT-300M). In a similar vein, DeiT achieves better performance with a more cautious training strategy and a token-based distillation. Given that the structure of ViT and that of transformer in NLP are similar, it becomes a question of how to explicitly identify the correlations in the intra-patch and inter-patch. Besides, although ViT treats patches that are the same size equally, the complexity of each patch is different. As yet, this characteristic has not been fully employed.

4.2. High/Mid-level Vision

Recently there has been growing interest in using transformer for high/mid-level computer vision tasks, such as object detection [16, 193, 8, 173, 109], lane detection [97], segmentation [159, 155, 190] and pose estimation [67, 68, 92, 168]. We review these methods in this section.

4.2.1 Generic Object Detection

Traditional object detectors are mainly built upon CNNs, but transformer-based object detection has gained significant interest recently due to its advantageous capability.

Some object detection methods have attempted to use transformer's self-attention mechanism and then enhance the specific modules for modern detectors, such as feature fusion module [180] and prediction head [27]. We discuss this later in Section 4.5. Transformer-based object detection methods are broadly categorized into two groups: transformer-based set prediction methods [15, 193, 143, 189] and transformer-based backbone methods [8, 110], as shown in Fig. 6.

Transformer-based Set Prediction for Detection. As a pioneer for transformer-based detection method, the detection transformer (DETR) proposed by Carion *et al*. [16] redesigns the framework of object detection. DETR, a simple and fully end-to-end object detector, treats the object detection task as an intuitive set prediction problem, eliminating traditional hand-crafted components such as anchor generation and non-maximum suppression (NMS) post-processing. As shown in Fig. 7, DETR starts with a CNN backbone to extract features from the input image. To supplement the image features with position information, fixed positional encodings are added to the flattened features before the features are fed into the encoder-decoder transformer. The decoder consumes the embeddings from the encoder along with N learned positional encodings (object queries), and produces N output embeddings. Here N is a predefined parameter and typically larger than the number of objects in an image. Simple feed-forward networks (FFNs) are used to compute the final predictions, which include the bounding box coordinates and class labels to indicate the specific class of object (or to indicate that no object exists). Unlike the original transformer, which computes predictions sequentially, DETR decodes N objects in parallel. DETR employs a bipartite matching algorithm to assign the predicted and ground-truth objects. As shown in Eq. 11, the Hungarian loss is exploited to compute the loss function for all matched pairs of objects.

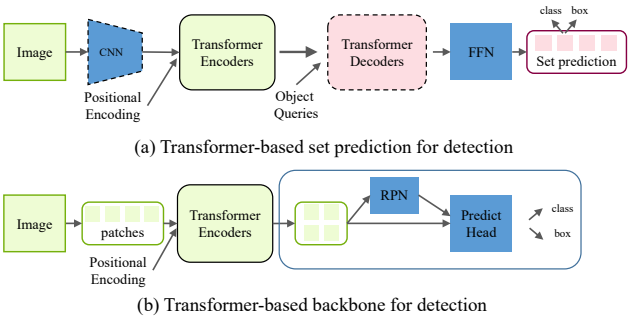

Figure 6: General framework of transformer-based object detection. (a) The object detection task is formulated as a set prediction task via transformer (*e.g*., DETR [15]). (b) The vision transformer is used instead of the CNN backbone for traditional object detectors (*e.g*., ViT-FRCNN [8]).

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \right], \quad (11)$$

where y and $\hat{y}$ are the ground truth and prediction of objects, $\hat{\sigma}$ is the optimal assignment, c_i and $\hat{p}_{\hat{\sigma}(i)}(c_i)$ are the target class label and predicted label, respectively, and b_i and $\hat{b}_{\hat{\sigma}}(i)$ are the ground truth and predicted bounding box, respectively. DETR shows impressive performance on object detection, delivering comparable accuracy and speed with the popular and well-established Faster R-CNN [126] baseline on COCO benchmark.

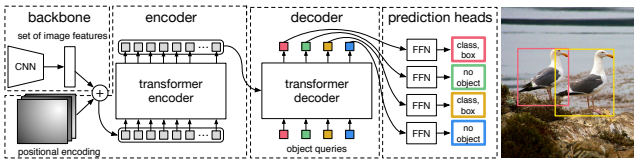

Figure 7: The overall architecture of DETR. The image is from [16].

DETR is a new design for the object detection framework based on transformer and empowers the community to develop fully end-to-end detectors. However, the vanilla

Table 3: Comparison of experimental results between CNN and transformers. †Result using knowledge distillation.

Method	Network	Pre-training policy	Pre-training data	Dataset	Params (M)	Acc (Top-1)
BiT-L [78]	CNN	Supervised	JFT-300M	CIFAR-10	60	99.37
				CIFAR-100	60	93.51
				ImageNet	60	87.54
iGPT [21]	Transformer	Self-supervised	ImageNet	CIFAR-10	1362	99.00
				CIFAR-100	1362	91.70
ViT [36]	Transformer	Supervised	ImageNet	CIFAR-10	86	98.13
				CIFAR-100	86	87.13
				ImageNet	86	77.91
			JFT-300M	CIFAR-10	86	99.00
				CIFAR-100	86	91.87
				ImageNet	86	84.15
				CIFAR-10	632	99.50
				CIFAR-100	632	94.55
				ImageNet	632	88.36
DeiT [148]	Transformer	Supervised	ImageNet	ImageNet	86	83.10
					86	84.40†

DETR poses several challenges, specifically, longer training schedule and poor performance for small objects. To address these challenges, Zhu *et al*. [193] proposed Deformable DETR, which has become a popular method that significantly improves the detection performance. The deformable attention module attends to a small set of key positions around a reference point rather than looking at all spatial locations on image feature maps as performed by the original multi-head attention mechanism in transformer. This approach significantly reduces the computational complexity and brings benefits in terms of fast convergence. More importantly, the deformable attention module can be easily applied for fusing multi-scale features. Deformable DETR achieves better performance than DETR with $10\times$ less training cost and $1.6\times$ faster inference speed. And by using an iterative bounding box refinement method and two-stage scheme, Deformable DETR can further improve the detection performance.

Given the high computation complexity associated with DETR, Zheng *et al*. [189] proposed an Adaptive Clustering Transformer (ACT) to reduce the computation cost of pre-trained DETR without any training process. ACT adaptively clusters the query features using a locality sensitivity hashing (LSH) method and broadcasts the attention output to the queries represented by the selected prototypes. ACT is used to replace the self-attention module of the pre-trained DETR model without requiring any re-training. This approach significantly reduces the computation cost while only slightly lowering the accuracy. The performance drop can be further reduced by utilizing a multi-task knowledge distillation (MTKD) method, which exploit the original transformer to distill the ACT module with a few epochs of fine-tuning.

Sun *et al*. [143] investigated why the DETR model has slow convergence and discovered that this is mainly due to the cross-attention module in the transformer decoder. To address this issue, an encoder-only version of DETR is proposed, achieving considerable improvement in terms of detection accuracy and training convergence. In addition, a new bipartite matching scheme is designed for greater training stability and faster convergence and two transformer-based set prediction models, *i.e*. TSP-FCOS and TSP-RCNN, are proposed to improve encoder-only DETR with feature pyramids. These new models achieve better performance compared with the original DETR model.

Inspired by the pre-training transformer scheme in NLP, Dai *et al*. [33] proposed unsupervised pre-training for object detection (UP-DETR). Specifically, a novel unsupervised pretext task named random query patch detection is proposed to pre-train the DETR model. With this unsupervised pre-training scheme, UP-DETR significantly improves the detection accuracy on a relatively small dataset (PASCAL VOC). On the COCO benchmark with sufficient training data, UP-DETR still outperforms DETR, demonstrating the effectiveness of the unsupervised pre-training scheme.

Transformer-based Backbone for Detection. Unlike DETR which redesigns object detection as a set prediction tasks via transformer, Beal *et al*. [8] proposed to utilize transformer as a backbone for common detection frameworks such as Faster R-CNN [127]. The input image is divided into several patches and fed into a vision transformer, whose output embedding features are reorganized according to spatial information before passing through a detection head for the final results. A massive pre-training transformer backbone could bring benefits to the proposed ViT-FRCNN.

Transformer-based methods have shown strong performance compared with CNN-based detectors, in terms of

Table 4: Comparison of different transformer-based object detectors using ResNet-50 on COCO 2017 val set. Running speed (FPS) is evaluated on an NVIDIA Tesla V100 GPU as reported in [193]. †Estimated speed according to the reported number in the paper. ‡ViT backbone is pre-trained on ImageNet-21k. *ViT backbone is pre-trained on an private dataset with 1.3 billion images.

Method	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	#Params (M)	GFLOPs	FPS
CNN based										
FCOS [147]	36	41.0	59.8	44.1	26.2	44.6	52.2	-	177	23†
Faster R-CNN + FPN [127]	109	42.0	62.1	45.5	26.6	45.4	53.4	42	180	26
Transformer based										
DETR [15]	500	42.0	62.4	44.2	20.5	45.8	61.1	41	86	28
DETR-DC5 [15]	500	43.3	63.1	45.9	22.5	47.3	61.1	41	187	12
Deformable DETR [193]	50	46.2	65.2	50.0	28.8	49.2	61.7	40	173	19
TSP-FCOS [143]	36	43.1	62.3	47.0	26.6	46.8	55.9	-	189	20†
TSP-RCNN [143]	96	45.0	64.5	49.6	29.7	47.7	58.0	-	188	15†
ACT+MKKD (L=32) [189]	-	43.1	-	-	61.4	47.1	22.2	-	169	14†
ACT+MKKD (L=16) [189]	-	40.6	-	-	59.7	44.3	18.5	-	156	16†
ViT-B/16-FRCNN‡ [8]	21	36.6	56.3	39.3	17.4	40.0	55.5	-	-	-
ViT-B/16-FRCNN* [8]	21	37.8	57.4	40.1	17.8	41.4	57.3	-	-	-
UP-DETR [33]	150	40.5	60.8	42.6	19.0	44.4	60.0	41	-	-
UP-DETR [33]	300	42.8	63.0	45.3	20.8	47.1	61.7	41	-	-

both accuracy and running speed. Table 4 shows the detection results for different transformer-based object detectors mentioned earlier on the COCO 2012 val set.

4.2.2 Other Detection Tasks

Pedestrian Detection. Because the distribution of objects is very dense in occlusion and crowd scenes, additional analysis and adaptation are often required when common detection networks are applied to pedestrian detection tasks. Lin *et al*. [93] revealed that sparse uniform queries and a weak attention field in the decoder result in performance degradation when directly applying DETR or Deformable DETR to pedestrian detection tasks. To alleviate these drawbacks, the authors proposes Pedestrian End-to-end Detector (PED), which employs a new decoder called Dense Queries and Rectified Attention field (DQRF) to support dense queries and alleviate the noisy or narrow attention field of the queries. They also proposed V-Match, which achieves additional performance improvements by fully leveraging visible annotations.

Lane Detection. Based on PolyLaneNet [145], Liu *et al*. [97] proposed a method called LSTR, which improves performance of curve lane detection by learning the global context with a transformer network. Similar to PolyLaneNet, LSTR regards lane detection as a task of fitting lanes with polynomials and uses neural networks to predict the parameters of polynomials. To capture slender structures for lanes and the global context, LSTR introduces a transformer network into the architecture. This enables processing of low-level features extracted by CNNs. In addition, LSTR uses Hungarian loss to optimize network parameters. As demonstrated in [97], LSTR outperforms PolyLaneNet, with 2.82% higher accuracy and 3.65-time higher FPS using 5-times fewer parameters. The combination of a transformer network, CNN and Hungarian Loss culminates in a lane detection framework that is precise, fast, and tiny.

4.2.3 Segmentation

DETR [16] can be naturally extended for panoptic segmentation tasks and achieve competitive results by appending a mask head on the decoder. Wang *et al*. [155] proposed Max-DeepLab to directly predict panoptic segmentation results with a mask transformer, without involving surrogate sub-tasks such as box detection. Similar to DETR, Max-DeepLab streamlines the panoptic segmentation tasks in an end-to-end fashion and directly predicts a set of non-overlapping masks and corresponding labels. Model training is performed using a panoptic quality (PQ) style loss, but unlike prior methods that stack a transformer on top of a CNN backbone, Max-DeepLab adopts a dual-path framework that facilitates combining the CNN and transformer.

VisTR, a transformer-based video instance segmentation model, was proposed by Wang *et al*. [159] to produce instance prediction results from a sequence of input images. A strategy for matching instance sequence is proposed to assign the predictions with ground truths. In order to obtain the mask sequence for each instance, VisTR utilizes the instance sequence segmentation module to accumulate the mask features from multiple frames and segment the mask sequence with a 3D CNN.

Cell-DETR [116], based on the DETR panoptic segmentation model, is an attempt to use transformer for cell instance segmentation. It adds skip connections that bridge features between the backbone CNN and the CNN decoder in the segmentation head in order to enhance feature fusion. Cell-DETR achieves state-of-the-art performance for cell instance segmentation from microscopy imagery.

Zheng *et al*. [190] proposed a transformer-based semantic segmentation network (SETR). SETR utilizes an encoder similar to ViT [36] as the encoder to extract features from an input image. A multi-level feature aggregation module is adopted for performing pixel-wise segmentation.

A number of other works exploring transformer architecture for point cloud learning have also emerged recently [38, 51, 186]. For example, Guo *et al*. [51] proposed a novel framework that replaces the original self-attention module with a more suitable offset-attention module, which includes implicit Laplace operator and normalization refinement. In addition, Zhao *et al*. [186] designed a novel transformer architecture called Point Transformer. The proposed self-attention layer is invariant to the permutation of the point set, making it suitable for point set processing tasks. Point Transformer shows strong performance for semantic segmentation task from 3D point clouds.

4.2.4 Pose Estimation

Human and hand pose estimation is a topic that has attracted significant interest from the research community. Articulated pose estimation is akin to a structured prediction task, aiming to predict the joint coordinates or mesh vertices from input RGB/D images. Here we discuss some methods [67, 68, 92, 168] that explore how to utilize transformer for modeling the global structure information of human pose.

Huang *et al*. [67] proposed a transformer based network for 3D hand pose estimation from point sets. The encoder first utilizes a PointNet [119] to extract point-wise features from input point clouds and then adopts standard multi-head self-attention module to produce embeddings. In order to expose more global pose-related information to the decoder, a feature extractor such as PointNet++ [120] is used to extract hand joint-wise features, which are then fed into the decoder as positional encodings.

Similarly, Huang *et al*. [68] proposed HOT-Net (short for hand-object transformer network) for 3D hand-object pose estimation. Unlike the preceding method which employs transformer to directly predict 3D hand pose from input point clouds, HOT-Net uses a ResNet to generate initial 2D hand-object pose and then feeds it into a transformer to predict the 3D hand-object pose. A spectral graph convolution network is therefore used to extract input embeddings for the encoder.

Lin *et al*. [92] proposed a mesh transformer (METRO) for predicting 3D human pose and mesh from a single RGB image. METRO extracts image features via a CNN and then perform position encoding by concatenating a template human mesh to the image feature. A multi-layer transformer encoder with progressive dimensionality reduction is proposed to gradually reduce the embedding dimensions and finally produce 3D coordinates of human joint and mesh vertices. To encourage the learning of non-local relationships between human joints, METRO randomly mask some input queries during training.

4.2.5 Discussions

As discussed in the preceding sections, transformers have shown strong performance on several high-level tasks, including detection, segmentation and pose estimation. The key issues that need to be resolved before transformer can be adopted for high-level tasks relate to input embedding, position encoding, and prediction loss. Some methods propose improving the self-attention module from different perspectives, for example, deformable attention [193], adaptive clustering [189] and point transformer [186]. Nevertheless, exploration into the use of transformers for high-level vision tasks is still in the preliminary stages and so further research may prove beneficial. For example, is it necessary to use feature extraction modules such as CNN and PointNet before transformer for final reasons? How can vision transformer be fully leveraged using large scale pre-training datasets as BERT and GPT-3 do in the NLP field? And is it possible to pre-train a single transformer model and fine-tune it for different downstream tasks with only a few epochs of fine-tuning? We hope more further research effort is conducted into exploring more powerful transformers for high-level vision.

4.3. Low-level Vision

Few works apply transformers on low-level vision fields, such as image super-resolution and generation. These tasks often take images as outputs (*e.g.*, high-resolution or denoised images), which is more challenging than high-level vision tasks such as classification, segmentation, and detection, whose outputs are labels or boxes.

Parmar *et al*. [113] proposed Image Transformer, taking the first step toward generalizing the transformer model to formulate image translation and generation tasks. Image Transformer consists of two parts: an encoder for extracting image representation and a decoder to generate pixels. For each pixel with value $0-255$, a $256 \times d$ dimensional embedding is learned for encoding each value into a d dimensional vector, which is fed into the encoder as input. The encoder and decoder adopt the same architecture as that in [152]. Figure 8 shows the structure of each layer

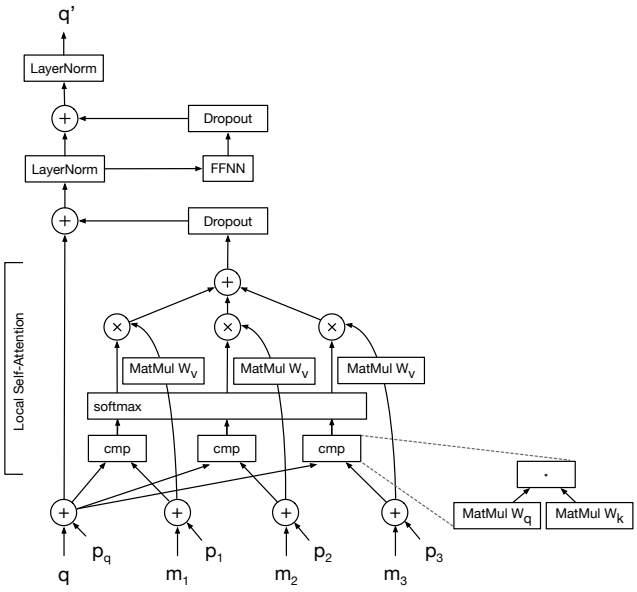

Figure 8: Diagram of a layer in the decoder of Image Transformer (figure from [113]). Each output pixel q' is generated by previously generated pixels $m_1, m_2, ...$ and the corresponding input pixel q.

in the decoder. Each output pixel q' is generated by calculating self-attention between the input pixel q and previously generated pixels $m_1, m_2, ...$ with position embedding $p_1, p_2, ...$ For image-conditioned generation, such as super-resolution and inpainting, an encoder-decoder architecture is used, where the encoder's input is the low-resolution or corrupted images. For unconditional and class-conditional generation (*i.e.*, noise to image), only the decoder is used for inputting noise vectors. Because the decoder's input is the previously generated pixels (involving high computation cost when producing high-resolution images), a local self-attention scheme is proposed. This scheme uses only the closest generated pixels as input for the decoder, enabling Image Transformer to achieve performance on par with CNN-based models for image generation and translation tasks, demonstrating the effectiveness of transformer-based models on low-level vision tasks.

A number of recent works eschew using each pixel as the input for transformer models and instead use patches (set of pixels) as input. For example, Yang *et al.* [167] proposed Texture Transformer Network for Image Super-Resolution (TTSR), using the transformer architecture in the reference-based image super-resolution problem. It aims to transfer relevant textures from reference images to low-resolution images. Taking a low-resolution image and reference image as the query Q and key K, respectively, the relevance $r_{i,j}$ is calculated between each patch q_i in Q and k_i in K as:

$$r_{i,j} = \left\langle \frac{q_i}{\|q_i\|}, \frac{k_i}{\|k_i\|} \right\rangle. \tag{12}$$

A hard-attention module is proposed to select high-resolution features V according to the reference image, so that the low-resolution image can be matched by using the relevance. The hard-attention map is calculated as:

$$h_i = \arg\max_j r_{i,j} \tag{13}$$

The most relevant reference patch is $t_i = v_{h_i}$, where t_i in T is the transferred features. A soft-attention module is then used to transfer V to the low-resolution feature F. The soft-attention is calculated as:

$$s_i = \max_j r_{i,j}. \tag{14}$$

The equation to transfer the high-resolution texture images to the low-resolution images can be formulated as:

$$F_{out} = F + \text{Conv}(\text{Concat}(F, T)) \odot S. \tag{15}$$

Here, F_{out} and F denote the output and input features of the low-resolution image,respectively; S is the soft-attention; and T is the transferred features from the high-resolution texture image. By leveraging the transformer-based architecture, TTSR can successfully transfer texture information from high-resolution reference images to low-resolution images in super-resolution tasks.

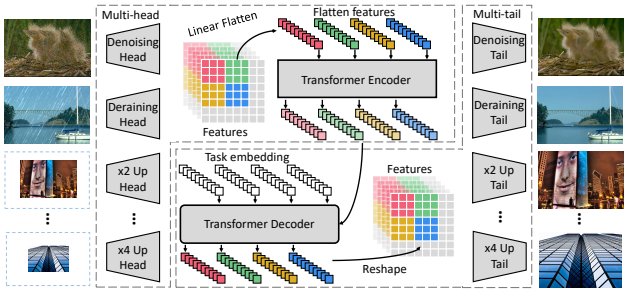

Figure 9: Diagram of the IPT architecture (figure from [20]). The multi-head, multi-tail structure handles different image processing tasks.

Different from the preceding methods that use transformer models on single tasks, Chen *et al.* [20] proposed Image Processing Transformer (IPT), which fully utilizes the advantages of transformers by using large pre-training datasets. It achieves state-of-the-art performance in several image processing tasks, including super-resolution, denoising, and deraining. As shown in Figure 9, IPT consists of multiple heads, an encoder, a decoder, and multiple

tails. The multi-head, multi-tail structure and task embeddings are introduced for different image processing tasks. The features are divided into patches, which are fed into the encoder-decoder architecture. Following this, the outputs are reshaped to features with the same size. Given the advantages of pre-training transformer models on large datasets, IPT uses the ImageNet dataset for pre-training. Specifically, images from this dataset are degraded by manually adding noise, rain streaks, or downsampling in order to generate corrupted images. The degraded images are used as inputs for IPT, while the original images are used as the optimization goal of the outputs. A self-supervised method is also introduced to enhance the generalization ability of the IPT model. Once the model is trained, it is fine-tuned on each task by using the corresponding head, tail, and task embedding. IPT largely achieves performance improvements on image processing tasks (*e.g.*, 2 dB in image denoising tasks), demonstrating the huge potential of applying transformer-based models to the field of low-level vision.

Besides single image generation, Wang *et al*. [158] proposed SceneFormer to utilize transformer in 3D indoor scene generation. By treating a scene as a sequence of objects, the transformer decoder can be used to predict series of objects and their location, category, and size. This has enabled SceneFormer to outperform conventional CNN-based methods in user studies.

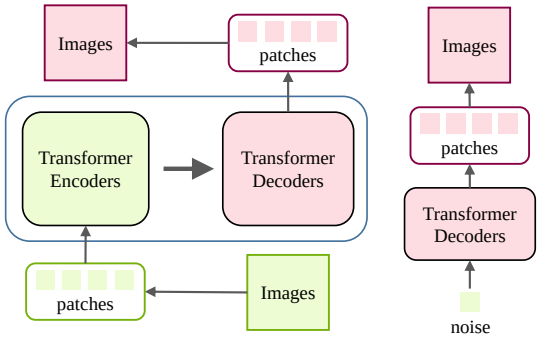

Figure 10: A generic framework for transformer in image processing and generation.

In conclusion, different to classification and detection tasks, the outputs of image generation and processing are images. Figure 10 illustrates using transformers in low-level vision. By taking images as a sequence of pixels or patches, the transformer encoder uses the sequence as input, allowing the transformer decoder to successfully produce desired images. A meaningful direction for future research would be designing a suitable architecture for different image processing tasks.

4.4. Video Processing

Transformer performs surprisingly well on sequence-based tasks and especially on NLP tasks. In computer vision (specifically, video tasks), spatial and temporal dimension information is favored, giving rise to the application of transformer in a number of video tasks, such as frame synthesis [99], action recognition [49], and video retrieval [96].

4.4.1 High-level Video Processing

Image Generation. Image generation tasks refer to generate images from a natural image dataset. The parallelizable architectures are attractive because of their efficiency on predicting the pixel blocks. Based on the likelihood generative models, Parmar *et al.* [113] used the transformer to generate images. The local attention module is integrated in the image transformer to predict the consecutive pixel blocks.

Activity Recognition. Activity recognition tasks involve identifying the activities of a human in a group of people. Former methods applied to such tasks have been based on the location of individual actors. Gavrilyuk *et al.* proposed an actor-transformer [48] architecture to learn the representation, using the static and dynamic representations generated by the 2D and 3D networks as input. The output of the transformer is the predicted activity.

Human Action Recognition. Video human action tasks, as the name suggests, involves identifying and localizing human actions in videos. Context (such as other people and objects) plays a critical role in recognizing human actions. Rohit *et al.* proposed the action transformer [49] to model the underlying relationship between the human of interest and the surrounding context. Specifically, the I3D [17] is used as the backbone to extract high-level feature maps. The features extracted (using RoI pooling) from intermediate feature maps are viewed as the query (Q), while the key (K) and values (V) are calculated from the intermediate features. A self-attention mechanism is applied to the three components, and it outputs the classification and regressions predictions. Lohit *et al.* [101] proposed an interpretable differentiable module, named temporal transformer network, to reduce the intra-class variance and increase the inter-class variance. In addition, Fayyaz and Gall proposed a temporal transformer [42] to perform action recognition tasks under weakly supervised settings.

Face Alignment. Video-based face alignment tasks aim to localize facial landmarks. Overall performance of such tasks is heavily dependent on temporal dependency and spatial information. However, former methods have failed to capture both the temporal information across consecutive frames and the complementary spatial information on a still frame. To address this issue, Liu *et al.* [96] proposed two-stream transformer networks, which separately learn the

temporal and spatial features. This approach jointly optimizes two streams in an end-to-end manner and weights the features in order to obtain the final predictions.

Video Retrieval. The key to content-based video retrieval is to find the similarity between videos. Leveraging only the image-level of video-level features to overcome the associated challenges, Shao *et al.* [135] suggested using the transformer to model the long-range semantic dependency. They also introduced the supervised contrastive learning strategy to perform hard negative mining. The results of using this approach on benchmark datasets demonstrate its performance and speed advantages. In addition, Gabeur *et al.* [47] presented a multi-modal transformer to learn different cross-modal cues in order to represent videos.

Video Object Detection. To detect objects in a video, both global and local information is required. Chen *et al.* introduced the memory enhanced global-local aggregation (MEGA) [22] to capture more content. The representative features enhance the overall performance and address the *ineffective* and *insufficient* problems. Furthermore, Yin *et al.* [171] proposed a spatiotemporal transformer to aggregate spatial and temporal information. Together with another spatial feature encoding component, these two components perform well on 3D video object detection tasks.

Multi-task Learning. Untrimmed video usually contains many frames that are irrelevant to the target tasks. It is therefore crucial to mine the relevant information and discard the redundant information. To extract such information, Seong *et al.* proposed the video multi-task transformer network [133], which handles multi-task learning on untrimmed videos. For the CoVieW dataset, the tasks are scene recognition, action recognition and importance score prediction. Two pre-trained networks on ImageNet and Places365 extract the scene features and object features. The multi-task transformers are stacked to implement feature fusion, leveraging the class conversion matrix (CCM).

4.4.2 Low-level Video Processing

Frame/Video Synthesis. Frame synthesis tasks involve synthesizing the frames between two consecutive frames or after a frame sequence while video synthesis tasks involve synthesizing a video. Liu *et al.* proposed the ConvTransformer [99], which is comprised of five components: feature embedding, position encoding, encoder, query decoder, and the synthesis feed-forward network. Compared with LSTM based works, the ConvTransformer achieves superior results with a more parallelizable architecture. Another transformer-based approach was proposed by Schatz *et al.* [132], which uses a recurrent transformer network to synthetize human actions from novel views.

Video Inpainting. Video inpainting tasks involve completing any missing regions within a frame. This is challenging, as it requires information along the spatial and temporal dimensions to be merged. Zeng *et al.* proposed a spatial-temporal transformer network [178], which uses all the input frames as input and fills them in parallel. The spatial-temporal adversarial loss is used to optimize the transformer network.

4.5. Self-attention for Computer Vision

The preceding sections reviewed methods that use a transformer architecture for visual tasks. We can conclude that self-attention is the key part of transformer. In this section, we delve deep into the models based on self-attention designed for challenging tasks in computer vision. Such tasks include semantic segmentation, instance segmentation, object detection, keypoint detection, and depth estimation. We start with formulating the algorithm of self-attention in Section 4.5.1, and then summarize the existing applications using self-attention for computer vision in Section 4.5.2.

4.5.1 General Formulation of Self-attention

The self-attention module [152] for machine translation computes the responses at one position in a sequence by attending to all positions and summing them up accordingly based on weighting in an embedding space. This can be viewed as a form of non-local filtering operations [157, 12] applicable to computer vision. We follow the convention [157] to formulate the self-attention module. Given an input signal (*e.g.*, image, sequence, video and feature) $X \in \mathbb{R}^{n \times c}$, where $n = h \times w$ (indicating the number of pixels in feature) and c is the number of channels, the output signal is generated as:

$$y_i = \frac{1}{C(x_i)} \sum_{\forall j} f(x_i, x_j) g(x_j), \tag{16}$$

Here, $x_i \in \mathbb{R}^{1 \times c}$ and $y_i \in \mathbb{R}^{1 \times c}$ indicate the i^{th} position (*e.g.*, space, time and spacetime) of the input signal X and output signal Y, respectively. Subscript j is the index that enumerates all positions, and a pairwise function $f(\cdot)$ computes a representing relationship (such as affinity) between i and all j. The function $g(\cdot)$ computes a representation of the input signal at position j, and the response is normalized by a factor $C(x_i)$.

Note that there are many choices for the pairwise function $f(\cdot)$. For example, a simple extension of the Gaussian function could be used to compute the similarity in an embedding space. As such, the function $f(\cdot)$ can be formulated as:

$$f(x_i, x_j) = e^{\theta(x_i)\phi(x_j)^T} \tag{17}$$

where $\theta(\cdot)$ and $\phi(\cdot)$ can be any embedding layers. If we consider the $\theta(\cdot), \phi(\cdot), g(\cdot)$ in the form of linear embedding: $\theta(X) = XW_\theta$, $\phi(X) = XW_\phi$, $g(X) = XW_g$ where

$W_\theta \in \mathbb{R}^{c\times d_k}, W_\phi \in \mathbb{R}^{c\times d_k}, W_g \in \mathbb{R}^{c\times d_v}$, and set the normalization factor as $C(x_i) = \sum_{\forall j} f(x_i, x_j)$, the Eq. 16 can be rewritten as:

$$y_i = \frac{e^{x_i w_{\theta,i} w_{\phi,j}^T x_j^T}}{\sum_j e^{x_i w_{\theta,i} w_{\phi,j}^T x_j^T}} x_j w_{g,j}, \tag{18}$$

where $w_{\theta,i} \in \mathbb{R}^{c\times 1}$ is the i^{th} row of the weight matrix W_θ. For a given index i, $\frac{1}{C(x_i)} f(x_i, x_j)$ becomes the softmax output along the dimension j. The formulation can be further rewritten as:

$$Y = \text{softmax}(XW_\theta W_\phi^T X)g(X), \tag{19}$$

where $Y \in \mathbb{R}^{n\times c}$ is the output signal of the same size as X. Compared with the query, key and value representations $Q = XW_q, K = XW_k, V = XW_v$ from the translation module, once $W_q = W_\theta, W_k = W_\phi, W_v = W_g$, Eq. 19 can be formulated as:

$$Y = \text{softmax}(QK^T)V = \text{Attention}(Q, K, V), \tag{20}$$

The self-attention module [152] proposed for machine translation is, to some extent, the same as the preceding non-local filtering operations proposed for computer vision.

Generally, the final output signal of the self-attention module for computer vision will be wrapped as:

$$Z = YW_Z + X \tag{21}$$

where Y is generated through Eq. 19. If W_Z is initialized as zero, this self-attention module can be inserted into any existing model without breaking its initial behavior.

4.5.2 Applications on Visual Tasks

The self-attention module is considered a building block of CNN architectures, which have low scaling properties concerning the large receptive fields. This building block is always used on top of the networks to capture long-range interactions and enhance high-level semantic features for computer vision tasks. In what follows, we review the proposed methods, which are based on self-attention, for image based tasks such as image classification, semantic segmentation, and object detection.

Image Classification. Trainable attention for classification consists of two main streams: hard attention [4, 105, 166] regarding the use of an image region, and soft attention [154, 71, 53, 125] generating non-rigid feature maps. Ba *et al*. [4] first proposed the term "visual attention" for image classification tasks, and used attention to select relevant regions and locations within the input image. This can also reduce the computational complexity of the proposed model regarding the size of the input image. For medical image classification, AG-CNN [50] was proposed to crop a sub-region from a global image by the attention heat map. And instead of using hard attention and recalibrating the crop of feature maps, SENet [65] was proposed to reweight the channel-wise responses of the convolutional features using soft self-attention. Jetley *et al*. [71] used attention maps generated by corresponding estimators to reweight intermediate features in DNNs. In addition, Han *et al*. [53] utilized the attribute-aware attention to enhance the representation of CNNs.

Semantic Segmentation. PSANet [187], OCNet [172], DANet [46] and CFNet [182] are the pioneering works to propose using the self-attention module in semantic segmentation tasks. These works consider and augment the relationship and similarity [181, 89, 56, 108, 160, 88] between the contextual pixels. DANet [46] simultaneously leverages the self-attention module on spatial and channel dimensions, whereas A^2Net [23] groups the pixels into a set of regions, and then augments the pixel representations by aggregating the region representations with the generated attention weights. DGCNet [184] employs a dual graph CNN to model coordinate space similarity and feature space similarity in a single framework. To improve the efficiency of the self-attention module for semantic segmentation, several works [174, 70, 69, 90, 81] have been proposed, aiming to alleviate the huge amount of parameters brought by calculating pixel similarities. For example, CGNL [174] applies the Taylor series of the RBF kernel function to approximate the pixel similarities. CCNet [70] approximates the original self-attention scheme via two consecutive criss-cross attention modules. In addition, ISSA [69] factorizes the dense affinity matrix as the product of two sparse affinity matrices. There are other related works using attention based graph reasoning modules [91, 24, 90] to enhance both the local and global representations.

Object Detection. Ramachandran *et al*. [125] proposes an attention-based layer and swapped the conventional convolution layers to build a fully attentional detector that outperforms the typical RetinaNet [94] on COCO benchmark [95]. GCNet [14] assumes that the global contexts modeled by non-local operations are almost the same for different query positions within an image, and unifies the simplified formulation and SENet [65] into a general framework for global context modeling [87, 63, 41, 114]. Vo *et al*. [153] designs a bidirectional operation to gather and distribute information from a query position to all possible positions. Zhang *et al*. [180] suggests that previous methods fail to interact with cross-scale features, and proposes Feature Pyramid Transformer, based on the self-attention module, to fully exploit interactions across both space and scales.

Conventional detection methods usually exploit a single visual representation (*e.g*., bounding box and corner point) for predicting the final results. Hu *et al*. [64] pro-

poses a relation module based on self-attention to process a set of objects simultaneously through interaction between their appearance features. Cheng *et al*. [28] proposes RelationNet++ with the bridging visual representations (BVR) module to combine different heterogeneous representations into a single one similar to that in the self-attention module. Specifically, the master representation is treated as the query input and the auxiliary representations are regarded as the key input. The enhanced feature can therefore bridge the information from auxiliary representations and benefit final detection results.

Other Vision Tasks. Zhang *et al*. [183] proposes a resolution-wise attention module to learn enhanced feature maps when training multi-resolution networks to obtain accurate human keypoint locations for pose estimation task. Furthermore, Chang *et al*. [18] uses an attention-mechanism based feature fusion block to improve the accuracy of the human keypoint detection model.

To explore more generalized contextual information for improving the self-supervised monocular trained depth estimation, Johnston *et al*. [74] directly leverages self-attention module. Chen *et al*. [25] also proposes an attention-based aggregation network to capture context information that differs in diverse scenes for depth estimation. And Aich *et al*. [1] proposes bidirectional attention modules that utilize the forward and backward attention operations for better results of monocular depth estimation.

4.6. Efficient Transformer

Although transformer models have achieved success in various tasks, their high requirements for memory and computing resources block their implementation on resource-limited devices such as mobile phones. In this section, we review the researches carried out into compressing and accelerating transformer models for efficient implementation. This includes including network pruning, low-rank decomposition, knowledge distillation, network quantization, and compact architecture design. Table 5 lists some representative works for compressing transformer-based models.

Table 5: List of representative compressed transformer-based models. The data of the Table is from [121].

Models	Compress Type	#Layer	Params	Speed Up
BERT$_{BASE}$ [34]	Baseline	12	110M	×1
ALBERT [82]	Decomposition	12	12M	×5.6
BERT-of-Theseus [165]	Architecture design	6	66M	×1.94
Q-BERT [136]	Quantization	12	-	-
Q8BERT [176]		12		
TinyBERT [73]	Distillation	4	14.5M	×9.4
DistilBERT [131]		6	6.6m	×1.63
BERT-PKD [142]		3∼6	45.7∼67M	×3.73∼1.64
MobileBERT [144]		24	25.3M	×4.0
PD [149]		6	67.5M	×2.0

4.6.1 Pruning and Decomposition

In transformer based pre-trained models (*e.g*., BERT), multiple attention operations are performed in parallel to independently model the relationship between different tokens [152, 34]. However, specific tasks do not require all heads to be used. For example, Michel *et al*. [103] presented empirical evidence that a large percentage of attention heads can be removed at test time without impacting performance significantly. The number of heads required varies across different layers — some layers may even require only one head. Considering the redundancy on attention heads, importance scores are defined to estimate the influence of each head on the final output in [103], and unimportant heads can be removed for efficient deployment. Dalvi *et al*. [117] analyzed the redundancy in pre-trained transformer models from two perspectives: general redundancy and task-specific redundancy. Following the lottery ticket hypothesis [44], Prasanna *et al*. [117] analyzed the lotteries in BERT and showed that good sub-networks also exist in transformer-based models, reducing both the FFN layers and attention heads in order to achieve high compression rates.

In addition to the width of transformer models, the depth (*i.e*., the number of layers) can also be reduced to accelerate the inference process [39]. Differing from the concept that different attention heads in transformer models can be computed in parallel, different layers have to be calculated sequentially because the input of the next layer depends on the output of previous layers. Fan *et al*. [39] proposed a layer-wisely dropping strategy to regularize the training of models, and then the whole layers are removed together at the test phase. Given that the resources available in different devices may vary, Hou *et al*. [62] proposed to adaptively reduce the width and depth of pre-defined transformer models. This approach obtains multiple models with different sizes simultaneously, and shares important attention heads and neurons across different sub-networks via a rewiring mechanism.

Beyond the pruning methods that directly discard modules in transformer models, matrix decomposition aims to approximate the large matrices with multiple small matrices based on the low-rank assumption. For example, Wang *et al*. [161] decomposed the standard matrix multiplication in transformer models, improving the inference efficiency.

4.6.2 Knowledge Distillation

Knowledge distillation aims to train student networks by transferring knowledge from large teacher networks [59, 13, 3]. Compared with teacher networks, student networks

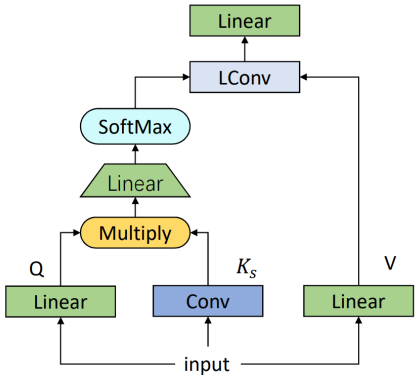

Figure 11: Span-based dynamic convolution (figure from [72]). LConv denotes the light-weight depth-wise convolution.

usually have thinner and shallower architectures, which are easier to be deployed on resource-limited resources. Both the output and intermediate features of neural networks can also be used to transfer effective information from teachers to students. Focused on transformer models, Mukherjee *et al*. [106] used the pre-trained BERT [34] as a teacher to guide the training of small models, leveraging large amounts of unlabeled data. Wang *et al*. [156] train the student networks to mimic the output of self-attention layers in the pre-trained teacher models. The dot-product between values is introduced as a new form of knowledge for guiding students. A teacher's assistant [104] is also introduced in [156], reducing the gap between large pre-trained transformer models and compact student networks, thereby facilitating the mimicking takes. Due to the various types of layers in the transformer model (*i.e*., self-attention layer, embedding layer, and prediction layers), Jiao *et al*. [73] design different objective functions to transfer knowledge from teachers to students. For example, the outputs of student models' embedding layers imitate those of teachers via MSE losses. A learnable linear transformation is also imposed to map different features into the same space. For the output of prediction layers, KL-divergence is adopted to measure the difference between different models.

4.6.3 Quantization

Quantization aims to reduce the number of bits needed to represent network weight or intermediate features [151, 170]. Quantization methods for general neural networks have been discussed at length and achieve performance on par with the original networks [112, 45, 7]. Recently, there has been growing interest in how to specially quantize transformer models [10, 40]. For example, Shridhar *et al*. [137] suggested embedding the input into binary high-dimensional vectors, and then using the binary input representation to train the binary neural networks. Cheong *et al*. [26] represented the weights in the transformer models by low-bit (*e.g*., 4-bit) representation. Zhao *et al*. [188] empirically investigated various quantization methods and showed that k-means quantization has a huge development potential. Aimed at machine translation tasks, Prato *et al*. [118] proposed a fully quantized transformer, which, as the paper claims, is the first 8-bit model not to suffer any loss in translation quality.

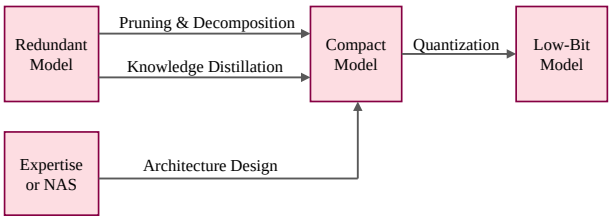

Figure 12: Different methods for compressing transformers.

4.6.4 Compact Architecture Design

Beyond compressing pre-defined transformer models into smaller ones, some works attempt to design compact models directly [164, 72]. Jiang *et al*. [72] simplified the calculation of self-attention by proposing a new module — called span-based dynamic convolution — that combine the fully-connected layers and the convolutional layers, as shown in Figure 11. The local dependence between representation from different tokens is calculated by convolution operations, which is significantly more efficient than the dense fully-connected layers in standard transformers. Depth-wise convolution is also used to further reduce the computing cost. Interesting "hamburger" layers are proposed in [2], using matrix decomposition to substitute the original self-attention layers.Compared with standard self-attention operations, matrix decomposition can be calculated more efficiently while clearly reflecting the dependence between different tokens. The design of efficient transformer architectures can also be automated with neural architecture search (NAS) [52, 138], which automatically searches how to combine different components.

The self-attention operation in transformer models calculates the dot product between representations from different input tokens in a given sequence (patches in image recognition tasks [37]), whose complexity is $O(N)$, where N is the length of the sequence. Recently, there has been a targeted focus to reduce the complexity to $O(N)$ in large methods so that transformer models can scale to long sequences. For example, Katharopoulos *et al*. [76] approximated self-attention as a linear dot-product of kernel feature maps and revealed the relationship between tokens via RNNs. Zaheer *et al*. [177] considered each token as a vertex

in a graph and defined the inner product calculation between two tokens as an edge. Inspired by graph theories [139, 30], various sparse graph are combined to approximate the dense graph in transformer models, and can achieve $O(N)$ complexity. From a theoretical perspective, Yun *et al*. [175] proved that a sparse transformer with $O(N)$ complexity is sufficient to reflect any kind of relationship between tokens and can make universal approximation, providing theoretical guarantees for further research about transformer with $O(N)$ complexity.

Discussion. The preceding methods take different approaches in how they attempt to identify redundancy in transformer models (see Figure 12). Pruning and decomposition methods usually require pre-defined models with redundancy. Specifically, pruning focuses on reducing the number of components (*e.g*., layers, heads) in transformer models while decomposition represents an original matrix with multiple small matrices. Compact models also can be directly designed either manually (requiring sufficient expertise) or automatically (*e.g*., via NAS). The obtained compact models can be further represented with low-bits via quantization methods for efficient deployment on resource-limited devices.

5. Conclusions and Discussions

Transformer is becoming a hot topic in the field of computer vision due to its competitive performance and tremendous potential compared with CNNs. To discover and utilize the power of transformer, as summarized in this survey, a number of methods have been proposed in recent years. These methods show excellent performance on a wide range of visual tasks, including backbone, high/mid-level vision, low-level vision, and video processing. Nevertheless, the potential of transformer for computer vision has not yet been fully explored, meaning that several challenges still need to be resolved. In this section, we discuss these challenges and provide insights on the future prospects.

5.1. Challenges

Although researchers have proposed many transformer-based models to tackle computer vision tasks, these works are only the first steps in this field and still have much room for improvement. For example, the transformer architecture in ViT [36] follows the standard transformer for NLP [152], but an improved version specifically designed for CV remains to be explored. Moreover, it is necessary to apply transformer to more tasks other than those mentioned earlier.

The generalization and robustness of transformers for computer vision are also challenging. Compared with CNNs, pure transformers lack some inductive biases and rely heavily on massive datasets for large-scale training [36]. Consequently, the quality of data has a significant influence on the generalization and robustness of transformers. Although ViT shows exceptional performance on downstream image classification tasks such as CIFAR [79] and VTAB [179], directly applying the ViT backbone on object detection has failed to achieve better results than CNNs [8]. There is still a long way to go in order to better generalize pre-trained transformers on more generalized visual tasks.

Although numerous works have explained the use of transformers in NLP [134, 162], it remains a challenging subject to clearly explain why transformer works well on visual tasks. The inductive biases, including translation equivariance and locality, are attributed to CNN's success, but transformer lacks any inductive bias. The current literature usually analyzes the effect in an intuitive way [36, 19]. For example, Dosovitskiy *et al*. [36] claim that large-scale training can surpass inductive bias. Position embeddings are added into image patches to retain positional information, which is important in computer vision tasks. Inspired by the heavy parameter usage in transformers, over-parameterization [100, 107] may be a potential point to the interpretability of visual transformers.

Last but not least, developing efficient transformer models for CV remains an open problem. Transformer models are usually huge and computationally expensive. For example, the base ViT model [36] requires 18 billion FLOPs to process an image. In contrast, the lightweight CNN model GhostNet [54, 55] can achieve similar performance with only about 600 million FLOPs. Although several methods have been proposed to compress transformer, they remain highly complex. And these methods, which were originally designed for NLP, may not be suitable for CV. Consequently, efficient transformer models are urgently needed so that visual transformer can be deployed on resource-limited devices.

5.2. Future Prospects

In order to drive the development of visual transformers, we provide several potential directions for future study.

One direction is the effectiveness and the efficiency of transformers in computer vision. The goal is to develop highly effective and efficient visual transformers; specifically, transformers with high performance and low resource cost. The performance determines whether the model can be applied on real-world applications, while the resource cost influences the deployment on devices. The effectiveness is usually correlated with the efficiency, so determining how to achieve a better balance between them is a meaningful topic for future study.

Most of the existing visual transformer models are designed to handle only a single task. Many NLP models such as GPT-3 [11] have demonstrated how transformer can deal with multiple tasks in one model. IPT [20] in the CV field is

also able to process multiple low-level vision tasks, such as super-resolution, image denoising, and deraining. We believe that more tasks can be involved in only one model. Unifying all visual tasks and even other tasks in one transformer (i.e., a grand unified model) is an exciting topic.

There have been various types of neural networks, such as CNN, RNN, and transformer. In the CV field, CNNs used to be the mainstream choice [57, 146], but now transformer is becoming popular. CNNs can capture inductive biases such as translation equivariance and locality, whereas ViT uses large-scale training to surpass inductive bias [36]. From the evidence currently available [36], CNNs perform well on small datasets, whereas transformers perform better on large datasets. The question for the future is whether to use CNN or transformer.

By training with large datasets, transformers can achieve state-of-the-art performance on both NLP [11, 34] and CV benchmarks [36]. It is possible that neural networks need big data rather than inductive bias. In closing, we leave you with a question: Can transformer obtains satisfactory results with a very simple computational paradigm (e.g., with only fully connected layers) and massive data training?

References

[1] Shubhra Aich, Jean Marie Uwabeza Vianney, Md Amirul Islam, Mannat Kaur, and Bingbing Liu. Bidirectional attention network for monocular depth estimation. *arXiv preprint arXiv:2009.00743*, 2020. 16

[2] Anonymous. Is attention better than matrix decomposition? In *Submitted to International Conference on Learning Representations*, 2021. under review. 17

[3] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27:2654–2662, 2014. 16

[4] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. In *International Conference on Learning Representations*, 2014. 15

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 1

[7] Yu Bai, Yu-Xiang Wang, and Edo Liberty. Proxquant: Quantized neural networks via proximal operators. *arXiv preprint arXiv:1810.00861*, 2018. 17

[8] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection. *arXiv preprint arXiv:2012.09958*, 2020. 8, 9, 10, 18

[9] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*, 2019. 5

[10] Aishwarya Bhandare, Vamsi Sripathi, Deepthi Karkada, Vivek Menon, Sun Choi, Kushal Datta, and Vikram Saletore. Efficient 8-bit quantization of transformer neural machine language translation model. *arXiv preprint arXiv:1906.00532*, 2019. 17

[11] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 1, 2, 5, 18, 19

[12] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 60–65, 2005. 14

[13] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006. 16

[14] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019. 15

[15] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872*, 2020. 2, 8, 10

[16] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 8, 10

[17] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 6299–6308, 2017. 13

[18] Yuan Chang, Zixuan Huang, and Qiwei Shen. The same size dilated attention network for keypoint detection. In *International Conference on Artificial Neural Networks*, pages 471–483, 2019. 16

[19] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. *arXiv preprint arXiv:2012.09838*, 2020. 18

[20] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *arXiv preprint arXiv:2012.00364*, 2020. 2, 12, 18

[21] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, pages 1691–1703. PMLR, 2020. 2, 6, 9

[22] Yihong Chen, Yue Cao, Han Hu, and Liwei Wang. Memory enhanced global-local aggregation for video object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10337–10346, 2020. 14

[23] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Aˆ 2-nets: Double attention networks. *Advances in neural information processing systems*, pages 352–361, 2018. 15

[24] Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Yan Shuicheng, Jiashi Feng, and Yannis Kalantidis. Graph-based global reasoning networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 433–442, 2019. 15

[25] Yuru Chen, Haitao Zhao, and Zhengwei Hu. Attention-based context aggregation network for monocular depth estimation. *arXiv preprint arXiv:1901.10137*, 2019. 16

[26] Robin Cheong and Robel Daniel. transformers. zip: Compressing transformers with pruning and quantization. Technical report, tech. rep., Stanford University, Stanford, California, 2019. 17

[27] Cheng Chi, Fangyun Wei, and Han Hu. Relationnet++: Bridging visual representations for object detection via transformer decoder. In *NeurIPS*, 2020. 8

[28] Cheng Chi, Fangyun Wei, and Han Hu. Relationnet++: Bridging visual representations for object detection via transformer decoder. *Advances in Neural Information Processing Systems*, 2020. 16

[29] Yung-Sung Chuang, Chi-Liang Liu, and Hung-Yi Lee. Speechbert: Cross-modal pre-trained language model for end-to-end spoken question answering. *arXiv preprint arXiv:1910.11559*, 2019. 6

[30] Fan Chung and Linyuan Lu. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 99(25):15879–15882, 2002. 18

[31] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 5

[32] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020. 5

[33] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. UP-DETR: unsupervised pre-training for object detection with transformers. *arXiv preprint arXiv:2011.09094*, 2020. 2, 9, 10

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, 2019. 1, 5, 16, 17, 19

[35] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13063–13075, 2019. 5

[36] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 6, 7, 9, 11, 18, 19

[37] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 17

[38] Nico Engel, Vasileios Belagiannis, and Klaus Dietmayer. Point transformer. *arXiv preprint arXiv:2011.00931*, 2020. 11

[39] Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*, 2019. 16

[40] Chaofei Fan. Quantized transformer. Technical report, Technical report, Stanford University, Stanford, California, 2019. URL https 17

[41] Qi Fan, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. Few-shot object detection with attention-rpn and multi-relation detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4013–4022, 2020. 15

[42] Mohsen Fayyaz and Jurgen Gall. Sct: Set constrained temporal transformer for set supervised action segmentation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 501–510, 2020. 13

[43] Robert B Fisher. Cvonline: The evolving, distributed, non-proprietary, on-line compendium of computer vision. *Retrieved January 28, 2006 from http://homepages. inf. ed. ac. uk/rbf/CVonline*, 2008. 2

[44] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. 16

[45] Joshua Fromm, Meghan Cowan, Matthai Philipose, Luis Ceze, and Shwetak Patel. Riptide: Fast end-to-end binarized neural networks. *Proceedings of Machine Learning and Systems*, 2:379–389, 2020. 17

[46] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3146–3154, 2019. 15

[47] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *European Conference on Computer Vision (ECCV)*, pages 214–229, 2020. 14

[48] Kirill Gavrilyuk, Ryan Sanford, Mehrsan Javan, and Cees GM Snoek. Actor-transformers for group activity recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 839–848, 2020. 13

[49] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2019. 13

[50] Qingji Guan, Yaping Huang, Zhun Zhong, Zhedong Zheng, Liang Zheng, and Yi Yang. Diagnose like a radiologist: Attention guided convolutional neural network for thorax disease classification. In *arXiv preprint arXiv:1801.09927*, 2018. 15

[51] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *arXiv preprint arXiv:2012.09688*, 2020. 11

[52] Yong Guo, Yin Zheng, Mingkui Tan, Qi Chen, Jian Chen, Peilin Zhao, and Junzhou Huang. Nat: Neural architecture transformer for accurate and compact architectures. In *Advances in Neural Information Processing Systems*, pages 737–748, 2019. 17

[53] Kai Han, Jianyuan Guo, Chao Zhang, and Mingjian Zhu. Attribute-aware attention model for fine-grained representation learning. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 2040–2048, 2018. 15

[54] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1580–1589, 2020. 18

[55] Kai Han, Yunhe Wang, Qiulin Zhang, Wei Zhang, Chunjing Xu, and Tong Zhang. Model rubik's cube: Twisting resolution, depth and width for tinynets. *Advances in Neural Information Processing Systems*, 33, 2020. 18

[56] Junjun He, Zhongying Deng, Lei Zhou, Yali Wang, and Yu Qiao. Adaptive pyramid context network for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7519–7528, 2019. 15

[57] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. pages 770–778, 2016. 2, 19

[58] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 4

[59] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 16

[60] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1

[61] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 5

[62] Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. Dynabert: Dynamic bert with adaptive width and depth. *Advances in Neural Information Processing Systems*, 33, 2020. 16

[63] Ting-I Hsieh, Yi-Chen Lo, Hwann-Tzong Chen, and Tyng-Luh Liu. One-shot object detection with co-attention and co-excitation. In *Advances in Neural Information Processing Systems*, pages 2725–2734, 2019. 15

[64] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3588–3597, 2018. 15

[65] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 15

[66] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*, 2019. 5

[67] Lin Huang, Jianchao Tan, Ji Liu, and Junsong Yuan. Hand-transformer: Non-autoregressive structured modeling for 3d hand pose estimation. In *European Conference on Computer Vision*, pages 17–33, 2020. 2, 8, 11

[68] Lin Huang, Jianchao Tan, Jingjing Meng, Ji Liu, and Junsong Yuan. Hot-net: Non-autoregressive transformer for 3d hand-object pose estimation. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3136–3145, 2020. 2, 8, 11

[69] Lang Huang, Yuhui Yuan, Jianyuan Guo, Chao Zhang, Xilin Chen, and Jingdong Wang. Interlaced sparse self-attention for semantic segmentation. *arXiv preprint arXiv:1907.12273*, 2019. 15

[70] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 603–612, 2019. 15

[71] Saumya Jetley, Nicholas A Lord, Namhoon Lee, and Philip HS Torr. Learn to pay attention. In *International Conference on Learning Representations*, 2018. 15

[72] Zi-Hang Jiang, Weihao Yu, Daquan Zhou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. Convbert: Improving bert with span-based dynamic convolution. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 17

[73] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Nov. 2020. 2, 16, 17

[74] Adrian Johnston and Gustavo Carneiro. Self-supervised monocular trained depth estimation using self-attention and discrete disparity volume. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4756–4765, 2020. 16

[75] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020. 5

[76] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020. 17

[77] Ruth Kimchi, Marlene Behrmann, and Carl R Olson. *Perceptual organization in vision: Behavioral and neural perspectives*. Psychology Press, 2003. 2

[78] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, 2020. 9

[79] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 18

[80] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012. 1

[81] Saumya Kumaar, Ye Lyu, Francesco Nex, and Michael Ying Yang. Cabinet: Efficient context aggregation network for low-latency semantic segmentation. *arXiv preprint arXiv:2011.00993*, 2020. 15

[82] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019. 16

[83] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[84] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020. 5

[85] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019. 5

[86] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019. 6

[87] Wei Li, Kai Liu, Lizhe Zhang, and Fei Cheng. Object detection based on an adaptive attention mechanism. *Scientific Reports*, pages 1–13, 2020. 15

[88] Xiangtai Li, Li Zhang, Ansheng You, Maoke Yang, Kuiyuan Yang, and Yunhai Tong. Global aggregation then local distribution in fully convolutional networks. *arXiv preprint arXiv:1909.07229*, 2019. 15

[89] Xia Li, Zhisheng Zhong, Jianlong Wu, Yibo Yang, Zhouchen Lin, and Hong Liu. Expectation-maximization attention networks for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9167–9176, 2019. 15

[90] Yin Li and Abhinav Gupta. Beyond grids: Learning graph representations for visual recognition. *Advances in Neural Information Processing Systems*, pages 9225–9235, 2018. 15

[91] Xiaodan Liang, Zhiting Hu, Hao Zhang, Liang Lin, and Eric P Xing. Symbolic graph reasoning meets convolutions. *Advances in Neural Information Processing Systems*, pages 1853–1863, 2018. 15

[92] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. *arXiv preprint arXiv:2012.09760*, 2020. 2, 8, 11

[93] Matthieu Lin, Chuming Li, Xingyuan Bu, Ming Sun, Chen Lin, Junjie Yan, Wanli Ouyang, and Zhidong Deng. Detr for pedestrian detection. *arXiv preprint arXiv:2012.06785*, 2020. 10

[94] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 15

[95] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755, 2014. 15

[96] Hao Liu, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Two-stream transformer networks for video-based face alignment. *IEEE transactions on pattern analysis and machine intelligence*, 40(11):2546–2554, 2017. 13

[97] Ruijin Liu, Zejian Yuan, Tie Liu, and Zhiliang Xiong. End-to-end lane shape prediction with transformers. In *WACV*, 2021. 8, 10

[98] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 5

[99] Zhouyong Liu, Shun Luo, Wubin Li, Jingben Lu, Yufan Wu, Chunguo Li, and Luxi Yang. Convtransformer: A convolutional transformer network for video frame synthesis. *arXiv preprint arXiv:2011.10185*, 2020. 13, 14

[100] Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. On the computational efficiency of training neural networks. *Advances in neural information processing systems*, 27:855–863, 2014. 18

[101] Suhas Lohit, Qiao Wang, and Pavan Turaga. Temporal transformer networks: Joint learning of invariant and discriminative time warping. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12426–12435, 2019. 13

[102] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2

[103] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, pages 14014–14024, 2019. 2, 16

[104] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5191–5198, 2020. 17

[105] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. *Advances in neural information processing systems*, pages 2204–2212, 2014. 15

[106] Subhabrata Mukherjee and Ahmed Hassan Awadallah. Xtremedistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2221–2234, 2020. 17

[107] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. *arXiv preprint arXiv:1805.12076*, 2018. 18

[108] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. At-

tention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999*, 2018. 15

[109] Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3d object detection with pointformer. *arXiv preprint arXiv:2012.11409*, 2020. 8

[110] Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3d object detection with pointformer. *arXiv preprint arXiv:2012.11409*, 2020. 8

[111] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, 2016. 1

[112] Eunhyeok Park and Sungjoo Yoo. Profit: A novel training method for sub-4-bit mobilenet models. In *European Conference on Computer Vision*, pages 430–446. Springer, 2020. 17

[113] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, pages 4055–4064. PMLR, 2018. 2, 11, 12, 13

[114] Hughes Perreault, Guillaume-Alexandre Bilodeau, Nicolas Saunier, and Maguelonne Héritier. Spotnet: Self-attention multi-task network for object detection. In *2020 17th Conference on Computer and Robot Vision (CRV)*, pages 230–237, 2020. 15

[115] Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019. 5

[116] Tim Prangemeier, Christoph Reich, and Heinz Koeppl. Attention-based transformers for instance segmentation of cells in microstructures. *arXiv preprint arXiv:2011.09763*, 2020. 11

[117] Sai Prasanna, Anna Rogers, and Anna Rumshisky. When bert plays the lottery, all tickets are winning. *arXiv preprint arXiv:2005.00561*, 2020. 16

[118] Gabriele Prato, Ella Charlaix, and Mehdi Rezagholizadeh. Fully quantized transformer for machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1–14, 2020. 2, 17

[119] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 11

[120] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30:5099–5108, 2017. 11

[121] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *arXiv preprint arXiv:2003.08271*, 2020. 5, 16

[122] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. 5

[123] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 5, 6

[124] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019. 5

[125] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Standalone self-attention in vision models. *arXiv preprint arXiv:1906.05909*, 2019. 15

[126] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2, 8

[127] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 9, 10

[128] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957. 1

[129] FRANK ROSENBLATT. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, CORNELL AERONAUTICAL LAB INC BUFFALO NY, 1961. 1

[130] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985. 1

[131] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 16

[132] Kara Marie Schatz, Erik Quintanilla, Shruti Vyas, and Yogesh Singh Rawat. A recurrent transformer network for novel view action synthesis. In *ECCV (27)*, pages 410–426, 2020. 14

[133] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Video multitask transformer network. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019. 14

[134] Sofia Serrano and Noah A Smith. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, 2019. 18

[135] Jie Shao, Xin Wen, Bingchen Zhao, and Xiangyang Xue. Temporal context aggregation for video retrieval with contrastive learning. 14

[136] Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Q-bert: Hessian based ultra low precision quantization of bert. In *AAAI*, pages 8815–8821, 2020. 16

[137] Kumar Shridhar, Harshil Jain, Akshat Agarwal, and Denis Kleyko. End to end binarized neural networks for text clas-

sification. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 29–34, 2020. 17

[138] David R So, Chen Liang, and Quoc V Le. The evolved transformer. *arXiv preprint arXiv:1901.11117*, 2019. 17

[139] Daniel A Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011. 18

[140] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019. 6

[141] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473, 2019. 6

[142] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*, 2019. 16

[143] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris Kitani. Rethinking transformer-based set prediction for object detection. *arXiv preprint arXiv:2011.10881*, 2020. 2, 8, 9, 10

[144] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*, 2020. 16

[145] Lucas Tabelini, Rodrigo Berriel, Thiago M Paixão, Claudine Badue, Alberto F De Souza, and Thiago Oliveira-Santos. Polylanenet: Lane estimation via deep polynomial regression. *arXiv preprint arXiv:2004.10924*, 2020. 10

[146] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 19

[147] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, pages 9627–9636, 2019. 10

[148] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020. 6, 7, 9

[149] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: The impact of student initialization on knowledge distillation. *arXiv preprint arXiv:1908.08962*, 2019. 16

[150] Shimon Ullman et al. *High-level vision: Object recognition and visual cognition*, volume 2. MIT press Cambridge, MA, 1996. 2

[151] Vincent Vanhoucke, Andrew Senior, and Mark Z Mao. Improving the speed of neural networks on cpus. 2011. 17

[152] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30:5998–6008, 2017. 1, 3, 4, 5, 11, 14, 15, 16, 18

[153] Xuan-Thuy Vo, Lihua Wen, Tien-Dat Tran, and Kang-Hyun Jo. Bidirectional non-local networks for object detection. In *International Conference on Computational Collective Intelligence*, pages 491–501, 2020. 15

[154] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang. Residual attention network for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2017. 15

[155] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan L. Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. *arXiv preprint arXiv:2012.00759*, 2020. 2, 8, 10

[156] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv preprint arXiv:2002.10957*, 2020. 17

[157] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 14

[158] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. *arXiv preprint arXiv:2012.09793*, 2020. 13

[159] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv preprint arXiv:2011.14503*, 2020. 2, 8, 10

[160] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12275–12284, 2020. 15

[161] Ziheng Wang, Jeremy Wohlwend, and Tao Lei. Structured pruning of large language models. *arXiv preprint arXiv:1910.04732*, 2019. 16

[162] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, 2019. 18

[163] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv preprint arXiv:2006.03677*, 2020. 6

[164] Zhanghao Wu, Zhijian Liu, Ji Lin, Yujun Lin, and Song Han. Lite transformer with long-short range attention. *arXiv preprint arXiv:2004.11886*, 2020. 17

[165] Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. Bert-of-theseus: Compressing bert by progressive module replacing. *arXiv preprint arXiv:2002.02925*, 2020. 16

[166] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015. 15

[167] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5791–5800, 2020. 2, 12

[168] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Towards explainable human pose estimation by transformer. *arXiv preprint arXiv:2012.14214*, 2020. 8, 11

[169] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763, 2019. 5

[170] Zhaohui Yang, Yunhe Wang, Kai Han, Chunjing Xu, Chao Xu, Dacheng Tao, and Chang Xu. Searching for low-bit weights in quantized neural networks. In *NeurIPS*, 2020. 17

[171] Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11495–11504, 2020. 14

[172] Yuhui Yuan and Jingdong Wang. Ocnet: Object context network for scene parsing. *arXiv preprint arXiv:1809.00916*, 2018. 15

[173] Zhenxun Yuan, Xiao Song, Lei Bai, Wengang Zhou, Zhe Wang, and Wanli Ouyang. Temporal-channel transformer for 3d lidar-based video object detection in autonomous driving. *arXiv preprint arXiv:2011.13628*, 2020. 8

[174] Kaiyu Yue, Ming Sun, Yuchen Yuan, Feng Zhou, Errui Ding, and Fuxin Xu. Compact generalized non-local network. In *Advances in Neural Information Processing Systems*, pages 6510–6519, 2018. 15

[175] Chulhee Yun, Yin-Wen Chang, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. $o(n)$ connections are expressive enough: Universal approximability of sparse transformers. *arXiv preprint arXiv:2006.04862*, 2020. 18

[176] Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. Q8bert: Quantized 8bit bert. *arXiv preprint arXiv:1910.06188*, 2019. 16

[177] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*, 2020. 17

[178] Yanhong Zeng, Jianlong Fu, and Hongyang Chao. Learning joint spatial-temporal transformations for video inpainting. In *European Conference on Computer Vision*, pages 528–543. Springer, 2020. 2, 3, 14

[179] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 18

[180] Dong Zhang, Hanwang Zhang, Jinhui Tang, Meng Wang, Xiansheng Hua, and Qianru Sun. Feature pyramid transformer. In *ECCV*, 2020. 8, 15

[181] Fan Zhang, Yanqin Chen, Zhihang Li, Zhibin Hong, Jingtuo Liu, Feifei Ma, Junyu Han, and Errui Ding. Acfnet: Attentional class feature network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6798–6807, 2019. 15

[182] Hang Zhang, Han Zhang, Chenguang Wang, and Junyuan Xie. Co-occurrent features in semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 548–557, 2019. 15

[183] Kun Zhang, Peng He, Ping Yao, Ge Chen, Rui Wu, Min Du, Huimin Li, Li Fu, and Tianyao Zheng. Learning enhanced resolution-wise features for human pose estimation. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 2256–2260, 2020. 16

[184] Li Zhang, Xiangtai Li, Anurag Arnab, Kuiyuan Yang, Yunhai Tong, and Philip HS Torr. Dual graph convolutional network for semantic segmentation. *arXiv preprint arXiv:1909.06121*, 2019. 15

[185] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*, 2019. 5

[186] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020. 11

[187] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 267–283, 2018. 15

[188] Zihan Zhao, Yuncong Liu, Lu Chen, Qi Liu, Rao Ma, and Kai Yu. An investigation on different underlying quantization schemes for pre-trained language models. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 359–371. Springer, 2020. 17

[189] Minghang Zheng, Peng Gao, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *arXiv preprint arXiv:2011.09315*, 2020. 2, 8, 9, 10, 11

[190] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2012.15840*, 2020. 2, 8, 11

[191] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8739–8748, 2018. 2, 3

[192] Jun Zhu, Yuanyuan Qiu, Rui Zhang, Jun Huang, and Wenjun Zhang. Top-down saliency detection via contextual pooling. *Journal of Signal Processing Systems*, 74(1):33–46, 2014. 2

[193] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2, 8, 9, 10, 11

[194] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015. 5