Incorporating Convolution Designs into Visual Transformers

Kun Yuan[†], Shaopeng Guo[†], Ziwei Liu[‡], Aojun Zhou[†], Fengwei Yu[†] and Wei Wu[†]
[†]SenseTime Research, [‡]S-Lab, Nanyang Technological University
{yuankun,guoshaopeng,zhouaojun,yufengwei,wuwei}@sensetime.com, ziwei.liu@ntu.edu.sg

Abstract

Motivated by the success of Transformers in natural language processing (NLP) tasks, there emerge some attempts (e.g., ViT and DeiT) to apply Transformers to the vision domain. However, pure Transformer architectures often require a large amount of training data or extra supervision to obtain comparable performance with convolutional neural networks (CNNs). To overcome these limitations, we analyze the potential drawbacks when directly borrowing Transformer architectures from NLP. Then we propose a new ***Convolution-enhanced image Transformer (CeiT)*** *which combines the advantages of CNNs in extracting low-level features, strengthening locality, and the advantages of Transformers in establishing long-range dependencies. Three modifications are made to the original Transformer:* ***1)*** *instead of the straightforward tokenization from raw input images, we design an* ***Image-to-Tokens (I2T)*** *module that extracts patches from generated low-level features;* ***2)*** *the feed-froward network in each encoder block is replaced with a* ***Locally-enhanced Feed-Forward (LeFF)*** *layer that promotes the correlation among neighboring tokens in the spatial dimension;* ***3)*** *a* ***Layer-wise Class token Attention (LCA)*** *is attached at the top of the Transformer that utilizes the multi-level representations.*

Experimental results on ImageNet and seven downstream tasks show the effectiveness and generalization ability of CeiT compared with previous Transformers and state-of-the-art CNNs, without requiring a large amount of training data and extra CNN teachers. Besides, CeiT models also demonstrate better convergence with $3\times$ *fewer training iterations, which can reduce the training cost significantly*[1].

1. Introduction

Transformers [36] have become the de-facto standard for natural language processing (NLP) tasks due to their abilities to model long-range dependencies and to train in parallel. Recently, there exist some attempts to apply Transformers to vision domains [6, 9, 34, 4, 50, 5, 48], leading promising results in different tasks. Among them, Vision Transformer (ViT) [9] is the first pure Transformer architecture that is directly inherited from NLP, and applied to image classification. It obtains promising results compared to many state-of-the-art CNNs [24, 41, 18]. But it relies heavily on the large amount of dataset of JFT-300M [32], which limits the application in the scenarios with limited computing resources or labeled training data. To alleviate the dependence on a large amount of data, the Data-efficient image Transformers (DeiT) [34] introduce a CNN model as a teacher and applies knowledge distillation [13] to improve the student model of ViT. Thus DeiT that is only trained on ImageNet can obtain satisfactory results. But the requirement of trained high-performance CNN models is a potential computation burden. Besides, the choice of teacher models, distillation types may affect the final performance. Therefore, we intend to design a new visual Transformer that can overcome these limitations.

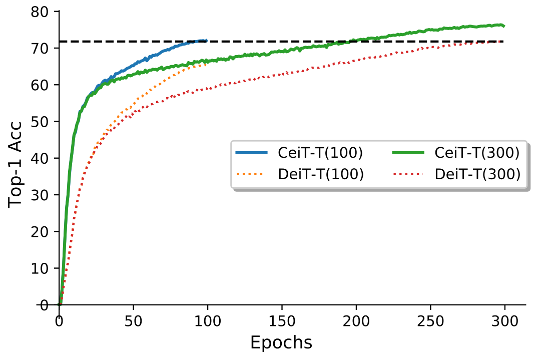

Figure 1. The fast convergence ability of CeiT models. CeiT models trained with 100 epochs obtain comparable results with DeiT models trained with 300 epochs. Other model settings are given in Table 8.

Some existing observations in these work can help us design desired architectures. In ViT, Transformer-based models underperform CNNs in the realm of ∼10M training samples. It claims that *"Transformers lack some of the inductive*

[1] Code and models will be released upon acceptance.

biases inherent to CNNs, and therefore do not generalize well when trained on insufficient data". In DeiT, a CNN teacher gives better performance than using a Transformer one, which probably due to *"the inductive bias inherited by the Transformer through distillation"*. These observations make us rethink whether it is appropriate to remove all convolutions from the Transformer. *And should the inductive biases inherited in the convolution be forgotten?*

Looking back to the convolution, the main characteristics are translation invariance and locality [21, 30]. Translation invariance is relevant to the weight sharing mechanism, which can capture information about the geometry and topology in vision tasks [22]. For the locality, it is a common assumption in visual tasks [10, 25, 8] that neighboring pixels always tend to be correlated. However, pure Transformer architectures do not fully utilize these prior biases that existed in images. *First*, ViT performs direct tokenization of patches from the raw input image with a size of 16×16 or 32×32. It is difficult to extract the low-level features which form some fundamental structures in images (e.g. corners and edges). *Second*, the self-attention modules concentrate on building long-range dependencies among tokens, ignoring the locality in the spatial dimension.

To address these problems, we design a *Convolution-enhanced image Transformer (CeiT)* to combine the advantages of CNNs in extracting low-level features, strengthening locality, and the advantages of Transformers in associating long-range dependencies. Three modifications are made compared to the vanilla ViT. *To solve the first problem*, instead of the straightforward tokenization from raw input images, we design an *Image-to-Tokens (I2T)* module that extracts patches from generated low-level features, where patches are in a smaller size and then flattened into a sequence of tokens. Due to a well-designed structure, the I2T module does not introduce more computation costs. *To solve the second problem*, the feed-froward network in each encoder block is replaced with a *Locally-enhanced Feed-Forward (LeFF)* layer that promotes the correlation among neighboring tokens in the spatial dimension. *To exploit the ability of self-attention*, a *Layer-wise Class token Attention (LCA)* is attached at the top of the Transformer that utilizes the multi-level representations to improve the final representation. In summary, our **contributions** are as follows:

- We design a new visual Transformer architecture namely *Convolution-enhanced image Transformer (CeiT)*. It combines the advantages of CNNs in extracting low-level features, strengthening locality, and the advantages of Transformers in establishing long-range dependencies.

- Experimental results on ImageNet and seven downstream tasks show the effectiveness and generalization ability of CeiT compared with previous Transformers and state-of-the-art CNNs, without requiring a large amount of training data and extra CNN teachers. For example, with a similar model size as ResNet-50, CeiT-S obtains a Top-1 accuracy of 82.0% on ImageNet. And the result boosts into 83.3% when fine-tuned in the resolution of 384×384.

- As shown in Figure 1, CeiT models demonstrate better convergence than pure Transformer models with $3\times$ fewer training iterations, which can reduce the training cost significantly.

2. Related Work

Transformer in Vision. iGPT [6] first introduce transformer to auto-regressively predict pixels, and obtaining pre-trained models without incorporating knowledge of the content in 2D images. However, it can only achieve reasonable performance in a tiny dataset (CIFAR10) with an extremely large model (1.4B). Recently, ViT [9] successfully makes standard Transformer scalable for image classification. It reshapes the images into a series of 16×16 patches as input tokens. However, ViT can only get comparable performance with state-of-the-art CNNs when trained on very large datasets. DeiT [34] augment ViT by introducing a mimic token and adopt knowledge distillation to mimic the output of a CNN teacher, which can obtain satisfactory results without training on large scale dataset. Besides, recent work also apply Transformers to various vision tasks, including object detection [4, 50], segmentation [39], image enhancement [5, 42] and video processing [45, 49].

Hybrid Models of Convolution and Self-attention. To utilize the advantages of self-attention in building long-range dependencies, some work introduces attention modules into CNNs [47, 43, 38, 2, 7, 16, 40]. Among these works, the Non-local network [38] insert non-local layers into the last several blocks of ResNet [11] and improve the performance on video recognition and instance segmentation. CCNet [17] attaches a criss-cross attention module at the top of a segmentation network. SASA [28], SANet [46] and Axial-SASA [37] propose to replace all convolutional layers by self-attention module to form a stand-alone self-attention network. Recent work also combines Transformers with CNNs. DETR [4] uses Transformer blocks outside the CNN backbone with the motivation to get rid of region proposals and non-maximal suppression for simplicity. ViLBERT [23] and VideoBERT [31] construct cross-modality models using CNN and BERT. Different from the above methods, CeiT incorporates convolutional designs into the basic building blocks of Transformer to inherit the inductive bias in CNNs, which a more elaborate design.

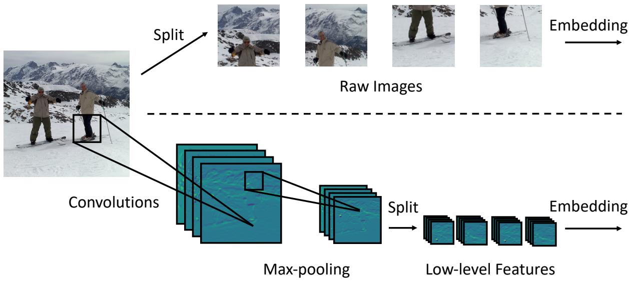

Figure 2. Comparisons of different tokenization methods. The upper one extracts patches from raw input images. The below one (I2T) uses the low-level features generated by a convolutional stem.

3. Methodology

Our CeiT is designed based on the ViT. First, we give a brief overview of the basic components of ViT in section 3.1. Next, we introduce three modifications that incorporate convolution designs and benefit visual Transformers, including an *Image-to-Tokens (I2T)* module in section 3.2, a *Locally-enhanced FeedForwad (LeFF)* module in section 3.3 and a *Layer-wise Class token Attention (LCA)* module in section 3.4. Last, we analyze the computation complexity of these proposed modules in section 3.5.

3.1. Revisiting Vision Transformer

We first revisit the basic components in ViT, including tokenization, encoder blocks, multi-head self-attention (MSA) layers, and feed-forward network (FFN) layers.

Tokenization. The standard Transformer [36] receives a sequence of token embeddings as input. To handle 2D images, ViT reshapes the image $\mathbf{x} \in \mathbb{R}^{H\times W\times 3}$ into a sequence of flattened 2D patches $\mathbf{x}_p \in \mathbb{R}^{N\times(P^2\cdot 3)}$, where (H, W) is the resolution of the original image, 3 is the number of channels of RGB images, (P, P) is the resolution of each image patch, and $N = HW/P^2$ is the resulting number of patches, which also serves as the effective input sequence length for the Transformer. And these patches are flattened and mapped to latent embeddings with a size of C. Then an extra class token is added to the sequence and serves as the image representation, resulting in the input of sequence with a size of $\mathbf{x}_t \in \mathbb{R}^{(N+1)\times C}$.

In practice, ViT splits each image with a patch size of 16×16 or 32×32. But the straightforward tokenization of input images with large patches may have two limitations: 1) it is difficult to capture low-level information in images (such as edges and corners); 2) large kernels are over-parameterized and are often hard to optimize, thus requires much more training samples or training iterations.

Encoder blocks. ViT is composed of a series of stacked encoders. Each encoder has two sub-layers of MSA and FFN. A residual connection [11] is employed around each sub-layer, followed by layer normalization (LN) [1]. The output for each encoder is:

$$\mathbf{y} = \text{LN}(\mathbf{x}' + \text{FFN}(\mathbf{x}')), \text{and } \mathbf{x}' = \text{LN}(\mathbf{x} + \text{MSA}(\mathbf{x})) \quad (1)$$

Different from CNNs where feature maps are down-sampled at the beginning of each stage, the length of tokens is not reduced in different Encoder blocks. The effective receptive field cannot be expanded efficiently, which may affect the efficiency of optimization in visual Transformers.

MSA. For a self-attention (SA) module, the sequence of input tokens $\mathbf{x}_t \in \mathbb{R}^{(N+1)\times C}$ are linear transformed into `qkv` spaces, *i.e.*, queries $\mathbf{Q} \in \mathbb{R}^{(N+1)\times C}$, keys $\mathbf{K} \in \mathbb{R}^{(N+1)\times C}$ and values $\mathbf{V} \in \mathbb{R}^{(N+1)\times C}$. Then a weighted sum over all values in the sequence is computed through:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{C}})\mathbf{V} \quad (2)$$

And a linear transformation is performed to the weighted values. MSA is an extension of SA. It splits queries, keys, and values for $\mathbf{h}$ times and performs the attention function in parallel, then projects their concatenated outputs.

Through computing dot-product, the similarity between different tokens is calculated, resulting in long-range and global attention. And a linear aggregation is performed for corresponding values $\mathbf{V}$.

FFN. FFN performs point-wise operations, which are applied to each token separately. It consists of two linear transformations with a non-linear activation in between:

$$\text{FFN}(\mathbf{x}) = \sigma(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 \quad (3)$$

where $\mathbf{W}_1 \in \mathbb{R}^{C\times K}$ is the weight of the first layer, projecting each token into a higher dimension K. And $\mathbf{W}_2 \in \mathbb{R}^{K\times C}$ is the weight of the second layer. $\mathbf{b}_1 \in \mathbb{R}^K$ and $\mathbf{b}_2 \in \mathbb{R}^C$ are the biases. And $\sigma(\cdot)$ is the non-linear activation of GELU [12] in ViT.

Complementary to the MSA module, the FFN module performs dimensional expansion/reduction and non-linear transformation on each token, thereby enhancing the representation ability of tokens. However, the spatial relationship among tokens, which is important in vision, is not considered. This leads that the original ViT needs a mass of training data to learn these inductive biases.

3.2. Image-to-Tokens with Low-level Features

To solve the above-mentioned problems in tokenization, we propose a simple but effective module named as *Image-to-Tokens (I2T)* that extracts patches from feature maps instead of raw input images. As shown in Figure 2, the I2T

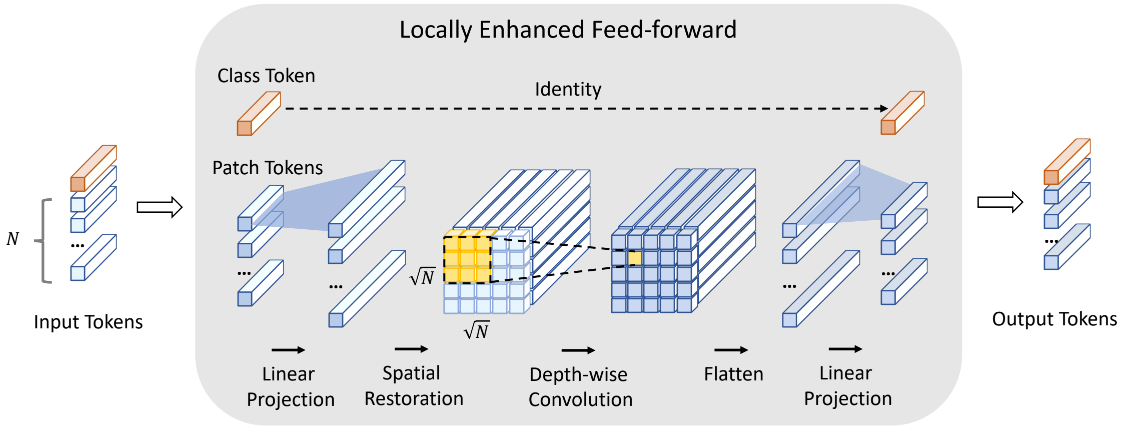

Figure 3. Illustration of the Locally-enhanced Feed-Forward module. First, patch tokens are projected into a higher dimension. Second, they are restored to "images" in the spatial dimension based on the original positions. Third, a depth-wise convolution is performed on the restored tokens as shown in the yellow region. Then the patch tokens are flattened and projected to the initial dimension. Besides, the class token conducts an identical mapping.

module is an lightweight stem that consists of a convolutional layer and a max-pooling layer. Ablation studies also suggest that a BatchNorm layer following the convolution layer benefits the training process. It can be denoted as:

$$\mathbf{x}' = \mathrm{I2T}(\mathbf{x}) = \mathrm{MaxPool}(\mathrm{BN}(\mathrm{Conv}(\mathbf{x}))) \tag{4}$$

where $\mathbf{x}' \in \mathbb{R}^{\frac{H}{S} \times \frac{W}{S} \times D}$, S is the stride w.r.t the raw input images, and D is the number of enriched channels. Then the learned feature maps are extracted into a sequence of patches in the spatial dimension. To keep the number of generated tokens consistent with ViT, we shrink the resolution of patches into $(\frac{P}{S}, \frac{P}{S})$. In practice, we set $S = 4$.

I2T fully utilizes the advantage of CNNs in extracting low-level features and reduces the training difficulty of embedding by shrinking the patch size. This is also different from the hybrid type of Transformer proposed in ViT, where a regular ResNet-50 is used to extract high-level features from the last two stages. Our I2T is much lighter.

3.3. Locally-Enhanced Feed-Forward Network

To combine the advantage of CNNs to extract local information with the ability of Transformer to establish long-range dependencies, we propose a *Locally-enhanced Feed-Forward Network (LeFF)* layer. In each Encoder block, we keep the MSA module unchanged, remaining the ability to capture global similarities among tokens. Instead, the original feed-forward network layer is replaced with the LeFF. The structure is given in Figure 3.

A LeFF module performs following procedures. *First*, given tokens $\mathbf{x}_t^h \in \mathbb{R}^{(N+1)\times C}$ generated from the preceding MSA module, we split them into patch tokens $\mathbf{x}_p^h \in \mathbb{R}^{(N+1)\times C}$ and a class token $\mathbf{x}_c^h \in \mathbb{R}^{C}$ accordingly. A linear projection is conducted to expand the embeddings of patch tokens to a higher dimension of $\mathbf{x}_p^{l_1} \in \mathbb{R}^{N\times(e\times C)}$, where e is the expand ratio. *Second*, the patch tokens are restored to "images" of $\mathbf{x}_p^s \in \mathbb{R}^{\sqrt{N}\times\sqrt{N}\times(e\times C)}$ on spatial dimension based on the position relative to the original image. *Third*, we perform a depth-wise convolution with kernel size of k on these restored patch tokens, enhancing the representation correlation with neighboring $k^2 - 1$ tokens, obtaining $\mathbf{x}_p^d \in \mathbb{R}^{\sqrt{N}\times\sqrt{N}\times(e\times C)}$. *Fourth*, these patch tokens are flattened into sequence of $\mathbf{x}_p^f \in \mathbb{R}^{N\times(e\times C)}$. *Last*, the patch tokens are projected to the initial dimension with $\mathbf{x}_p^{l_2} \in \mathbb{R}^{N\times C}$, and concatenated with the class token, resulting in $\mathbf{x}_t^{h+1} \in \mathbb{R}^{(N+1)\times C}$. Following each linear projection and depth-wise convolution, a BatchNorm and a GELU is added. These procedures can be noted as:

$$\mathbf{x}_c^h, \mathbf{x}_p^h = \mathrm{Split}(\mathbf{x}_t^h) \tag{5}$$

$$\mathbf{x}_p^{l_1} = \mathrm{GELU}(\mathrm{BN}(\mathrm{Linear1}(\mathbf{x}_p^h))) \tag{6}$$

$$\mathbf{x}_p^s = \mathrm{SpatialRestore}(\mathbf{x}_p^{l_1}) \tag{7}$$

$$\mathbf{x}_p^d = \mathrm{GELU}(\mathrm{BN}(\mathrm{DWConv}(\mathbf{x}_p^s))) \tag{8}$$

$$\mathbf{x}_p^f = \mathrm{Flatten}(\mathbf{x}_p^d) \tag{9}$$

$$\mathbf{x}_p^{l_2} = \mathrm{GELU}(\mathrm{BN}(\mathrm{Linear2}(\mathbf{x}_p^f))) \tag{10}$$

$$\mathbf{x}_t^{h+1} = \mathrm{Concat}(\mathbf{x}_c^h, \mathbf{x}_p^{l_2}) \tag{11}$$

3.4. Layer-wise Class-Token Attention

In CNNs, as the network deepens, the receptive field of the feature map increases. Similar observations are also found in ViT, whose "attention distance" increases with depth. Therefore, feature representations will be different at different layers. To integrate information across different layers, we design a *Layer-wise Class-token Attention (LCA)* module. Unlike the standard ViT that takes the class token $\mathbf{x}_c^{(L)}$ at the last L-th layer as the final representation, LCA makes attention over class tokens at different layers.

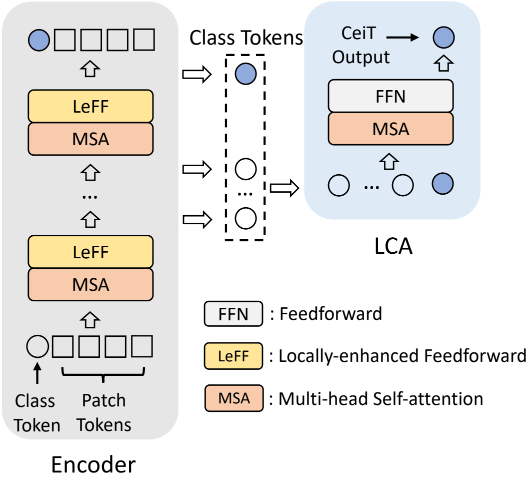

Figure 4. The proposed Layer-wise Class-token Attention block. It integrates information across different layers through receiving a sequence of class tokens as inputs.

As shown in Figure 4, LCA gets a sequence of class tokens as the input, which can be denoted as $\mathbf{X}_c = [\mathbf{x}_c^{(1)}, \cdots, \mathbf{x}_c^{(l)}, \cdots, \mathbf{x}_c^{(L)}]$, where l denotes the layer depth. LCA follows the standard Transformer block, which contains a MSA and a FFN layer. Particularly, it only computes unidirectional similarities between the L-th class token $\mathbf{x}_c^{(L)}$ and other class tokens. These modifications reduce the complexity of computing attention from $O(n^2)$ to $O(n)$. And the corresponding value of $\mathbf{x}_c^{(L)}$ is aggregated with others through attention. Then the aggregated value is sent into a FFN layer, resulting in the final representations of $\mathbf{x}_c^{(L)'}$.

3.5. Computational Complexity Analysis

We analyze the extra computational complexity (in terms of FLOPs) brought by our modifications. Generally, with a small increase in computational cost, our CeiT model can efficiently combine the advantage of CNNs and Transformers, resulting in higher performance and better convergence.

I2T vs Original. The type of tokenization affects the computational cost of embedding. For the original one with a patch size of 16×16, the FLOPs are $3C(HW)^2$. For I2T, the FLOPs are consist of two parts, including feature generation and embedding. In this paper, the generated features are $4\times$ smaller than the input. And the detailed architecture of I2T is given in section 4.1. The total FLOPs of I2T are $(\frac{147}{4}+\frac{9}{64})DHW+\frac{1}{64}DCHW$. For a ViT-B/16 model, the ratio between I2T and the original one is around 1.1. In this way, the extra computational cost is negligible.

LeFF vs FFN. In a FFN layer with $e = 4$, the FLOPs are $8(N+1)^2C^2$. The main extra computation cost of LeFF is introduced by the depth-wise convolution, whose FLOPs are $4k^2N^2C$. The increase of FLOPs is small since $O(C^2) \gg O(C)$ in practice as given in Table 1.

LCA vs Encoder Block. Compared with the standard Encoder block, the LCA only computes attention over the L-th class token. Both the computation cost in MSA and FFN has been reduced to $\frac{1}{N}$. The cost can be ignored compared with the other 12 encoder blocks.

4. Experiments

We perform extensive experiments to demonstrate the effectiveness of our proposed CeiT. In section 4.1, we give the details of used visual datasets and training settings. In section 4.2, we compare CeiT with other state-of-the-art architectures including CNNs and Transformers in ImageNet. In section 4.3, we transfer CeiT models trained on ImageNet to other benchmark datasets, showing the strong generalization ability. In section 4.4, we conduct ablation studies on our modifications. In section 4.5, we show the fast convergence ability of our CeiT models.

4.1. Experimental Settings

Network Architectures. We build our CeiT architectures by following the basic configurations of ViT and DeiT. The details are given in Table 1. The I2T module consists of a convolutional layer with a kernel size of 7 and a stride of 2, generating enriched channels of 32. And a BatchNorm layer is added for stable training. Then a max-pooling layer with a kernel size of 3 and a stride of 2 is followed, resulting in feature maps with $4\times$ smaller than the input image. Compared to the patch size of 16×16 in ViT, we use a patch size of 4×4 in generating a sequence of tokens. We follow the standard setting in the number of the depth of 12. For the LeFF module, we set the expand ratio e to be 4. And the kernel size for the depth-wise convolution is 3×3. For the LCA module, the number of heads and the ratio of MLP follow those of the standard Encoder blocks.

Implementation Details. All of our experiments are performed on the NVIDIA Tesla V100 GPUs. We adopt the same training strategy in DeiT. We list the detailed settings for training, fine-tuning, and transfer learning in Table 2.

Datasets. Instead of using a large-scale training dataset of JFT300M or ImageNet22K, we adopt the mid-sized ImageNet [44] dataset. It consists of 1.2M training images belonging to 1000 classes, and 50K validation images. Besides, we also test on some downstream benchmarks to evaluate the transfer ability of our trained CeiT models. These datasets consist different scenes, including fine-grained recognition (Standford Cars [19], Oxford-102

Table 1. Variants of our CeiT architecture. The FLOPs are calculated for images at resolution 224×224. And $7ks2$ means a convolution/pooling with kernel size of 7 and stride of 2.

Model	I2T			encoder	embedding	heads	LeFF		Params	FLOPs
	conv	maxpool	channels	blocks	dimension		e	k	(M)	(G)
CeiT-T	$k7s2$	$k3s2$	32	12	192	3	4	3	6.4	1.2
CeiT-S	$k7s2$	$k3s2$	32	12	384	6	4	3	24.2	4.5
CeiT-B	$k7s2$	$k3s2$	32	12	768	12	4	3	86.6	17.4

Table 2. The hyper-paramters that varies depends on tasks. The training and fine-tuning on ImageNet in our experienet adopt the same setting in DeiT. We use the same settings for finetuning on different downstream datasets.

Task	dataset	input	Epochs	batch	learning	LR	warmup	weight	repeated
		size		size	rate	scheduler	epoch	decay	aug [14]
training	ImageNet	224	300	1024	1e-3	cosine	5	0.05	✓
fine-tuning	ImageNet	384	30	1024	5e-6	constant	0	1e-8	✓
transferring	downstream	224&384	100	512	5e-4	cosine	2	1e-8	✗

Table 3. Details of used visual datasets.

dataset	classes	train data	val data
ImageNet	1000	1,281,167	50000
iNaturalist2018	8142	437513	24426
iNaturelist2019	1010	265240	3003
Standford Cars	196	8133	8041
Oxford-102 Followers	102	2040	6149
Oxford-IIIT-Pets	37	3680	3669
CIFAR100	100	50000	10000
CIFAR10	10	50000	10000

Followers [26] and Oxford-IIIT-Pets [27]), long-tailed classification (iNaturalist18 [15], iNaturalist19 [15]) and superordinate level classification (CIFAR10 [20], CIFAR100 [20]). The details are given in Table 3.

4.2. Results on ImageNet

We report the results on ImageNet validation dataset and ImageNet Real dataset [3] in Table 4. For comparison, we select CNNs (ResNets [11], EfficieNets [33], RegNets [29]) and Transformers (ViTs, DeiTs) to evaluate the effectiveness of our CeiT models.

CeiT vs CNNs. We first compare CeiT models with CNN models. CeiT-T achieves a Top-1 accuracy of 76.4% in ImageNet, which is close to the performance of ResNet-50. But CeiT-T only requires 3× fewer FLOPs and 4× fewer Params than ResNet-50. For the CeiT-S of a similar size as ResNet-50, its performance is 82.0%, achieving a higher performance (+5.3%) than that of ResNet-50 (76.7%). This performance also outperforms larger CNN models of ResNet-152 and RegNetY-8GF. When trained on the resolution of 384×384, CeiT-S$\uparrow$ 384 surpasses EfficientNet-B4 by 0.4%. It shows that we have obtained comparable results with EfficientNets, and have almost closed the gap between vision Transformers and CNNs.

CeiT vs ViT/DeiT. CeiT-T achieves a similar result of 76.4% with ViT-L/16 of 76.5%. This is a surprising result since the size of the CeiT-T model is only one-fifth the size of ViT-L/16. But this result is produced by the improvements of the training strategy and the modifications of the model structure. To further demonstrate the improvements brought by the structure, we compare CeiT with DeiT. CeiT models follow the same training strategy as given in section 4.1. Our modifications only increase the number of parameters by about 10%, and have almost no effect on FLOPs. In this way, CeiT-T outperforms DeiT-T by a large margin of 4.2% for the Top-1 accuracy. And CeiT-S obtains higher results than that of DeiT-S and DeiT-B by 2.1% and 0.2% respectively.

CeiT vs DeiT-Teacher. DeiT introduces a CNN teacher model as the extra supervision to optimize the Transformer, achieving higher performances. But it requires extra computation cost to obtain the trained CNN model. While CeiT does not need an additional CNN model to provide supervision information, except for the ground truth. Meanwhile, CeiT-T surpasses DeiT-T-Teacher by 1.9% of the Top-1 accuracy. And CeiT-S also outperforms DeiT-S-Teacher by 0.8%. These experimental results demonstrate the effectiveness of our CeiT.

4.3. Transfer Learning

To demonstrate the generalization power of pre-trained CeiT models, we conduct experiments of transfer learning in 7 downstream benchmarks. And the results are given in Table 6. Training details are given in the previous Table 2. It can be seen that CeiT-S outperforms DeiT-B in most datasets with fewer parameters and FLOPs. CeiT-S$\uparrow$ 384

Table 4. Accuracies on ImageNet and ImageNet Real of CeiT and of several state-of-the-art CNNs and Transformers, for models trained with no extra data. The notation $\uparrow 384$ means the model is fine-tuned on the resolution of 384×384. CeiT models outperform CNNs with similar FLOPs. Directly trained in ImageNet, CeiT models also achieve higher performances than DeiT models that are trained with extra CNN teachers.

Group	Model	FLOPs	Params	input	ImageNet		Real
		(G)	(M)	size	Top-1	Top-5	Top-1
CNNs	ResNet-18 [11]	1.8	11.7	224	70.3	86.7	77.3
	ResNet-50 [11]	4.1	25.6	224	76.7	93.3	82.5
	ResNet-101 [11]	7.8	44.5	224	78.3	94.1	83.7
	ResNet-152 [11]	11.5	60.2	224	78.9	94.4	84.1
	EfficientNet-B0 [33]	0.4	5.3	224	77.1	93.3	83.5
	EfficientNet-B1 [33]	0.7	7.8	240	79.1	94.4	84.9
	EfficientNet-B2 [33]	1.0	9.1	260	80.1	94.9	85.9
	EfficientNet-B3 [33]	1.8	12.2	300	81.6	95.7	86.8
	EfficientNet-B4 [33]	4.4	19.3	380	82.9	96.4	88.0
	RegNetY-4GF [29]	4.0	20.6	224	80.0	-	86.4
	RegNetY-8GF [29]	8.0	39.2	224	81.7	-	87.4
Transformers	ViT-B/16 [9]	18.7	86.5	384	77.9	-	-
	ViT-L/16 [9]	65.8	304.33	384	76.5	-	-
	DeiT-T [34]	1.2	5.7	224	72.2	91.1	80.6
	DeiT-S [34]	4.5	22.1	224	79.9	95.0	85.7
	DeiT-B [34]	17.3	86.6	224	81.8	95.6	86.7
	DeiT-T + Teacher [34]	1.2	5.7	224	74.5	91.9	82.1
	DeiT-S + Teacher [34]	4.5	22.1	224	81.2	95.4	86.8
	DeiT-B$\uparrow$384 [34]	52.8	86.6	384	83.1	96.2	87.7
	CeiT-T	1.2	6.4	224	**76.4**	93.4	83.6
	CeiT-S	4.5	24.2	224	**82.0**	95.9	87.3
	CeiT-T$\uparrow$384	3.6	6.4	384	**78.8**	94.7	85.6
	CeiT-S$\uparrow$384	12.9	24.2	384	**83.3**	96.5	88.1

Table 5. Ablation study results on the type of I2T. Top-1 accuracy and changes are reported.

I2T Type				Top-1
conv	maxpool	BN	channels	
✗	✗	✗	3	72.2
$k7s4$	✗	✗	64	71.4 (-0.8)
$k5s4$	✗	✗	64	71.1 (-1.1)
$k3s2$ + $k3s2$	✗	✗	64	70.4 (-1.8)
$k7s2$	$k3s2$	✗	32	72.9 (+0.7)
$k7s2$	$k3s2$	✓	32	73.4 (+1.2)

achieves state-of-the-arts results in most datasets. Notably, CeiT-S$\uparrow$ 384 get comparable results with EfficientNet-B7 with an input size of 600. It shows the strong potential of visual Transformers against CNNs.

4.4. Ablation Studies

To further identify the effects of the proposed modules, we conduct ablation studies on the main components of I2T, LeFF, and LCA. All of our ablation experiments are based on the DeiT-T model on ImageNet.

Different types of I2T module. The influencing factors in I2T include the kernel size of the convolution, the stride of the convolution, the existence of Max-pooling and BatchNorm layers. The results are given in Table 5. Without the Max-pooling layer, one convolution layer with a kernel of $k7s4$ and $k5s4$ each decreases the performance. An I2T with two convolution layers with a kernel of $k3s2$ also suffers from a drop. Both the Max-pooling and BatchNorm layers benefit the training. Therefore, we adopt the best structure (in the last row) in all of our experiments.

Different types of LeFF module. In a LeFF module, the size of the kernel determines the region size in which patch tokens establish local correlation. So we test using kernel sizes of 1×1, 3×3 and 5×5 in Table 7. Compared to the baseline without the middle depth-wise convolution, the type of 1×1 shows poor performance with a drop of 1.9%. This shows that simply increasing the number of layers for the Transformer does not certainly bring improve-

Table 6. Results on downstream tasks with ImageNet pre-training. CeiT models achieve state-of-the-arts performance. The results with the first two highest accuracies are bolded.

Model	FLOPs	ImageNet	iNat18	iNat19	Cars	Followers	Pets	CIFAR10	CIFAR100
Grafit ResNet-50 [35]	4.1G	79.6	69.8	75.9	92.5	98.2	-	-	-
Grafit RegNetY-8GF [35]	8.0G	-	76.8	80.0	94.0	**99.0**	-	-	-
EfficientNet-B5 [33]	10.3G	83.6	-	-	-	98.5	-	98.1	**91.1**
EfficientNet-B7 [33]	37.3G	**84.3**	-	-	**94.7**	**98.8**	-	98.9	**91.7**
ViT-B/16 [9]	18.7G	77.9	-	-	-	89.5	93.8	98.1	87.1
ViT-L/16 [9]	65.8G	76.5	-	-	-	89.7	93.6	97.9	86.4
Deit-B [34]	17.3G	81.8	73.2	77.7	92.1	98.4	-	**99.1**	90.8
Deit-B↑384 [34]	52.8G	83.1	**79.5**	**81.4**	93.3	98.5	-	**99.1**	90.8
CeiT-T	1.2G	76.4	64.3	72.8	90.5	96.9	93.8	98.5	88.4
CeiT-T↑384	3.6G	78.8	72.2	77.9	93.0	97.8	94.5	98.5	88.0
CeiT-S	4.5G	82.0	73.3	78.9	93.2	98.2	**94.6**	99.0	90.8
CeiT-S↑384	12.9G	**83.3**	**79.4**	**82.7**	**94.1**	98.6	**94.9**	**99.1**	90.8

Table 7. Ablation study results on the type of LeFF. Top-1 accuracy and changes are reported.

LeFF Type		Top-1
kernel size	BN	
✗	✗	72.2
1 × 1	✗	70.3 (-1.9)
3 × 3	✗	72.7 (+0.5)
5 × 5	✗	73.1 (+0.9)
3 × 3	✓	74.3 (+2.1)
5 × 5	✓	74.4 (+2.2)

ments. When increasing the kernel size to larger ones, each token can accumulate with neighboring tokens through the non-linear transformation. Both the types of 3 × 3 and 5 × 5 obtain gains. When adopting the BatchNorm layer, the model can achieve further accuracy improvements up to 2.2% of Top-1 accuracy. Based on the trade-off between the number of parameters and accuracy, we choose the kernel size of 3 × 3. The same as I2T, the presence of BatchNorm layers following transformation layers significantly improves the performance.

Effectiveness of LCA. We compare the performances w/wo the LCA module. Through adopting LCA, the performance improves from 72.2% to 72.8%, showing multi-level information contributes to the final image representation.

4.5. Fast Convergence

The standard visual Transformers, such as ViT and DeiT, usually require a large number of training epochs to converge. Using 3× fewer training epochs, the performances of DeiT suffer significant declines. As shown in Table 8, CeiT models demonstrate better convergence than DeiT models, resulting in higher performances in a large margin. And CeiT models trained in 100 epochs can obtain comparable results with DeiT models trained in 300 epochs. It shows that incorporating these inductive biases inherent in CNNs benefits the optimization procedure of visual Transformers.

Table 8. Comparisons of the ability of convergence between DeiT and CeiT models. CeiT models trained with 100 epochs obtain comparable results with DeiT models trained with 300 epochs. 1× means 100 epochs.

3×	Top-1	1×	Top-1	1×	Top-1
DeiT-T	72.2	DeiT-T	65.3	CeiT-T	72.2 (+6.9)
DeiT-S	79.9	DeiT-S	74.5	CeiT-S	78.9 (+4.4)
DeiT-B	81.8	DeiT-B	76.8	CeiT-B	81.8 (+5.0)

5. Conclusion

In this paper, we propose a new CeiT architecture that combines the advantages of CNNs in extracting low-level features, strengthening locality, and the advantages of Transformers in establishing long-range dependencies. CeiT obtains state-of-the-art performances on ImageNet and various downstream tasks, without requiring a large amount of training data and extra CNN teachers. Besides, CeiT models demonstrate better convergence than pure Transformer with 3× fewer training iterations, reducing the training cost significantly. Through incorporating convolution designs, we provide a new perspective for more effective visual Transformers.

References

[1] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. 3

[2] Irwan Bello, Barret Zoph, Quoc Le, Ashish Vaswani, and Jonathon Shlens. Attention augmented convolutional networks. In *ICCV*, pages 3285–3294. IEEE, 2019. 2

[3] Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? *CoRR*, abs/2006.07159, 2020. 6

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV (1)*, volume 12346 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2020. 1, 2

[5] 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. *CoRR*, abs/2012.00364, 2020. 1, 2

[6] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 1691–1703. PMLR, 2020. 1, 2

[7] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. A^2-nets: Double attention networks. In *NeurIPS*, pages 350–359, 2018. 2

[8] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. Image Process.*, 13(9):1200–1212, 2004. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2, 7, 8

[10] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. In *ECCV (1)*, volume 6311 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2010. 2

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE Computer Society, 2016. 2, 3, 6, 7

[12] Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016. 3

[13] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. 1

[14] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *CVPR*, pages 8126–8135. IEEE, 2020. 6

[15] Grant Van Horn, Oisin Mac Aodha, Yang Song, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist challenge 2017 dataset. *CoRR*, abs/1707.06642, 2017. 6

[16] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *ICCV*, pages 3463–3472. IEEE, 2019. 2

[17] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*, pages 603–612. IEEE, 2019. 2

[18] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV (5)*, volume 12350 of *Lecture Notes in Computer Science*, pages 491–507. Springer, 2020. 1

[19] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshops*, pages 554–561. IEEE Computer Society, 2013. 5

[20] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 6

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012. 2

[22] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, 1989. 2

[23] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, pages 13–23, 2019. 2

[24] Dhruv Mahajan, Ross B. Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV (2)*, volume 11206 of *Lecture Notes in Computer Science*, pages 185–201. Springer, 2018. 1

[25] Mehmet Kivanç Mihçak, Igor Kozintsev, Kannan Ramchandran, and Pierre Moulin. Low-complexity image denoising based on statistical modeling of wavelet coefficients. *IEEE Signal Process. Lett.*, 6(12):300–303, 1999. 2

[26] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, pages 722–729. IEEE Computer Society, 2008. 6

[27] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, pages 3498–3505. IEEE Computer Society, 2012. 6

[28] Niki Parmar, Prajit Ramachandran, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, pages 68–80, 2019. 2

[29] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces, 2020. 6, 7

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 2

[31] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *ICCV*, pages 7463–7472. IEEE, 2019. 2

[32] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, pages 843–852. IEEE Computer Society, 2017. 1

[33] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. 6, 7, 8

[34] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *CoRR*, abs/2012.12877, 2020. 1, 2, 7, 8

[35] Hugo Touvron, Alexandre Sablayrolles, Matthijs Douze, Matthieu Cord, and Hervé Jégou. Grafit: Learning fine-grained image representations with coarse labels, 2020. 8

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. 1, 3

[37] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan L. Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *ECCV (4)*, volume 12349 of *Lecture Notes in Computer Science*, pages 108–126. Springer, 2020. 2

[38] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pages 7794–7803. IEEE Computer Society, 2018. 2

[39] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *CoRR*, abs/2011.14503, 2020. 2

[40] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: convolutional block attention module. In *ECCV (7)*, volume 11211 of *Lecture Notes in Computer Science*, pages 3–19. Springer, 2018. 2

[41] Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *CVPR*, pages 10684–10695. IEEE, 2020. 1

[42] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *CVPR*, pages 5790–5799. IEEE, 2020. 2

[43] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alexander J. Smola. Stacked attention networks for image question answering. In *CVPR*, pages 21–29. IEEE Computer Society, 2016. 2

[44] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis E. H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *CoRR*, abs/2101.11986, 2021. 5

[45] Yanhong Zeng, Jianlong Fu, and Hongyang Chao. Learning joint spatial-temporal transformations for video inpainting. In *ECCV (16)*, volume 12361 of *Lecture Notes in Computer Science*, pages 528–543. Springer, 2020. 2

[46] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, pages 10073–10082. IEEE, 2020. 2

[47] 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 *ECCV (9)*, volume 11213 of *Lecture Notes in Computer Science*, pages 270–286. Springer, 2018. 2

[48] 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. *CoRR*, abs/2012.15840, 2020. 1

[49] Luowei Zhou, Yingbo Zhou, Jason J. Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *CVPR*, pages 8739–8748. IEEE Computer Society, 2018. 2

[50] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 1, 2