CvT: Introducing Convolutions to Vision Transformers

Haiping Wu\textsuperscript{1,2}\textsuperscript{*}, Bin Xiao\textsuperscript{2}\textsuperscript{†}, Noel Codella\textsuperscript{2}, Mengchen Liu\textsuperscript{2}, Xiyang Dai\textsuperscript{2}, Lu Yuan\textsuperscript{2}, Lei Zhang\textsuperscript{2}

\textsuperscript{1}McGill University\textsuperscript{2}Microsoft Cloud + AI

\texttt{haiping.wu2@mail.mcgill.ca, \{bixi, ncodella, mengcliu, xidai, luyuan, leizhang\}@microsoft.com}

\section*{Abstract}
We present in this paper a new architecture, named Convolutional vision Transformer (CvT), that improves Vision Transformer (ViT) in performance and efficiency by introducing convolutions into ViT to yield the best of both designs. This is accomplished through two primary modifications: a hierarchy of Transformers containing a new convolutional token embedding, and a convolutional Transformer block leveraging a convolutional projection. These changes introduce desirable properties of convolutional neural networks (CNNs) to the ViT architecture (i.e., shift, scale, and distortion invariance) while maintaining the merits of Transformers (i.e., dynamic attention, global context, and better generalization). We validate CvT by conducting extensive experiments, showing that this approach achieves state-of-the-art performance over other Vision Transformers and ResNets on ImageNet-1k, with fewer parameters and lower FLOPs. In addition, performance gains are maintained when pretrained on larger datasets (e.g., ImageNet-22k) and fine-tuned to downstream tasks. Pretrained on ImageNet-22k, our CvT-W24 obtains a top-1 accuracy of 87.7% on the ImageNet-1k val set. Finally, our results show that the positional encoding, a crucial component in existing Vision Transformers, can be safely removed in our model, simplifying the design for higher resolution vision tasks. Code will be released at \url{https://github.com/leoxiaobin/CvT}.

\section{1. Introduction}
Transformers \cite{vaswani2017attention, lample2019neural} have recently dominated a wide range of tasks in natural language processing (NLP) \cite{conneau2020unsupervised}. The Vision Transformer (ViT) \cite{dosovitskiy2020image} is the first computer vision model to rely exclusively on the Transformer architecture to obtain competitive image classification performance at large scale. The ViT design adapts Transformer architectures \cite{lample2019neural} from language understanding with minimal modifications. First, images are split into discrete non-overlapping patches (e.g. \(16 \times 16\)). Then, these patches are treated as tokens (analogous to tokens in NLP), summed with a special positional encoding to represent coarse spatial information, and input into repeated standard Transformer layers to model global relations for classification.

Despite the success of vision Transformers at large scale, the performance is still below similarly sized convolutional neural network (CNN) counterparts (e.g., ResNets \cite{he2016deep}) when trained on smaller amounts of data. One possible reason may be that ViT lacks certain desirable properties inherently built into the CNN architecture that make CNNs uniquely suited to solve vision tasks. For example, images have a strong 2D local structure: spatially neighboring pixels are usually highly correlated. The CNN archi-
Table 1: Representative works of vision Transformers.

| Method     | Needs Position Encoding (PE) | Token Embedding       | Projection for Attention | Hierarchical Transformers |
|------------|------------------------------|-----------------------|--------------------------|---------------------------|
| ViT [11], DeiT [30] | yes                          | non-overlapping       | linear                   | no                        |
| CPVT [6]   | no (w/ PE Generator)         | non-overlapping       | linear                   | no                        |
| TNT [14]   | yes                          | non-overlapping (patch+pixel) | linear                   | no                        |
| T2T [41]   | yes                          | overlapping (concatenate) | linear                   | partial (tokenization)    |
| PVT [34]   | yes                          | non-overlapping       | spatial reduction        | yes                       |
| CvT (ours) | no                           | overlapping (convolution) | convolution              | yes                       |

In summary, our proposed Convolutional vision Transformer (CvT) employs all the benefits of CNNs: local receptive fields, shared weights, and spatial subsampling, while keeping all the advantages of Transformers: dynamic attention, global context fusion, and better generalization. Our results demonstrate that this approach attains state-of-the-art performance when CvT is pre-trained with ImageNet-1k, while being lightweight and efficient: CvT improves the performance compared to CNN-based models (e.g., ResNet) and prior Transformer-based models (e.g., ViT, DeiT) while utilizing fewer FLOPS and parameters. In addition, CvT achieves state-of-the-art performance when evaluated at larger scale pretraining (e.g. on the public ImageNet-22k dataset). Finally, we demonstrate that in this new design, we can drop the positional embedding for tokens without any degradation to model performance. This not only simplifies the architecture design, but also makes it readily capable of accommodating variable resolutions of input images that is critical to many vision tasks.

2. Related Work

Transformers that exclusively rely on the self-attention mechanism to capture global dependencies have dominated in natural language modelling [31, 10, 25]. Recently, the Transformer based architecture has been viewed as a viable alternative to the convolutional neural networks (CNNs) in visual recognition tasks, such as classification [11, 30], object detection [3, 45, 43, 8, 28], segmentation [33, 36], image enhancement [4, 40], image generation [24], video processing [42, 44] and 3D point cloud processing [12].

Vision Transformers. The Vision Transformer (ViT) is the first to prove that a pure Transformer architecture can attain state-of-the-art performance (e.g., ResNets [15], EfficientNet [29]) on image classification when the data is large enough (i.e., on ImageNet-22k, JFT-300M). Specifically, ViT decomposes each image into a sequence of tokens (i.e., non-overlapping patches) with fixed length, and then applies multiple standard Transformer layers, consisting of Multi-Head Self-Attention module (MHSA) and Position-wise Feed-forward module (FFN), to model these tokens. DeiT [30] further explores the data-efficient training and distillation for ViT. In this work, we study how to combine...
CNNs and Transformers to model both local and global dependencies for image classification in an efficient way.

In order to better model local context in vision Transformers, some concurrent works have introduced design changes. For example, the Conditional Position encodings Visual Transformer (CPVT) [6] replaces the pre-defined positional embedding used in ViT with conditional position encodings (CPE), enabling Transformers to process input images of arbitrary size without interpolation. Transformer-iN-Transformer (TNT) [14] utilizes both an outer Transformer block that processes the patch embeddings, and an inner Transformer block that models the relation among pixel embeddings, to model both patch-level and pixel-level representation. Tokens-to-Token (T2T) [41] mainly improves tokenization in ViT by concatenating multiple tokens within a sliding window into one token. However, this operation fundamentally differs from convolutions especially in normalization details, and the concatenation of multiple tokens greatly increases complexity in computation and memory. PVT [34] incorporates a multi-stage design (without convolutions) for Transformer similar to multi-scales in CNNs, favoring dense prediction tasks.

In contrast to these concurrent works, this work aims to achieve the best of both worlds by introducing convolutions, with image domain specific inductive biases, into the Transformer architecture. Table 1 shows the key differences in terms of necessity of positional encodings, type of token embedding, type of projection, and Transformer structure in the backbone, between the above representative concurrent works and ours.

**Introducing Self-attentions to CNNs.** Self-attention mechanisms have been widely applied to CNNs in vision tasks. Among these works, the non-local networks [35] are designed for capturing long range dependencies via global attention. The local relation networks [17] adapts its weight aggregation based on the compositional relations (similarity) between pixels/features within a local window, in contrast to convolution layers which employ fixed aggregation weights over spatially neighboring input feature. Such an adaptive weight aggregation introduces geometric priors into the network which are important for the recognition tasks. Recently, BoTNet [27] proposes a simple yet powerful backbone architecture that just replaces the spatial convolutions with global self-attention in the final three bottleneck blocks of a ResNet and achieves a strong performance in image recognition. Instead, our work performs an opposite research direction: introducing convolutions to Transformers.

**Introducing Convolutions to Transformers.** In NLP and speech recognition, convolutions have been used to modify the Transformer block, either by replacing multi-head attentions with convolution layers [38], or adding additional convolution layers in parallel [39] or sequentially [13], to capture local relationships. Other prior work [37] proposes to propagate attention maps to succeeding layers via a residual connection, which is first transformed by convolutions. Different from these works, we propose to introduce convolutions to two primary parts of the vision Transformer: first, to replace the existing Position-wise Linear Projection for the attention operation with our Convolutional Projection, and second, to use our hierarchical multi-stage structure to enable varied resolution of 2D reshaped token maps, similar to CNNs. Our unique design affords significant performance and efficiency benefits over...
prior works.

3. Convolutional vision Transformer

The overall pipeline of the Convolutional vision Transformer (CvT) is shown in Figure 2. We introduce two convolution-based operations into the Vision Transformer architecture, namely the Convolutional Token Embedding and Convolutional Projection. As shown in Figure 2 (a), a multi-stage hierarchy design borrowed from CNNs [20, 15] is employed, where three stages in total are used in this work. Each stage has two parts. First, the input image (or 2D reshaped token maps) are subjected to the Convolutional Token Embedding layer, which is implemented as a convolution with overlapping patches with tokens reshaped to the 2D spatial grid as the input (the degree of overlap can be controlled via the stride length). An additional layer normalization is applied to the tokens. This allows each stage to progressively reduce the number of tokens (i.e., feature resolution) while simultaneously increasing the width of the tokens (i.e., feature dimension), thus achieving spatial downsampling and increased richness of representation, similar to the design of CNNs. Different from other prior Transformer-based architectures [11, 30, 41, 34], we do not sum the ad-hoc position embedding to the tokens. Next, a stack of the proposed Convolutional Transformer Blocks comprise the remainder of each stage. Figure 2 (b) shows the architecture of the Convolutional Transformer Block, where a depth-wise separable convolution operation [5], referred as Convolutional Projection, is applied for query, key, and value embeddings respectively, instead of the standard position-wise linear projection in ViT [11]. Additionally, the classification token is added only in the last stage. Finally, an MLP (i.e., fully connected) Head is utilized upon the classification token of the final stage output to predict the class.

We first elaborate on the proposed Convolutional Token Embedding layer. Next we show how to perform Convolutional Projection for the Multi-Head Self-Attention module, and its efficient design for managing computational cost.

3.1. Convolutional Token Embedding

This convolution operation in CvT aims to model local spatial contexts, from low-level edges to higher order semantic primitives, over a multi-stage hierarchy approach, similar to CNNs.

Formally, given a 2D image or a 2D-reshaped output token map from a previous stage \(x_{i-1} \in \mathbb{R}^{H_{i-1} \times W_{i-1} \times C_{i-1}}\) as the input to stage \(i\), we learn a function \(f(\cdot)\) that maps \(x_{i-1}\) into new tokens \(f(x_{i-1})\) with a channel size \(C_i\), where \(f(\cdot)\) is 2D convolution operation of kernel size \(s \times s\), stride \(s - o\) and \(p\) padding (to deal with boundary conditions). The new token map \(f(x_{i-1}) \in \mathbb{R}^{H_i \times W_i \times C_i}\) has height and width

\[
H_i = \left\lfloor \frac{H_{i-1} + 2p - s}{s - o} + 1 \right\rfloor, W_i = \left\lfloor \frac{W_{i-1} + 2p - s}{s - o} + 1 \right\rfloor. \tag{1}
\]

\(f(x_{i-1})\) is then flattened into size \(H_i W_i \times C_i\) and normalized by layer normalization [1] for input into the subsequent Transformer blocks of stage \(i\).

The Convolutional Token Embedding layer allows us to adjust the token feature dimension and the number of tokens at each stage by varying parameters of the convolution operation. In this manner, in each stage we progressively decrease the token sequence length, while increasing the token feature dimension. This gives the tokens the ability to represent increasingly complex visual patterns over increasingly larger spatial footprints, similar to feature layers of CNNs.

3.2. Convolutional Projection for Attention

The goal of the proposed Convolutional Projection layer is to achieve additional modeling of local spatial context, and to provide efficiency benefits by permitting the under-sampling of \(K\) and \(V\) matrices.

Fundamentally, the proposed Transformer block with Convolutional Projection is a generalization of the original Transformer block. While previous works [13, 39] try to add additional convolution modules to the Transformer Block for speech recognition and natural language processing, they result in a more complicated design and additional computational cost. Instead, we propose to replace the original position-wise linear projection for Multi-Head Self-Attention (MHSA) with depth-wise separable convolutions, forming the Convolutional Projection layer.

3.2.1 Implementation Details

Figure 3 (a) shows the original position-wise linear projection used in ViT [11] and Figure 3 (b) shows our proposed \(s \times s\) Convolutional Projection. As shown in Figure 3 (b), tokens are first reshaped into a 2D token map. Next, a Convolutional Projection is implemented using a depth-wise separable convolution layer with kernel size \(s\). Finally, the projected tokens are flattened into 1D for subsequent process. This can be formulated as:

\[
x_i^{q/k/v} = \text{Flatten}(\text{Conv2d}(\text{Reshape2D}(x_i), s)), \tag{2}
\]

where \(x_i^{q/k/v}\) is the token input for \(Q/K/V\) matrices at layer \(i\), \(x_i\) is the unperturbed token prior to the Convolutional Projection, \(\text{Conv2d}\) is a depth-wise separable convolution [5] implemented by: \(\text{Depth-wise Conv2d} \rightarrow \text{BatchNorm2d} \rightarrow \text{Point-wise Conv2d}\), and \(s\) refers to the convolution kernel size.

The resulting new Transformer Block with the Convolutional Projection layer is a generalization of the original
Transformer Block design. The original position-wise linear projection layer could be trivially implemented using a convolution layer with kernel size of $1 \times 1$.

### 3.2.2 Efficiency Considerations

There are two primary efficiency benefits from the design of our Convolutional Projection layer.

First, we utilize efficient convolutions. Directly using standard $s \times s$ convolutions for the Convolutional Projection would require $s^2C^2$ parameters and $O(s^2C^2T)$ FLOPs, where $C$ is the token channel dimension, and $T$ is the number of tokens for processing. Instead, we split the standard $s \times s$ convolution into a depth-wise separable convolution \[16\]. In this way, each of the proposed Convolutional Projection would only introduce an extra of $s^2C$ parameters and $O(s^2CT)$ FLOPs compared to the original position-wise linear projection, which are negligible with respect to the total parameters and FLOPs of the models.

Second, we leverage the proposed Convolutional Projection to reduce the computation cost for the MHSA operation. The $s \times s$ Convolutional Projection permits reducing the number of tokens by using a stride larger than 1. Figure 3 (c) shows the Convolutional Projection, where the key and value projection are subsampled by using a convolution with stride larger than 1. We use a stride of 2 for key and value projection, leaving the stride of 1 for query unchanged. In this way, the number of tokens for key and value is reduced 4 times, and the computational cost is reduced by 4 times for the later MHSA operation. This comes with a minimal performance penalty, as neighboring pixels/patches in images tend to have redundancy in appearance/semantics. In addition, the local context modeling of the proposed Convolutional Projection compensates for the loss of information incurred by resolution reduction.

### 3.3. Methodological Discussions

**Removing Positional Embeddings:** The introduction of Convolutional Projections for every Transformer block, combined with the Convolutional Token Embedding, gives us the ability to model local spatial relationships through the network. This built-in property allows dropping the position embedding from the network without hurting performance, as evidenced by our experiments (Section 4.4), simplifying design for vision tasks with variable input resolution.

**Relations to Concurrent Work:** Recently, two more related concurrent works also propose to improve ViT by incorporating elements of CNNs to Transformers. Tokens-to-Token ViT \[41\] implements a progressive tokenization, and then uses a Transformer-based backbone in which the length of tokens is fixed. By contrast, our CvT implements a progressive tokenization by a multi-stage process – containing both convolutional token embeddings and convolutional Transformer blocks in each stage. As the length of tokens are decreased in each stage, the width of the tokens (dimension of feature) can be increased, allowing increased richness of representations at each feature spatial resolution. Additionally, whereas T2T concatenates neighboring tokens into one new token, leading to increasing the complexity of memory and computation, our usage of convolutional token embedding directly performs contextual learning without concatenation, while providing the flexibility of controlling stride and feature dimension. To manage the complexity, T2T has to consider a deep-narrow architecture design with smaller hidden dimensions and MLP size than ViT in the subsequent backbone. Instead, we changed previous Transformer modules by replacing the position-wise linear projection with our convolutional projection.

Pyramid Vision Transformer (PVT) \[34\] overcomes the difficulties of porting ViT to various dense prediction tasks. In ViT, the output feature map has only a single scale with low resolution. In addition, computations and memory cost are relatively high, even for common input image sizes. To address this problem, both PVT and our CvT incorporate pyramid structures from CNNs to the Transformers structure. Compared with PVT, which only spatially subsamples the feature map or key/value matrices in projection, our CvT instead employs convolutions with stride to achieve this goal. Our experiments (shown in Section 4.4) demon-
Table 2: Architectures for ImageNet classification. Input image size is $224 \times 224$ by default. Conv. Embed.: Convolutional Token Embedding. Conv. Proj.: Convolutional Projection. $H_i$ and $D_i$ is the number of heads and embedding feature dimension in the $i$th MHSA module. $R_i$ is the feature dimension expansion ratio in the $i$th MLP layer.

| Stage | Output Size | Layer Name | CvT-13 | CvT-21 | CvT-W24 |
|-------|-------------|------------|--------|--------|---------|
| Stage1 | $56 \times 56$ | Conv. Proj. | $3 \times 3, 64$ | $3 \times 3, 64$ | $3 \times 3, 192$ |
|       |             | MHSAs      | $H_1 = 1, D_1 = 64$ | $H_1 = 1, D_1 = 64$ | $H_1 = 3, D_1 = 192$ |
|       |             | MLP        | $R_1 = 4$ | $R_1 = 4$ | $R_1 = 4$ |
| Stage2 | $28 \times 28$ | Conv. Proj. | $3 \times 3, 192$ | $3 \times 3, 192$ | $3 \times 3, 768$ |
|       |             | MHSAs      | $H_2 = 3, D_2 = 192$ | $H_2 = 3, D_2 = 192$ | $H_2 = 12, D_2 = 768$ |
|       |             | MLP        | $R_2 = 4$ | $R_2 = 4$ | $R_2 = 4$ |
| Stage3 | $14 \times 14$ | Conv. Proj. | $3 \times 3, 384$ | $3 \times 3, 384$ | $3 \times 3, 1024$ |
|       |             | MHSAs      | $H_3 = 6, D_3 = 384$ | $H_3 = 6, D_3 = 384$ | $H_3 = 16, D_3 = 1024$ |
|       |             | MLP        | $R_3 = 4$ | $R_3 = 4$ | $R_3 = 4$ |
| Head   | $1 \times 1$ | Linear     | 1000    |        |         |
| Params |             |            | 19.98 M | 31.54 M | 276.7 M |
| FLOPs  |             |            | 4.53 G  | 7.13 G  | 60.86 G |

strate that the fusion of local neighboring information plays an important role on the performance.

4. Experiments

In this section, we evaluate the CvT model on large-scale image classification datasets and transfer to various downstream datasets. In addition, we perform through ablation studies to validate the design of the proposed architecture.

4.1. Setup

For evaluation, we use the ImageNet dataset, with 1.3M images and 1k classes, as well as its superset ImageNet-22k with 22k classes and 14M images [9]. We further transfer the models pretrained on ImageNet-22k to downstream tasks, including CIFAR-10/100 [19], Oxford-IIIT-Pet [23], Oxford-IIIT-Flower [22], following [18, 11].

Model Variants We instantiate models with different parameters and FLOPs by varying the number of Transformer blocks of each stage and the hidden feature dimension used, as shown in Table 2. Three stages are adapted. We define CvT-13 and CvT-21 as basic models, with 19.98M and 31.54M parameters. CvT-X stands for Convolutional vision Transformer with $X$ Transformer Blocks in total. Additionally, we experiment with a wider model with a larger token dimension for each stage, namely CvT-W24 (W stands for Wide), resulting 298.3M parameters, to validate the scaling ability of the proposed architecture.

Training AdamW [21] optimizer is used with the weight decay of 0.05 for our CvT-13, and 0.1 for our CvT-21 and CvT-W24. We train our models with an initial learning rate of 0.02 and a total batch size of 2048 for 300 epochs, with a cosine learning rate decay scheduler. We adopt the same data augmentation and regularization methods as in ViT [30]. Unless otherwise stated, all ImageNet models are trained with an $224 \times 224$ input size.

Fine-tuning We adopt fine-tuning strategy from ViT [30]. SGD optimizer with a momentum of 0.9 is used for fine-tuning. As in ViT [30], we pre-train our models at resolution $224 \times 224$, and fine-tune at resolution of $384 \times 384$. We fine-tune each model with a total batch size of 512, for 20,000 steps on ImageNet-1k, 10,000 steps on CIFAR-10 and CIFAR-100, and 500 steps on Oxford-IIIT Pets and Oxford-IIIT Flowers-102.

4.2. Comparison to state of the art

We compare our method with state-of-the-art classification methods including Transformer-based models and representative CNN-based models on ImageNet [9], ImageNet Real [2] and ImageNet V2 [26] datasets in Table 3.

Compared to Transformer based models, CvT achieves a much higher accuracy with fewer parameters and FLOPs. CvT-21 obtains an 82.5% ImageNet Top-1 accuracy, which is 0.5% higher than DeiT-B with the reduction of 63% parameters and 60% FLOPs. When comparing to concurrent works, CvT still shows superior advantages. With fewer parameters, CvT-13 achieves a 81.6% ImageNet Top-1 accuracy, outperforming PVT-Small [34], T2T-ViT-14 [41], TNT-S [14] by 1.7%, 0.8%, 0.2% respectively.

Our architecture designing can be further improved in terms of model parameters and FLOPs by neural architecture search (NAS) [7]. In particular, we search the proper stride for each convolution projection of key and value terms of model parameters and FLOPs by neural architecture search (NAS) [7]. In particular, we search the proper stride for each convolution projection of key and value.
Table 3: Accuracy of manual designed architecture on ImageNet [9], ImageNet Real [2] and ImageNet V2 matched frequency [26]. Subscript 22k indicates the model pre-trained on ImageNet22k [9], and finetuned on ImageNet1k with the input size of 384 × 384, except BiT-M [18] finetuned with input size of 480 × 480.

| Method Type | Network | #Param. (M) | image size | FLOPs (G) | ImageNet top-1 (%) | Real top-1 (%) | V2 top-1 (%) |
|-------------|---------|-------------|-----------|-----------|-------------------|---------------|-------------|
| Convolutional Networks | ResNet-50 [15] | 25 | 224² | 4.1 | 76.2 | 82.5 | 63.3 |
| | ResNet-101 [15] | 45 | 224² | 7.9 | 77.4 | 83.7 | 65.7 |
| | ResNet-152 [15] | 60 | 224² | 11 | 78.3 | 84.1 | 67.0 |
| | ViT-B/16 [11] | 86 | 384² | 55.5 | 77.9 | 83.6 | – |
| | ViT-L/16 [11] | 307 | 384² | 191.1 | 76.5 | 82.2 | – |
| Transformers | DeiT-S [30][arxiv 2020] | 22 | 224² | 4.6 | 79.8 | 85.7 | 68.5 |
| | DeiT-B [30][arxiv 2020] | 86 | 224² | 17.6 | 81.8 | 86.7 | 71.5 |
| | PVT-Small [34][arxiv 2021] | 25 | 224² | 3.8 | 79.8 | – | – |
| | PVT-Medium [34][arxiv 2021] | 44 | 224² | 6.7 | 81.2 | – | – |
| | PVT-Large [34][arxiv 2021] | 61 | 224² | 9.8 | 81.7 | – | – |
| | T2T-ViT-r14 [41][arxiv 2021] | 22 | 224² | 6.1 | 80.7 | – | – |
| | T2T-ViT-r19 [41][arxiv 2021] | 39 | 224² | 9.8 | 81.4 | – | – |
| | T2T-ViT-r24 [41][arxiv 2021] | 64 | 224² | 15.0 | 82.2 | – | – |
| | TNT-S [14][arxiv 2021] | 24 | 224² | 5.2 | 81.3 | – | – |
| | TNT-B [14][arxiv 2021] | 66 | 224² | 14.1 | 82.8 | – | – |
| Convolutional Transformers | Ours: CvT-13 | 20 | 224² | 4.5 | 81.6 | 86.7 | 70.4 |
| | Ours: CvT-21 | 32 | 224² | 7.1 | 82.5 | 87.2 | 71.3 |
| | Ours: CvT-13:2384 | 20 | 384² | 16.3 | 83.0 | 87.9 | 71.9 |
| | Ours: CvT-21:2384 | 32 | 384² | 24.9 | 83.3 | 87.7 | 71.9 |
| | Ours: CvT-13-NAS | 18 | 224² | 4.1 | 82.2 | 87.5 | 71.3 |
| Convolution Networks22k | BiT-M:2384 [18] | 928 | 480² | 837 | 85.4 | – | – |
| Transformers22k | ViT-B/16:384 [11] | 86 | 384² | 55.5 | 84.0 | 88.4 | – |
| | ViT-L/16:384 [11] | 307 | 384² | 191.1 | 85.2 | 88.4 | – |
| | ViT-H/16:384 [11] | 632 | 384² | – | 85.1 | 88.7 | – |
| Convolutional Transformers22k | Ours: CvT-13:384 | 20 | 384² | 16 | 83.3 | 88.7 | 72.9 |
| | Ours: CvT-21:384 | 32 | 384² | 25 | 84.9 | 89.8 | 75.6 |
| | Ours: CvT-W24:384 | 277 | 384² | 193.2 | 87.7 | 90.6 | 78.8 |

number of model parameters ranging from 13.66M to 19.88M construct the search space. The NAS is evaluated directly on ImageNet-1k. The searched CvT-13-NAS, a bottleneck-like architecture with $\text{stride} = 2$, $\text{ratio}_{\text{MLP}} = 2$ at the first and last stages, and $\text{stride} = 1$, $\text{ratio}_{\text{MLP}} = 4$ at most layers of the middle stage, reaches to a 82.2% ImageNet Top-1 accuracy with fewer model parameters than CvT-13.

Compared to CNN-based models, CvT further closes the performance gap of Transformer-based models. Our smallest model CvT-13 with 20M parameters and 4.5G FLOPs surpasses the large ResNet-152 model by 3.2% on ImageNet Top-1 accuracy, while ResNet-151 has 3 times the parameters of CvT-13.

Furthermore, when more data are involved, our wide model CvT-W24* pretrained on ImageNet-22k reaches to 87.7% Top-1 Accuracy on ImageNet without extra data (e.g. JFT-300M), surpassing the previous best Transformer based models ViT-L/16 by 2.5% with similar number of model parameters and FLOPs.

### 4.3. Downstream task transfer

We further investigate the ability of our models to transfer by fine-tuning models on various tasks, with all models being pre-trained on ImageNet-22k. Table 4 shows the results. Our CvT-W24 model is able to obtain the best performance across all the downstream tasks considered, even when compared to the large BiT-R152x4 [18] model, which has more than $3 \times$ the number of parameters as CvT-W24.

### 4.4. Ablation Study

We design various ablation experiments to investigate the effectiveness of the proposed components of our architecture. First, we show that with our introduction of convolutions, position embeddings can be removed from the
### Table 4: Top-1 accuracy on downstream tasks. All the models are pre-trained on ImageNet-22k data

| Model   | Param (M) | CIFAR 10 | CIFAR 100 | Pets | Flowers 102 |
|---------|-----------|----------|-----------|------|-------------|
| BiT-M [18] | 928     | 98.91    | 92.17     | 94.46| 99.30       |
| ViT-B/16 [11] | 86      | 98.95    | 91.67     | 94.43| 99.38       |
| ViT-L/16 [11] | 307     | 99.16    | 93.44     | 94.73| 99.61       |
| ViT-H/16 [11] | 632     | 99.27    | 93.82     | 94.82| 99.51       |
| **Ours:** CvT-13 | 20      | 98.83    | 91.11     | 93.25| 99.50       |
| **Ours:** CvT-21 | 32      | 99.16    | 92.88     | 94.03| 99.62       |
| **Ours:** CvT-W24 | 277     | 99.39    | 94.09     | 94.73| 99.72       |

### Table 5: Ablations on position embedding.

| Method | Model   | Param (M) | Pos. Emb. | ImageNet Top-1 (%) |
|--------|---------|-----------|-----------|-------------------|
| a      | DeiT-S  | 22        | Default   | 79.8              |
| b      | DeiT-S  | 22        | N/A       | 78.0              |
| c      | CvT-13  | 20        | Every stage | 81.5            |
| d      | CvT-13  | 20        | First stage | 81.4            |
| e      | CvT-13  | 20        | Last stage | 81.4             |
| f      | CvT-13  | 20        | N/A       | 81.6              |

### Table 6: Ablations on Convolutional Token Embedding.

| Method | Conv. Embed. | Pos. Embed. | #Param (M) | ImageNet top-1 (%) |
|--------|--------------|-------------|------------|-------------------|
| a      |              |             | 19.5       | 80.7              |
| b      |              | ✓           | 19.9       | 81.1              |
| c      | ✓            | ✓           | 20.3       | 81.4              |
| d      | ✓            |             | 20.0       | 81.6              |

### Table 7: Ablations on Convolutional Projection with different strides for key and value projection. Conv. Proj. KV.: Convolutional Projection for key and value. We apply Convolutional Projection in all Transformer blocks.

| Method | Conv. Proj. KV. stride | Params (M) | FLOPs (G) | ImageNet top-1 (%) |
|--------|-------------------------|------------|-----------|-------------------|
| a      | 1                       | 20         | 6.55      | 82.3              |
| b      | 2                       | 20         | 4.53      | 81.6              |

### Table 8: Ablations on Convolutional Projection v.s. Position-wise Linear Projection. ✓ indicates the use of Convolutional Projection, otherwise use Position-wise Linear Projection.

| Method | Conv. Projection | Imagenet top-1 (%) |
|--------|------------------|--------------------|
|        | Stage 1         | Stage 2            | Stage 3           |
| a      |                 |                    | 80.6              |
| b      | ✓               |                    | 80.8              |
| c      | ✓               | ✓                  | 81.0              |
| d      | ✓               | ✓                  | 81.6              |

### Removing Position Embedding

Given that we have introduced convolutions into the model, allowing local context to be captured, we study whether position embedding is still needed for CvT. The results are shown in Table 5, and demonstrate that removing position embedding of our model does not degrade the performance. Therefore, position embeddings have been removed from CvT by default. As a comparison, removing the position embedding of DeiT-S would lead to 1.8% drop of ImageNet Top-1 accuracy, as it does not model image spatial relationships other than by adding the position embedding. This further shows the effectiveness of our introduced convolutions. Position Embedding is often realized by fixed-length learnable vectors, limiting the trained model adaptation of variable-length input. However, a wide range of vision applications take variable image resolutions. Recent work CPVT [6] tries to replace explicit position embedding of Vision Transformers with a conditional position encodings module to model position information on-the-fly. CvT is able to completely remove the positional embedding, providing the possibility of simplifying adaptation to more vision tasks without requiring a re-designing of the embedding.

### Convolutional Token Embedding

We study the effectiveness of the proposed Convolutional Token Embedding, and Table 6 shows the results. Table 6d is the CvT-13 model. When we replace the Convolutional Token Embedding with non-overlapping Patch Embedding [11], the performance drops 0.8% (Table 6a v.s. Table 6d). When position embedding is used, the introduction of Convolutional Token Embedding still obtains 0.3% improvement (Table 6b v.s. Table 6c). Further, when using both Convolutional Token Embedding and position embedding as Table 6d, it slightly drops 0.1% accuracy. These results validate the introduction of Convolutional Token Embedding not only improves the performance, but also helps CvT model spatial relationships without position embedding.

### Convolutional Projection

First, we compare the proposed Convolutional Projection with different strides in Table 7. By using a stride of 2 for key and value projection, we observe a 0.3% drop in ImageNet Top-1 accuracy, but
with 30% fewer FLOPs. We choose to use Convolutional Projection with stride 2 for key and value as default for less computational cost and memory usage.

Then, we study how the proposed Convolutional Projection affects the performance by choosing whether to use Convolutional Projection or the regular Position-wise Linear Projection for each stage. The results are shown in Table 8. We observe that replacing the original Position-wise Linear Projection with the proposed Convolutional Projection improves the Top-1 Accuracy on ImageNet from 80.6% to 81.5%. In addition, performance continually improves as more stages use the design, validating this approach as an effective modeling strategy.

5. Conclusion

In this work, we have presented a detailed study of introducing convolutions into the Vision Transformer architecture to merge the benefits of Transformers with the benefits of CNNs for image recognition tasks. Extensive experiments demonstrate that the introduced convolutional token embedding and convolutional projection, along with the multi-stage design of the network enabled by convolutions, make our CvT architecture achieve superior performance while maintaining computational efficiency. Furthermore, due to the built-in local context structure introduced by convolutions, CvT no longer requires a position embedding, giving it a potential advantage for adaption to a wide range of vision tasks requiring variable input resolution.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. 4
[2] Lucas Beyer, Olivier J Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? arXiv preprint arXiv:2006.07159, 2020. 6, 7
[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In European Conference on Computer Vision, pages 213–229. Springer, 2020. 2
[4] 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
[5] François Chollet. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1251–1258, 2017. 2, 4
[6] Xiangxiang Chu, Bo Zhang, Zhi Tian, Xiaolin Wei, and Huaxia Xia. Do we really need explicit position encodings for vision transformers? arXiv preprint arXiv:2102.10882, 2021. 3, 8
[7] Xiyang Dai, Dongdong Chen, Mengchen Liu, Yinpeng Chen, and Lu YuAn. Da-nas: Data adapted pruning for efficient neural architecture search. In European Conference on Computer Vision, 2020. 6
[8] 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] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255. Ieee, 2009. 6, 7
[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. 1, 2
[11] 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. 1, 2, 4, 5, 6, 7, 8
[12] Nico Engel, Vásileios Belagiannis, and Klaus Dietmayer. Point transformer. arXiv preprint arXiv:2111.00931, 2020. 2
[13] Anmol Gulati, James Qin, Chung-Cheng Chiu, Nikki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. arXiv preprint arXiv:2005.08100, 2020. 3, 4
[14] Kai Han, An Xiao, Enhuai Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. arXiv preprint arXiv:2103.00112, 2021. 1, 3, 6, 7
[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770–778, 2016. 1, 2, 4, 7
[16] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017. 5
[17] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. arXiv preprint arXiv:1904.11491, 2019. 3
[18] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. arXiv preprint arXiv:1912.11370, 6(2):8, 2019. 1, 6, 7
[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6
[20] Yann Lecun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In Contour and Grouping in Computer Vision. Springer, 1999. 2, 4
[21] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101, 2017. 6
