Escaping the Big Data Paradigm with Compact Transformers

Ali Hassani[1,*], Steven Walton[1*], Nikhil Shah[1],
Abulikemu Abuduweili[1], Jiachen Li[2,1], Humphrey Shi[1,2,3]
[1]SHI Lab @ University of Oregon, [2]University of Illinois at Urbana-Champaign, [3]Picsart AI Research (PAIR)

Abstract

With the rise of Transformers as the standard for language processing, and their advancements in computer vision, there has been a corresponding growth in parameter size and amounts of training data. Many have come to believe that because of this, transformers are not suitable for small sets of data. This trend leads to concerns such as: limited availability of data in certain scientific domains and the exclusion of those with limited resource from research in the field. In this paper, we aim to present an approach for small-scale learning by introducing Compact Transformers. We show for the first time that with the right size, convolutional tokenization, transformers can avoid overfitting and outperform state-of-the-art CNNs on small datasets. Our models are flexible in terms of model size, and can have as little as 0.28M parameters while achieving competitive results. Our best model can reach 98% accuracy when training from scratch on CIFAR-10 with only 3.7M parameters, which is a significant improvement in data-efficiency over previous Transformer based models being over 10x smaller than other transformers and is 15% the size of ResNet50 while achieving similar performance. CCT also outperforms many modern CNN based approaches, and even some recent NAS-based approaches. Additionally, we obtain a new SOTA result on Flowers-102 with 99.76% top-1 accuracy, and improve upon the existing baseline on ImageNet (82.71% accuracy with 29% as many parameters as ViT), as well as NLP tasks. Our simple and compact design for transformers makes them more feasible to study for those with limited computing resources and/or dealing with small datasets, while extending existing research efforts in data efficient transformers.

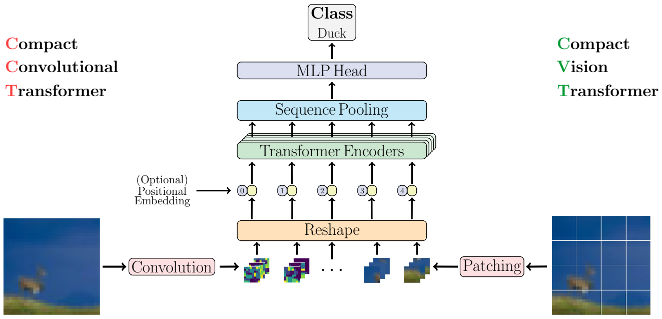


Figure 1: Overview of **CVT** (right), the basic compact transformer, and **CCT** (left), the convolutional variant of our compact transformer models. CCT can be quickly trained from scratch on small datasets, while achieving high accuracy (in under 30 minutes one can get 90% on an NVIDIA 2080Ti GPU or 80% on an AMD 5900X CPU on CIFAR-10 dataset).

1. Introduction

Convolutional neural networks (CNNs) [23] have been the standard for computer vision, since the success of AlexNet [22]. Krizhevsky *et al*. showed that convolutions are adept at vision based problems due to their invariance to spatial translations as well as having low relational inductive bias. He *et al*. [16] extended this work by introducing residual connections, allowing for significantly deeper models to perform efficiently. Convolutions leverage three important concepts that lead to their efficiency: *sparse interaction*, *weight sharing*, and *equivariant representations* [14]. Translational equivariance and invariance are properties of the convolutions and pooling layers, respectively [14, 36]. They allow CNNs to leverage natural image statistics and subsequently allow models to have higher sampling efficiency [34, 34].

[*] Equal contribution. Our code and pre-trained models are publicly available at https://github.com/SHI-Labs/Compact-Transformers

On the other end of the spectrum, Transformers have become increasingly popular and a major focus of modern machine learning research. Since the advent of Attention is All You Need [41], the research community saw a spike in transformer-based and attention-based research. While this work originated in natural language processing, these models have been applied to other fields, such as computer vision. Vision Transformer (ViT) [12] was the first major demonstration of a pure transformer backbone being applied to computer vision tasks. ViT highlights not only the power of such models, but also that large-scale training can trump inductive biases. The authors argued that "*Transformers lack some of the inductive biases inherent to CNNs, such as translation equivariance and locality, and therefore do not generalize well when trained on insufficient amounts of data.*" Over the past few years, an explosion in model sizes and datasets has also become noticeable which has led to a "data hungry" paradigm, making training transformers from scratch seem intractable for many types of pressing problems, where there are typically several orders of magnitude less data. It also limits major contributions in the research to those with vast computational resources.

As a result, CNNs are still the go-to models for smaller datasets because they are more efficient, both computationally and in terms of memory, when compared to transformers. Additionally, local inductive bias shows to be more important in smaller images. They require less time and data to train while also requiring a lower number of parameters to accurately fit data. However, they do not enjoy the long range interdependence that attention mechanisms in transformers provide. Reducing machine learning's dependence on large sums of data is important, as many domains, such as science and medicine, would hardly have datasets the size of ImageNet [10]. This is because events are far more rare and it would be more difficult to properly assign labels, let alone create a set of data which has low bias and is appropriate for conventional neural networks. In medical research, for instance, it may be difficult to compile positive samples of images for a rare disease without other correlating factors, such as medical equipment being attached to patients who are actively being treated. Additionally, for a sufficiently rare disease there may only be a few thousand images for positive samples, which is typically not enough to train a network with good statistical prediction unless it can sufficiently be pre-trained on data with similar attributes. This inability to handle smaller datasets has impacted the scientific community where they are much more limited in the models and tools that they are able to explore. Frequently, problems in scientific domains have little in common with domains of pre-trained models and when domains are sufficiently distinct pre-training can have little to no effect on the performance within a new domain [54]. In addition, it has been shown that strong performance on ImageNet does not necessarily result in equally strong performance in other domains, such as medicine [20]. Furthermore, the requisite of large data results in a requisite of large computational resources and this prevents many researchers from being able to provide insight. This not only limits the ability to apply models in different domains, but also limits its reproducibility. Verification of state of the art machine learning algorithms should not be limited to those with large infrastructures and computational resources.

The above concerns motivated our efforts to build more efficient models that can be effective in less data intensive domains and allow for training on datasets that are orders of magnitude smaller than those conventionally seen in computer vision and natural language processing (NLP) problems. Both Transformers and CNNs have highly desirable qualities for statistical inference and prediction, but each comes with their own costs. In this work, we try to bridge the gap between these two architectures and develop an architecture that can both attend to important features within images, while also being spatially invariant, where we have sparse interactions and weight sharing. This allows for a Transformer based model to be trained from scratch on small datasets like CIFAR-10 and CIFAR-100, providing competitive results with fewer parameters and low computational requirements.

In this paper we introduce ViT-Lite, a smaller and more compact version of ViT, which can obtain over 90% accuracy on CIFAR-10. We expand on ViT-Lite by introducing a sequence pooling and forming the Compact Vision Transformer (CVT). We further iterate by adding convolutional blocks to the tokenization step and thus creating the Compact Convolutional Transformer (CCT). Both of these simple additions add to significant increases in performance, leading to a top-1%accuracy of 98% on CIFAR-10. This makes our work the only transformer based model in the top 25 best performing models on CIFAR-10, without pre-training, and significantly smaller than the vast majority. Our model also outperforms most comparable CNN-based models within this domain, with the exception of certain Neural Architectural Search techniques [5]. Additionally, we show that our model can be lightweight, only needing 0.28 million parameters and still reach close to 90% top-1% accuracy on CIFAR-10. On ImageNet, CCT achieves 80.67% accuracy while still maintaining a small number of parameters and reduced computation. CCT outperforms ViT, while containing less than a third of the number of parameters with about a third of the computational complexity (MACs). Additionally, CCT outperform similarly sized and more recent models, such as DeiT [19]. This demonstrates the scalability of our model while maintaining compactness and computational efficiency.

The main contributions of this paper are:

- Extending transformer-based research to small data regimes, by introducing ViT-Lite, which can be trained from scratch and achieve high accuracy on datasets such as CIFAR-10.
- Introducing Compact Vision Transformer (CVT) with a new sequence pooling strategy, which pools over output tokens and improves performance.
- Introducing Compact Convolutional Transformer (CCT) to increase performance and provide flexibility for input image sizes while also demonstrating that these variants do not depend as much on Positional Embedding compared to the rest.

In addition, we demonstrate that our CCT model is fast, obtaining 90% accuracy on CIFAR-10 using a single NVIDIA 2080Ti GPU and 80% when trained on a CPU (AMD 5900X), both in under 30 minutes. Additionally, since our model has a relatively small number of parameters, it can be trained on the majority of GPUs, even if researchers do not have access to top of the line hardware. Through these efforts, we aim to help enable and extend research around Transformers to cases with limited data and/or researchers with limited resources.

2. Related Works

In NLP research, attention mechanisms [15, 2, 28] gained popularity for their ability to weigh different features within sequential data. Transformers [41] were introduced as a fully attention-based model, primarily for machine translation and NLP in general. Following this, attention-based models, specifically transformers have been applied to a wide variety of tasks beyond machine translation [11, 25, 46], including: visual question answering [27, 38], action recognition [4, 13], and the like. Many researchers also leveraged a combination of attention and convolutions in neural networks for visual tasks [42, 18, 3, 51]. Ramachandran *et al*. [33] introduced one of the first vision models that rely primarily on attention. Dosovitskiy *et al*. [12] introduced the first stand-alone transformer based model for image classification (ViT). In the following subsections, we briefly revisit ViT and several other related works.

2.1. Vision Transformer

Dosovitskiy *et al*. [12] introduced ViT primarily to show that reliance on CNNs or their structure is unnecessary, as prior to it, most attention-based models for vision were used either with convolutions [42, 3, 51, 6], or kept some of their properties [33]. The motivation, beyond self-attention's many desirable properties for a network, specifically its ability to make long range connections, was scalability. It was shown that ViT can successfully keep scaling, while CNNs start saturating in performance as the number of training samples grew. Through this, they concluded that large-scale training triumphs over the advantage of inductive bias that CNNs have, allowing their model to be competitive with CNN based architectures given sufficiently large amount of training data. ViT is composed of several parts: Image Tokenization, Positional Embedding, Classification Token, the Transformer Encoder, and a Classification Head. These subjects are discussed in more detail below.

Image Tokenization: A standard transformer takes as input a sequence of vectors, called tokens. For traditional NLP based transformers, word ordering provides a natural order to sequence the data, but this is not so obvious for images. To tokenize an image, ViT subdivides an image into non-overlapping square patches in raster-scan order. The sequence of patches, $\mathbf{x_p} \in \mathbb{R}^{H \times (P^2 C)}$ with patch size P, are flattened into 1D vectors and transformed into latent vectors of dimension d. This is equivalent to a convolutional layer with d filters, and $P \times P$ kernel size and stride. This simple patching and embedding method has a few limitations, in particular: loss of information along the boundary regions.

Positional Embedding: Positional embedding adds spatial information into the sequence. Since the model does not actually know anything about the spatial relationship between tokens, adding extra information to reflect that can be useful. Typically, this is either a learned embedding or tokens are given weights from two sine waves with high frequencies, which is sufficient for the model to learn that there exists a positional relationship between these tokens.

Transformer Encoder: A transformer encoder consists of a series of stacked encoding layers. Each encoder layer is comprised of two sub-layers: Multi-Headed Self-Attention (MHSA) and a Multi-Layer Perceptron (MLP) head. Each sub-layer is preceded by a layer normalization (LN), and followed by a residual connection to the next sub-layer.

Classification: Vision transformers typically add an extra learnable `[class]` token to the sequence of the embedded patches, representing the class parameter of an entire image and its state after transformer encoder can be used for classification. `[class]` token contains latent information, and through self-attention accumulates more information about the sequence, which is later used for classification. ViT [12] also explored averaging output tokens instead, but found no significant difference in performance. They did however find that the learning rates have to be adjusted between the two variants: `[class]` token *vs*. average pooling.

2.2. Data-Efficient Transformers

In an effort to reduce dependence on data, Touvron *et al*. [40] proposed Data-Efficient Image Transformers (DeiT). Using more advanced training techniques, and a novel knowledge transfer method, DeiT improves the classification performance of ViT on ImageNet-1k without large-scale pre-training on datasets such as JFT-300M [39] or ImageNet-21k [10]. By relying only on more augmentations [8] and training techniques [50, 49], it is shown that much smaller ViT variants that were unexplored by Dosovitskiy *et al*. can outperform the larger ones on ImageNet-1k without pre-training. Furthermore, DeiT variants were pushed even further through their novel knowledge transfer technique, specifically when using a convolutional model as the teacher. This work pushes forward accessibility of transformers in medium-sized datasets, and we aim to follow by extending the study to even smaller sets of data and smaller models. However, we base our work on the notion that *if a small dataset happens to be sufficiently novel, pre-trained models will not help train on that domain* and the model will not be appropriate for that dataset. While knowledge transfer is a strong technique, it requires a pre-trained model for any given dataset, adding to training time and complexity, with an additional forward pass, and as pointed out by Touvron *et al*. is usually only significant when there's a convolutional teacher available to transfer the inductive biases. As a result, it can be argued that if a network utilized just the bare minimum of convolutions, while keeping the pure transformer structure, it may need to rely less on large-scale training and transfer of inductive biases through knowledge transfer.

Yuan *et al*. [48] proposed Tokens-to-token ViT (T2T-ViT), which adopts a window- and attention-based tokenization strategy. Their tokenizer extracts patches of the input feature map, similar to a convolution, applies three sets of kernel weights, and produces three sets of feature maps, which are fed to self-attention as query and key-value pairs. This process is equivalent to convolutions producing the QKV projections in a self-attention module. Finally, this strategy is repeated twice, followed by a final patching and embedding. The entire process replaces patch and embedding in ViT. This strategy, along with their small-strided patch extraction, allows their network to model local structures, including along the boundaries between patches. This attention-based patch interaction leads to finer-grained tokens which allow T2T-ViT to outperform previous Transformer-based models on ImageNet. T2T-ViT differs from our work, in that it focuses on medium-sized datasets like ImageNet, which are not only far too large for many research problems in science and medicine but also resource demanding. T2T tokenizer also has more parameters and complexity compared to a convolutional one.

2.3. Convolution-inspired Transformers

Many works have been motivated to improve vision transformers and eliminate the need for large-scale pre-training. ConViT [9] introduces a *gated positional self-attention* (GPSA) that allows for a “soft” convolutional inductive bias within their model. GPSA allows their network to have more flexibility with respect to positional information. Since GPSA is able to be initialized as a convolutional layer, this allows their network to sometimes have the properties of convolutions or alternatively having the properties of attention. Its *gating parameter* can be adjusted by the network, allowing it to become more expressive and adapt to the needs of the dataset. Convolution-enhanced image Transformers (Ceit) [47] utilize convolutions throughout their model. They propose a convolution-based Image-to-Token module for tokenization. They also re-design the encoder with layers of multi-headed self-attention and their novel Locally Enhanced Feedforward Layer, which processes the spatial information form the extracted token. This allows creates a network that is competitive with other works such as DeiT on ImageNet. Convolutional vision Transformer (CvT) [45] introduces convolutional transformer encoder layers, which use convolutions instead of linear projections for the QKV in self-attention. They also introduce convolutions into their tokenization step, and report competitive results compared to other vision transformers on ImageNet-1k. All of these works report results when trained from scratch on ImageNet (or larger) datasets.

2.4. Comparison

Our work differs from the aforementioned in several ways, in that it focuses on answering the following question: **Can vision transformers be trained from scratch on small datasets?** Focusing on a small datasets, we seek to create a model that can be trained, from scratch, on datasets that are orders of magnitude smaller than ImageNet. Having a model that is compact, small in size, and efficient allows greater accessibility, as training on ImageNet is still a difficult and data intensive task for many researchers. Thus our focus is on an accessible model, with few parameters, that can quickly and efficiently be trained on smaller platforms while still maintaining SOTA results.

3. Method

In order to provide empirical evidence that vision transformers are trainable from scratch when dealing with small sets of data, we propose three different models: ViT-Lite, **C**ompact **V**ision **T**ransformers (CVT), and **C**ompact **C**onvolutional **T**ransformers (CCT). ViT-Lite is nearly identical to the original ViT in terms of architecture, but with a more suitable size and patch size for small-scale learning. CVT builds on this by using our **Seq**uence **Pool**ing method (SeqPool), that pools the entire sequence of tokens produced by the transformer encoder. SeqPool replaces the conventional `[class]` token. CCT builds

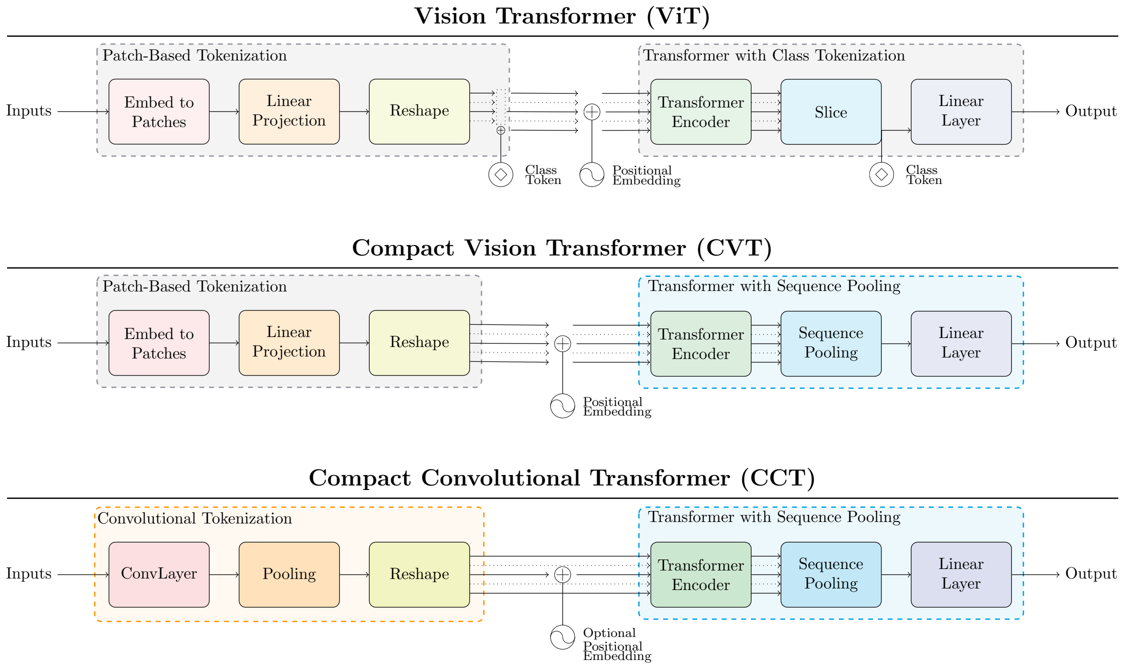


Figure 2: Comparing ViT (top) to CVT (middle) and CCT (bottom). CVT can be thought of as an ablated version of CCT, only utilizing sequence pooling and not a convolutional tokenizer. CVT may be preferable with more limited compute, as the patch-based tokenization is faster.

on CVT and utilizes a convolutional tokenizer, generating richer tokens and preserving local information. The convolutional tokenizer is better at encoding relationships between patches compared to the original ViT [12]. A detailed modular-level comparison of these models can be viewed in Figure 2.

The components of our compact transformers are further discussed in the following subsections: Transformer-based Backbone, Small and Compact Models, SeqPool, and Convolutional Tokenizer.

3.1. Transformer-based Backbone

In terms of model design, we follow the original Vision Transformer [12], and original Transformer [41]. As mentioned in Section 2.1, the encoder consists of transformer blocks, each including an MHSA layer and an MLP block. The encoder also applies Layer Normalization, $GELU$ activation, and dropout. Positional embeddings can be learnable or sinusoidal, both of which are effective.

3.2. Small and Compact Models

We propose smaller and more compact vision transformers. The smallest ViT variant, ViT-Base, includes a 12 layer transformer encoder with 12 attention heads, 64 dimensions per head, and 2048-dimensional hidden layers in the MLP blocks. This, along with the classifier and 16x16 patch and embedder results in over 85M parameters. We propose variants with as few as 2 layers, 2 heads, and 128-dimensional hidden layers. In Appendix A, we summarized the details of the variants we propose, the smallest of which can have as little as 0.22M parameters, while the largest (for small-scale learning) only have 3.8M parameters. We also adjust the tokenizer (patch size) according to the dataset we're training on, based on its image resolution. These variants, which are mostly similar in architecture to ViT, but different in size, are referred to as ViT-Lite. In our notation, we use the number of layers to specify size, as well as tokenization details: for instance, ViT-Lite-*12*/**16** has *12* transformer encoder layers, and a **16×16** patch size.

3.3. SeqPool

In order to map the sequential outputs to a singular class index, ViT [12] and most other common transformer-based classifiers follow BERT [11], in forwarding a learnable class or query token through the network and later feeding it to the classifier. Other common practices include global average pooling (averaging over tokens), which have been shown to be preferable in some scenarios. We introduce SeqPool, an attention-based method which pools over the output sequence of tokens. Our motivation is that the output sequence contains relevant information across different parts of the input image, therefore preserving this information can improve performance, and at no additional parameters compared to the learnable token. Additionally, this change slightly decreases computation, due one less token being forwarded. This operation consists of mapping the output sequence using the transformation $T : \mathbb{R}^{b\times n\times d} \mapsto \mathbb{R}^{b\times d}$. Given:

$$\mathbf{x}_L = \mathrm{f}(\mathbf{x}_0) \in \mathbb{R}^{b\times n\times d}$$

where $\mathbf{x}_L$ is the output of an L layer transformer encoder f, b is batch size, n is sequence length, and d is the total embedding dimension. $\mathbf{x}_L$ is fed to a linear layer $\mathrm{g}(\mathbf{x}_L) \in \mathbb{R}^{d\times 1}$, and softmax activation is applied to the output:

$$\mathbf{x}'_L = \mathrm{softmax}\left(\mathrm{g}(\mathbf{x}_L)^T\right) \in \mathbb{R}^{b\times 1\times n}$$

This generates an importance weighting for each input token, which is applied as follows:

$$\mathbf{z} = \mathbf{x}'_L\mathbf{x}_L = \mathrm{softmax}\left(\mathrm{g}(\mathbf{x}_L)^T\right) \times \mathbf{x}_L \in \mathbb{R}^{b\times 1\times d} \quad (1)$$

By flattening, the output $z \in \mathbb{R}^{b\times d}$ is produced. This output can then be sent through a classifier.

SeqPool allows our network to weigh the sequential embeddings of the latent space produced by the transformer encoder and correlate data across the input data. This can be thought of this as attending to the sequential data, where we are assigning importance weights across the sequence of data, only after they have been processed by the encoder. We tested several variations of this pooling method, including learnable and static methods, and found that the learnable pooling performs the best. Static methods, such as global average pooling have already been explored by ViT as well, as pointed out in section 2.1. We believe that the learnable weighting is more efficient because each embedded patch does not contain the same amount of entropy. This allows the model to apply weights to tokens with respect to the relevance of their information. Additionally, sequence pooling allows our model to better utilize information across spatially sparse data. We will further study the effects of this pooling in the ablation study (Sec 4.4). By replacing the conventional `class` token in ViT-Lite with SeqPool, Compact Vision Transformer is created. We use the same notations for this model: for instance, CVT-*7*/**4** has *7* transformer encoder layers, and a **4×4** patch size.

3.4. Convolutional Tokenizer

In order to introduce an inductive bias into the model, we replace patch and embedding in ViT-Lite and CVT, with a simple convolutional block. This block follows conventional design, which consists of a single convolution, *ReLU* activation, and a max pool. Given an image or feature map $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$:

$$\mathbf{x}_0 = \mathrm{MaxPool}(\mathrm{ReLU}(\mathrm{Conv2d}(\mathbf{x}))) \quad (2)$$

where the Conv2d operation has d filters, same number as the embedding dimension of the transformer backbone. Additionally, the convolution and max pool operations can be overlapping, which could increase performance by injecting inductive biases. This allows our model to maintain locally spatial information. Additionally, by using this convolutional block, the models enjoy an added flexibility over models like ViT, by no longer being tied to the input resolution strictly divisible by the pre-set patch size. We seek to use convolutions to embed the image into a latent representation, because we believe that it will be more efficient and produce richer tokens for the transformer. These blocks can be adjusted in terms of downsampling ratio (kernel size, stride and padding), and are repeatable for even further downsampling. Since self-attention has a quadratic time and space complexity with respect to the number of tokens, and number of tokens is equal to the resolution of the input feature map, more downsampling results in fewer tokens which noticeably decreases computation (at the expense of performance). We found that on top of the added performance gains, this choice in tokenization also gives more flexibility toward removing the positional embedding in the model, as it manages to maintain a very good performance. This is further discussed in Appendix C.1.

This convolutional tokenizer, along with SeqPool and the transformer encoder create Compact Convolutional Transformers. We use a similar notation for CCT variants, with the exception of also denoting the number of convolutional layers: for instance, CCT-*7*/**3**x2 has *7* transformer encoder layers, and a 2-layer convolutional tokenizer with **3×3** kernel size.

4. Experiments

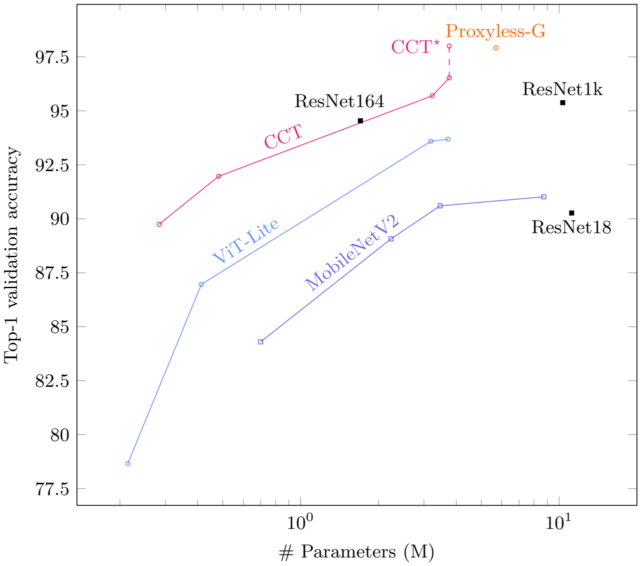


Figure 3: CIFAR-10 accuracy *vs*. model size (sizes < 12M). CCT* was trained longer.

Table 1: Top-1 validation accuracy comparisons. $\star$ variants were trained longer (see Table 2)

Model	**C-10**	**C-100**	**Fashion**	**MNIST**	**# Params**	**MACs**
Convolutional Networks (Designed for ImageNet)						
ResNet18	90.27%	66.46%	94.78%	99.80%	11.18 M	0.04 G
ResNet34	90.51%	66.84%	94.78%	99.77%	21.29 M	0.08 G
MobileNetV2/0.5	84.78%	56.32%	93.93%	99.70%	0.70 M	< **0.01** G
MobileNetV2/2.0	91.02%	67.44%	95.26%	99.75%	8.72 M	0.02 G
Convolutional Networks (Designed for CIFAR)						
ResNet56[16]	94.63%	74.81%	95.25%	99.27%	0.85 M	0.13 G
ResNet110[16]	95.08%	76.63%	95.32%	99.28%	1.73 M	0.26 G
ResNet1k-v2*[17]	95.38%	–	–	–	10.33 M	1.55 G
Proxyless-G[5]	97.92%	–	–	–	5.7 M	–
Vision Transformers						
ViT-12/16	83.04%	57.97%	93.61%	99.63%	85.63 M	0.43 G
ViT-Lite-7/16	78.45%	52.87%	93.24%	99.68%	3.89 M	0.02 G
ViT-Lite-7/8	89.10%	67.27%	94.49%	99.69%	3.74 M	0.06 G
ViT-Lite-7/4	93.57%	73.94%	95.16%	99.77%	3.72 M	0.26 G
Compact Vision Transformers						
CVT-7/8	89.79%	70.11%	94.50%	99.70%	3.74 M	0.06 G
CVT-7/4	94.01%	76.49%	95.32%	99.76%	3.72 M	0.25 G
Compact Convolutional Transformers						
CCT-2/3×2	89.75%	66.93%	94.08%	99.70%	**0.28** M	0.04 G
CCT-7/3×2	95.04%	77.72%	95.16%	99.76%	3.85 M	0.29 G
CCT-7/3×1	96.53%	80.92%	**95.56**%	**99.82**%	3.76 M	1.19 G
CCT-7/3×1*	**98.00%**	**82.72%**	–	–	3.76 M	1.19 G

Table 2: **CCT-7/3×1** top-1 accuracy on CIFAR-10/100 when trained longer

# Epochs	**Pos. Emb.**	**CIFAR-10**	**CIFAR-100**
300	Learnable	96.53%	80.92%
1500	Sinusoidal	97.48%	82.72%
5000	Sinusoidal	**98.00%**	**82.87%**

4.1. Datasets

We conducted image classification experiments using our method on the following datasets: CIFAR-10, CIFAR-100 (MIT License) [21], MNIST, Fashion-MNIST, Oxford Flowers-102 [30], and ImageNet-1k [10]. The first four datasets not only have a small number of training samples, but they are also small in resolution. Additionally, MNIST and Fashion-MNIST only contain a single channel, greatly reducing the information density. Flowers-102 has a relatively small number of samples, while having relatively higher resolution images and 102 classes. We divided these datasets into three categories: small-scale small resolution datasets (CIFAR-10/100, MNIST, and Fashion-MNIST), small-scale larger resolution (Flowers-102), and medium-scale (ImageNet-1k) datasets. We also include a study on NLP classification, presented in appendix G.

4.2. Hyperparameters

We used the timm package [43] to train the models (see Appendix E for details), except for cited works which are reported directly. For all experiments, we conducted a hyperparameter sweep for every different method and report the best results we were able to achieve. We will release all checkpoints corresponding to the reported numbers, and detailed training settings in the form of YAML files, with our

Table 3: ImageNet Top-1 validation accuracy comparison (no extra data or pretraining). This shows that larger variants of CCT could also be applicable to medium-sized datasets

Model	Top-1	# Params	MACs	Training Epochs
ResNet50 [16]	77.15%	25.55 M	4.15 G	120
ResNet50 (2021) [44]	79.80%	25.55 M	4.15 G	300
ViT-S [19]	79.85%	**22.05** M	**4.61** G	300
CCT-14/7×2	**80.67%**	22.36 M	5.53 G	300
DeiT-S [19]	81.16%	22.44M	**4.63** G	300
CCT-14/7×2 Distilled	**81.34%**	**22.36** M	5.53 G	300

Table 4: Flowers-102 Top-1 validation accuracy comparison. CCT outperforms other competitive models, having significantly fewer parameters and GMACs. This demonstrates the compactness on small datasets even with large images

Model	Resolution	Pretraining	Top-1	# Params	MACs
CCT-14/7×2	224	-	97.19%	22.17 M	18.63 G
DeiT-B	384	ImageNet-1k	98.80%	86.25 M	55.68 G
ViT-L/16	384	JFT-300M	99.74%	304.71 M	191.30 G
ViT-H/14	384	JFT-300M	99.68%	661.00 M	504.00 G
CCT-14/7×2	384	ImageNet-1k	**99.76%**	**22.17** M	**18.63** G

code. We also provide a report on hyperparamter settings in Appendix E. Unless stated otherwise, all tests were run for 300 epochs, and the learning rate is reduced per epoch based on cosine annealing [26]. All transformer based models (ViT-Lite, CVT, and CCT) were trained using the AdamW optimizer.

4.3. Performance Comparison

Small-scale small resolution training: In order to demonstrate that vision transformers can be as effective as convolutional neural networks, even in settings with small sets of data, we compare our compact transformers to ResNets [16], which are still very useful CNNs for small to medium amounts of data, as well as to MobileNetV2 [35], which are very compact and small-sized CNNs. We also compare with results from [17] where He *et al.* designed very deep (up to 1001 layers) CNNs specifically for CIFAR. The results are presented in Table 1, all of which are of models trained from scratch. We highlight the top performers. CCT-7/3x2 achieves on par results with the CNN models, while having significantly fewer parameters in some cases. We also compare our method to the original ViT [12] in order to express the effectiveness of smaller sized backbones, convolutional layers, as well our pooling technique. As these datasets were not trained from scratch in the original paper, we attempted to train the smallest variant: ViT-B/16 (ViT-12/16). We trained our best performing model, CCT-7/3x1, for longer than the 300 epochs to see how far it can go. Surprisingly, this model can get as high as 98% accuracy on CIFAR-10, and 82.87% accuracy on CIFAR-100 when trained for 5000 epochs, which is still fewer iterations an ImageNet pre-training would have. We present results from training on CIFAR-10/100 for 300, 1500 and 5000 epochs in Table 2. We observed that sinusoidal positional embedding had a small but noticeable edge over learnable when training longer. This represents the only transformer based model in the top 25 results on PapersWithCode for CIFAR-10 where models have no extra data or pre-training[1]. In addition to this, it is also one of the smallest models, being 15% the size of ResNet50 while maintaining similar performance. We present a plot of different models in Table 1 in Figure 3.

Medium-scale training: ImageNet training results are presented in Table 3, and compared to ResNet50 [16], ViT, and DeiT. We report ResNet50 from the original paper [16], as well as from Wightman *et al.* [44] which uses a similar training schedule to ours, and is therefore a fairer comparison. We also report a smaller ViT variant as proposed by Touvron *et al.* [40]. We also report CCT's performance with knowledge distillation, in order to compare it to DeiT [40]. Similar to DeiT, we trained our CCT-14/7x2 with a convolutional teacher and hard distillation loss. We used a RegNetY-16GF [32] (84M parameters), the same model

[1] https://paperswithcode.com/sota/image-classification-on-cifar-10

DeiT selected as the teacher. It is noticeable that distillation does not have as significant of an effect on CCT it does on DeiT. This can be attributed to the already existing inductive biases from the convolutional tokenizer. DeiT authors argued that a convolutional teacher would be able to transfer inductive biases to the student model.

Small-scale higher-resolution training: We also present our results on Flowers-102, in which we successfully reach reasonable performance without any pretraining, and with the same model size as our ImageNet model. We also claim state of the art with **99.76%** top-accuracy with ImageNet pretraining, which exceeds even far larger models pre-trained on JFT-300M. In addition to this we note that our model is at least a quarter the size of the next best model and almost 30× smaller than ViT-H/14. It can also be seen that CCT is $3 - 27\times$ more computationally efficient.

4.4. Ablation Study

We extend our previous comparisons by doing an ablation study on our methods. In this study, we progressively transform the original ViT into ViT-Lite, CVT, and CCT, and compare their top-1 accuracy scores. In this particular study, we report the results on CIFAR-10 and CIFAR-100 in Table 8 in Appendix F.

5. Conclusion

Transformers have commonly been perceived to be only applicable to larger-scale or medium-scale training. While their scalability is undeniable, we have shown within this paper that with proper configuration, a transformer can be successfully used in small data regimes as well, and outperform convolutional models of equivalent, and even larger, sizes. Our method is simple, flexible in size, and the smallest of our variants can be easily loaded on even a minimal GPU, or even a CPU. While part of research has been focused on large-scale models and datasets, we focus on smaller scales in which there is still much research to be done in data efficiency. We show that CCT can outperform other transformer based models on small datasets while also having a significant reduction in computational costs and memory constraints. This work demonstrates that transformers do not require vast computational resources and can allow for their applications in even the most modest of settings. This type of research is important to many scientific domains where data is far more limited that the conventional machine learning datasets which are used in general research. Continuing research in this direction will help open research up to more people and domains, extending machine learning research.

References

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007. 13

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016. 3

[3] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3286–3295, 2019. 3

[4] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? *arXiv preprint arXiv:2102.05095*, 2021. 3

[5] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations*, 2018. 2, 7, 17

[6] 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. 3

[7] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019. 13

[8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 4, 13

[9] Stéphane d'Ascoli, Hugo Touvron, Matthew Leavitt, Ari Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. *arXiv preprint arXiv:2103.10697*, 2021. 4

[10] 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. 2, 4, 7

[11] 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, 2019. 3, 5

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 3, 5, 8

[13] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2019. 3

[14] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. MIT press Cambridge, 2016. 1

[15] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines, 2014. 3

[16] 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, 7, 8, 17

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016. 7, 8, 17

[18] 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. 3

[19] Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. Improving transformer optimization through better initialization. In *International Conference on Machine Learning*, pages 4475–4483. PMLR, 2020. 2, 8

[20] Alexander Ke, William Ellsworth, Oishi Banerjee, Andrew Y. Ng, and Pranav Rajpurkar. Chextransfer: performance and parameter efficiency of imagenet models for chest x-ray interpretation. *Proceedings of the Conference on Health, Inference, and Learning*, Apr. 2021. 2

[21] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images, 2009. 7

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 1

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

[24] Xin Li and Dan Roth. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002. 13

[25] 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. 3

[26] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *ICLR*, 2017. 8

[27] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*, 2019. 3

[28] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation, 2015. 3

[29] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011. 13

[30] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 7

[31] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. 14

[32] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10428–10436, 2020. 8

[33] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*, 2019. 3

[34] Daniel L Ruderman and William Bialek. Statistics of natural images: Scaling in the woods. *Physical review letters*, 73(6):814, 1994. 1

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 8

[36] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015. 1

[37] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013. 13

[38] 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. 3

[39] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017. 4

[40] 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. 3, 8

[41] 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. 2, 3, 5

[42] 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. 3

[43] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 7, 13

[44] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021. 8

[45] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021. 4

[46] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019. 3

[47] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. *arXiv preprint arXiv:2103.11816*, 2021. 4

[48] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021. 4

[49] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019. 4, 13

[50] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 4, 13

[51] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019. 3

[52] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*, 2015. 13

[53] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13001–13008, 2020. 13

[54] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020. 2

A. Variants

Within this appendix, we present architectural details of our variants in Tables 5 and 6.

Table 5: Transformer backbones in each variant.

Model	# Layers	# Heads	Ratio	Dim
ViT-Lite-6	6	4	2	256
ViT-Lite-7	7	4	2	256
CVT-6	6	4	2	256
CVT-7	7	4	2	256
CCT-2	2	2	1	128
CCT-4	4	2	1	128
CCT-6	6	4	2	256
CCT-7	7	4	2	256
CCT-14	14	6	3	384

Table 6: Tokenizers in each variant.

Model	# Layers	# Convs	Kernel	Stride
ViT-Lite-7/8	7	1	8×8	8×8
ViT-Lite-7/4	7	1	4×4	4×4
CVT-7/8	7	1	8×8	8×8
CVT-7/4	7	1	4×4	4×4
CCT-2/3x2	2	2	3×3	1×1
CCT-7/3x1	7	1	3×3	1×1
CCT-7/7x2	7	2	7×7	2×2

B. Computational Resources

For most experiments, we used a machine with an Intel(R) Core(TM) i9-9960X CPU @ 3.10GHz and 4 NVIDIA(R) RTX(TM) 2080Tis (11GB). The exception was the CPU test which was performed with an AMD Ryzen 9 5900X. Each ImageNet experiment was performed on a single machine either with 2 AMD EPYC(TM) 7662s and 8 NVIDIA(R) RTX(TM) A6000s (48GB), or 2 AMD EPYC(TM) 7713s and 8 NVIDIA(R) A100s (80GB).

C. Additional analyses

Within this appendix we present some additional performance analyses which were conducted.

C.1. Positional Embedding

To determine the effects of our small & compact design, sequence pooling, and convolutional tokenizer, we perform an ablation study focused on positional embedding, seen in Table 7. In this study, we experiment with ViT (original sizing), ViT-Lite, CVT, and CCT, and investigate the effects of: a learnable positional embedding, a standard sinusoidal embedding, as well as no positional embedding. We finish the table with our best model, which also has augmented training and an optimal tuning (refer to Appendix E). In these experiments, we find that positional encoding matters in all variants, but to varying degrees. In particular, CCT relies less on positional encoding, and it can be safely removed much impact in accuracy. We also tested our CCT model without SeqPool, using the standard `[class]` token instead, and found that there was little to no effect from having a positional encoder or not, depending on model size. This suggests that convolutions are what helps provide spatially sparse information to the transformer, while also helping the model overcome some of the previous limitations, allowing for more efficient use of data. We do find that SeqPool helps slightly in this respect, but overall has a larger effect on increasing total accuracy. Lastly, we find that with proper data augmentation and tuning, the overall performance can be increased, and a low dependence on positional information can be maintained.

C.2. Performance vs Dataset Size

In this experiment, we evaluated model performance on smaller subsets of CIFAR-10 to determine the relationship between performance and the number of samples within a dataset. Samples were removed uniformly from each class in CIFAR-10. For this experiment, we compared ViT-Lite and CCT. In Figure 4, we see the comparison of each model's accuracy vs the number of samples per class. We show how each model performs when given only 500, 1000, 2000, 3000, 4000, or 5000 (original) samples per class, meaning the total training set ranges from one tenth the size to full. It can be ovserved that CCT is more robust since it is able to obtain higher accuracy with a lower number of samples per class, especially in the low sample regime.

C.3. Performance vs Dimensionality

In order to determine whether transformers are dependant on high dimensional data, as opposed to the number of samples (explored in Appendix C.2), we experimented with downsampled and upsampled versions of CIFAR-10. In Figure 5, we present the image dimensionality vs the performance of CCT *vs.* ViT-Lite. Both models were trained with images of sizes ranging from 16×16 to 64×64. It can be observed that CCT performs better on all image sizes, with a widening difference as the number of pixels increases. From this, it can be inferred that CCT is able to better utilize the information density of an image, while ViT does not see continued performance increases after the standard 32x32 size.

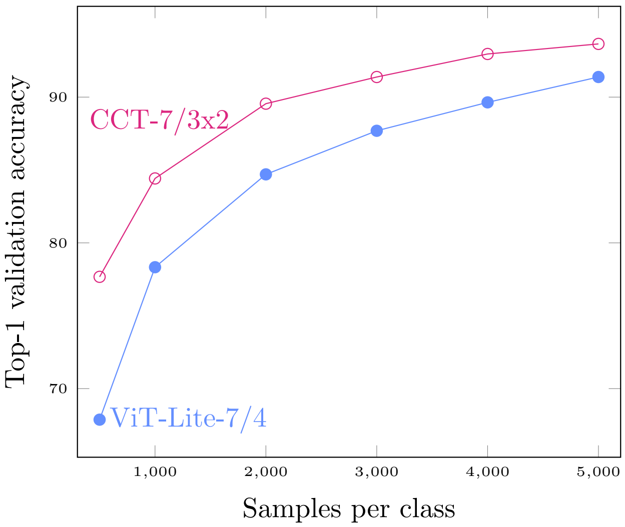


Figure 4: Reduced # samples / class (CIFAR-10)

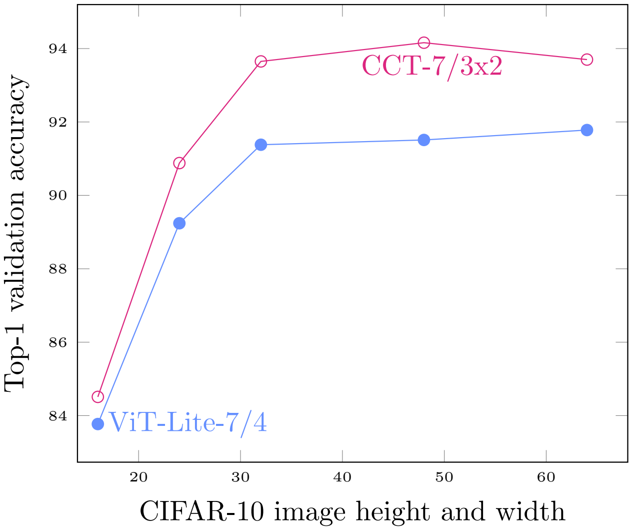


Figure 5: Image Size vs Accuracy (CIFAR-10)

D. Dimensionality Experiments

Within this appendix, we extend the analysis from Appendix C.3, showing the difference in performance when using different types of positional embedding. Figure 6 shows the difference of the accuracy when models are being trained from scratch. On the other hand, Figure 7 shows the performance difference when models are only used in inference and pre-trained on the 32x32 sized images. We note that in Figure 7(a) that we do not provide inference for image sizes greater than the pre-trained image because the learnable positional embeddings do not allow us to extend in this direction. We draw the reader's attention to Figure 6(c) and Figure 7(c) to denote the large difference between the models when positional embedding is not used. We can see that in training CCT has very little difference when positional embeddings are used. Additionally, it should be noted that when performing inference our non-positional embedding CCT model has much higher generalizability than its ViT-Lite counterpart.

E. Hyperparameter tuning

We used the timm package [43] for our experiments (excluding NLP experiments). We also sued CutMix [49], Mixup [50], Randaugment [8], and Random Erasing [53]. For our small-scale small-resolution experiments, we conducted a hyperparameter sweep for each model on each dataset separately. However, all experiments that trained models from scratch, were trained for 300 epochs, unless mentioned otherwise. ViT, CVT and CCT all used the weighted Adam optimizer ($\beta_1 = 0.9$ and $\beta_2 = 0.999$). For CNNs, we observed that some models and datasets achieved their best results using AdamW, while most others performed best with SGD with momentum (0.9). We will release model checkpoints (PyTorch pickle files), as well as a full list of hyperparameters and training settings (in the form of YAML files readable by `timm`) along with our code for reproduction.

F. Ablation Study

Here in Table 8 we present the results from section 4.4. We provide a full list of ablated terms showing which factors give the largest boost in performances. "Model" column refers to variant (see Table 5 for details), "Conv" specifies the number of convolutional blocks (if an), and "Conv Size" specifies the kernel size. "Aug" denotes the use of AutoAugment [7]. "Tuning" specifies a minor change in dropout, attention dropout, and/or stochastic depth (see Table 9). The first row in Table 8 is essentially ViT. The next three rows are modified variants of ViT, which are not proposed in the original paper. These variants are more compact and use smaller patch sizes. It should be noted that the numbers reported in this table are best out of 4.

G. NLP experiments

To demonstrate the general purpose nature of our model we extended it to the domain of Natural Language Processing, focusing on classification tasks. This shows that our model is a general purpose classifier and is not restricted to the domain of image classification. Within this section, we present our text classification results on 5 datasets: AG-News [52], TREC [24], SST [37], IMDb [29], DBpedia [1]. The results are summarized in Table 10. As can be seen, our model outperforms the vanilla transformer, demonstrating that the techniques we use here also help with NLP tasks.

Table 7: Top-1 validation accuracy comparison when changing the positional embedding method. Augmentations and training techniques such as Mixup and CutMix were turned off for these experiments to highlight differences better. The numbers reported are best out of 4 runs with random initializations. † denotes model trained with extra augmentation and hyperparameter tuning.

Model	PE	CIFAR-10	CIFAR-100
Conventional Vision Transformers are more dependent on Positional Embedding			
ViT-12/16	Learnable	69.82% *(+3.11%)*	40.57% *(+1.01%)*
	Sinusoidal	69.03% *(+2.32%)*	39.48% *(−0.08%)*
	None	66.71% (*baseline*)	39.56% (*baseline*)
ViT-Lite-7/8	Learnable	83.38% *(+7.25%)*	55.69% *(+7.15%)*
	Sinusoidal	80.86% *(+4.73%)*	53.50% *(+4.96%)*
	None	76.13% (*baseline*)	48.54% (*baseline*)
CVT-7/8	Learnable	84.24% *(+6.52%)*	55.49% *(+7.23%)*
	Sinusoidal	80.84% *(+3.12%)*	50.82% *(+2.56%)*
	None	77.72% (*baseline*)	48.26% (*baseline*)
Compact Convolutional Transformers are less dependent on Positional Embedding			
CCT-7/7	Learnable	82.03% *(+0.21%)*	63.01% *(+3.24%)*
	Sinusoidal	81.15% *(−0.67%)*	60.40% *(+0.63%)*
	None	81.82% (*baseline*)	59.77% (*baseline*)
CCT-7/3×2	Learnable	90.69% *(+1.67%)*	65.88% *(+2.82%)*
	Sinusoidal	89.93% *(+0.91%)*	64.12% *(+1.06%)*
	None	89.02% (*baseline*)	63.06% (*baseline*)
CCT-7/3×2†	Learnable	95.04% *(+0.64%)*	77.72% *(+0.20%)*
	Sinusoidal	94.80% *(+0.40%)*	77.82% *(+0.30%)*
	None	94.40% (*baseline*)	77.52% (*baseline*)
CCT-7/3×1†	Learnable	96.53% *(+0.29%)*	80.92% *(+0.65%)*
	Sinusoidal	96.27% *(+0.03%)*	80.12% *(−0.15%)*
	None	96.24% (*baseline*)	80.27% (*baseline*)
CCT-7/7×1-noSeqPool	Learnable	82.41% *(+0.12%)*	62.61% *(+3.31%)*
	Sinusoidal	81.94% *(−0.35%)*	61.04% *(+1.74%)*
	None	82.29% (*baseline*)	59.30% (*baseline*)
CCT-7/3×2-noSeqPool	Learnable	90.41% *(+1.49%)*	66.57% *(+1.40%)*
	Sinusoidal	89.84% *(+0.92%)*	64.71% *(−0.46%)*
	None	88.92% (*baseline*)	65.17% (*baseline*)

The network is slightly modified from the vision CCT. We use GloVe (Apache License 2.0) [31] to provide the word embedding for the model, and do not train these parameters. Note that model sizes do not reflect the number of parameters for GloVe, which is around 20M. We treat text as single channel data and the embedding dimension as size 300. Additionally, the convolution kernels have size 1. Finally, we include masking in the typical manner. By doing so, CCT can get upwards of a 3% improvement on some datasets while using less parameters than vanilla transformers. Similar to our vision results, we find that CCT performs well on small NLP datasets. We note that the CCT models that perform best all have less than 1M parameters, which are significantly smaller than there vanilla counterparts, while out performing them.

Table 8: CIFAR Top-1 validation accuracy when transforming ViT into CCT step by step. We disabled advanced training techniques and augmentations for these runs.

Model	CLS	# Conv	Conv Size	Aug	Tuning	C-10	C-100	# Params	MACs
ViT-12/16	CT	✗	✗	✗	✗	69.82%	40.57%	85.63 M	0.43 G
ViT-Lite-7/16	CT	✗	✗	✗	✗	71.78%	41.59%	3.89 M	0.02 G
ViT-Lite-7/8	CT	✗	✗	✗	✗	83.38%	55.69%	3.74 M	0.06 G
ViT-Lite-7/4	CT	✗	✗	✗	✗	83.59%	58.43%	3.72 M	0.26 G
CVT-7/16	SP	✗	✗	✗	✗	72.26%	42.37%	3.89 M	0.02 G
CVT-7/8	SP	✗	✗	✗	✗	84.24%	55.49%	3.74 M	0.06 G
CVT-7/8	SP	✗	✗	✓	✗	87.15%	63.14%	3.74 M	0.06 G
CVT-7/4	SP	✗	✗	✗	✗	88.06%	62.06%	3.72 M	0.25 G
CVT-7/4	SP	✗	✗	✓	✗	91.72%	69.59%	3.72 M	0.25 G
CVT-7/4	SP	✗	✗	✓	✓	92.43%	73.01%	3.72 M	0.25 G
CVT-7/2	SP	✗	✗	✗	✗	84.80%	57.98%	3.76 M	1.18 G
CCT-7/7×1	SP	1	7 × 7	✗	✗	87.81%	62.83%	3.74 M	0.26 G
CCT-7/7×1	SP	1	7 × 7	✓	✗	91.85%	69.43%	3.74 M	0.26 G
CCT-7/7×1	SP	1	7 × 7	✓	✓	92.29%	72.46%	3.74 M	0.26 G
CCT-7/3×2	SP	2	3 × 3	✓	✓	93.65%	74.77%	3.85 M	0.29 G
CCT-7/3×1	SP	1	3 × 3	✓	✓	**94.47%**	**75.59%**	3.76 M	1.19 G

Table 9: Difference between **tuned** and not tuned runs in Table 8.

Hyper Param	**Not Tuned**	**Tuned**
MLP Dropout	0.1	0
MSA Dropout	0	0.1
Stochastic Depth	0	0.1

H. Additional experiments

H.1. Extended small-scale experiments

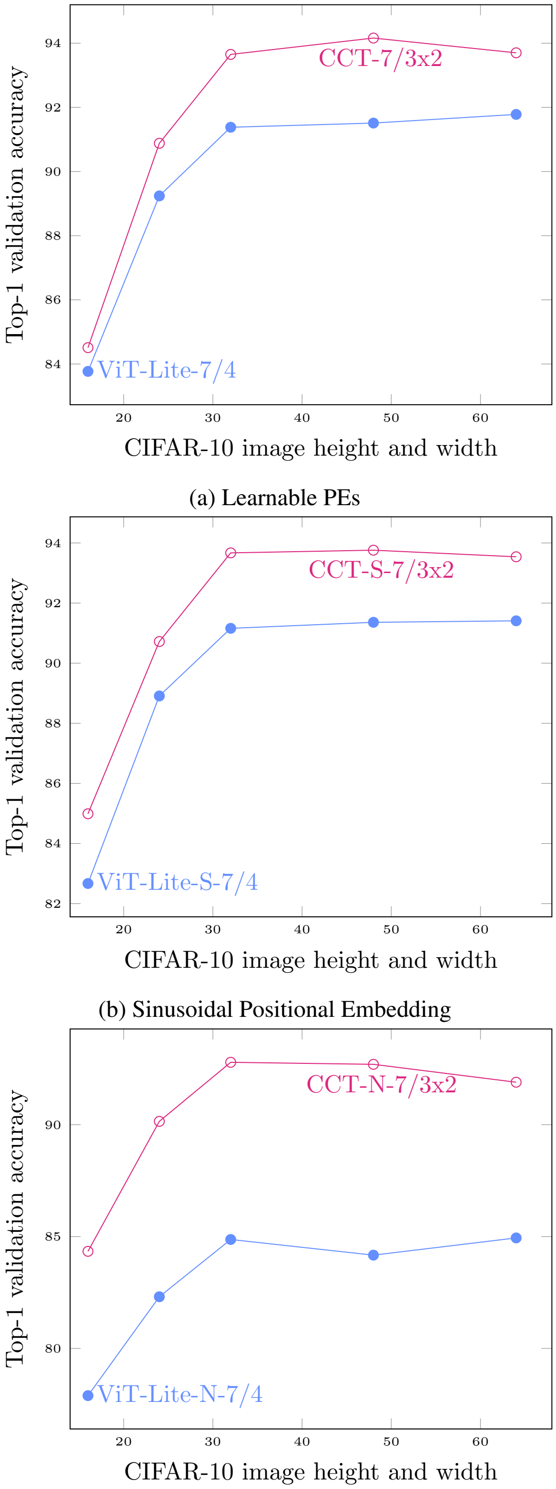
We present the extended version of Table 1 here with additional models in Table 11.

Table 10: Top-1 validation accuracy on text classification datasets. The number of parameters does not include the word embedding layer, because we use pretrained word-embeddings and freeze those layers while training.

Model	AGNews	TREC	SST	IMDb	DBpedia	# Params
Vanilla Transformer Encoders						
Transformer-2	93.28%	90.40%	67.15%	86.01%	98.63%	1.086 M
Transformer-4	93.25%	92.54%	65.20%	85.98%	96.91%	2.171 M
Transformer-6	93.55%	92.78%	65.03%	85.87%	98.24%	4.337 M
Vision Transformers						
ViT-Lite-2/1	93.02%	90.32%	67.66%	87.69%	98.99%	0.238 M
ViT-Lite-2/2	92.20%	90.12%	64.44%	87.39%	98.88%	0.276 M
ViT-Lite-2/4	90.53%	90.00%	62.37%	86.17%	98.72%	0.353 M
ViT-Lite-4/1	93.48%	91.50%	66.81%	87.38%	99.04%	0.436 M
ViT-Lite-4/2	92.06%	90.42%	63.75%	87.00%	98.92%	0.474 M
ViT-Lite-4/4	90.93%	89.30%	60.83%	86.71%	98.81%	0.551 M
ViT-Lite-6/1	93.07%	91.92%	64.95%	87.58%	99.02%	3.237 M
ViT-Lite-6/2	92.56%	89.38%	62.78%	86.96%	98.89%	3.313 M
ViT-Lite-6/4	91.12%	90.36%	60.97%	86.42%	98.72%	3.467 M
Compact Vision Transformers						
CVT-2/1	93.24%	90.44%	67.88%	87.68%	98.98%	0.238 M
CVT-2/2	92.29%	89.96%	64.26%	86.99%	98.93%	0.276 M
CVT-2/4	91.10%	89.84%	62.22%	86.39%	98.75%	0.353 M
CVT-4/1	93.53%	92.58%	66.64%	87.27%	99.04%	0.436 M
CVT-4/2	92.35%	90.36%	63.90%	86.96%	98.93%	0.474 M
CVT-4/4	90.71%	90.14%	61.98%	86.77%	98.80%	0.551 M
CVT-6/1	93.38%	92.06%	65.94%	86.78%	99.02%	3.237 M
CVT-6/2	92.57%	91.14%	64.57%	86.61%	98.86%	3.313 M
CVT-6/4	91.35%	91.66%	61.63%	86.13%	98.76%	3.467 M
Compact Convolutional Transformers						
CCT-2/1x1	93.40%	90.86%	**68.76%**	88.95%	99.01%	0.238 M
CCT-2/2x1	93.38%	91.86%	67.19%	**89.13%**	99.04%	0.276 M
CCT-2/4x1	**93.80%**	91.42%	64.47%	88.92%	99.04%	0.353 M
CCT-4/1x1	93.49%	91.84%	68.21%	88.71%	99.03%	0.436 M
CCT-4/2x1	93.30%	**93.54%**	66.42%	88.94%	**99.05%**	0.474 M
CCT-4/4x1	93.09%	93.20%	66.57%	88.86%	99.02%	0.551 M
CCT-6/1x1	93.73%	91.22%	66.59%	88.81%	98.99%	3.237 M
CCT-6/2x1	93.29%	92.10%	65.02%	88.74%	99.02%	3.313 M
CCT-6/4x1	92.86%	92.96%	65.84%	88.68%	99.02%	3.467 M

Table 11: Top-1 comparisons. $\star$ were trained longer (see Tab 2).

Model	**C-10**	**C-100**	**Fashion**	**MNIST**	**# Params**	**MACs**
Convolutional Networks (Designed for ImageNet)						
ResNet18	90.27%	66.46%	94.78%	99.80%	11.18 M	0.04 G
ResNet34	90.51%	66.84%	94.78%	99.77%	21.29 M	0.08 G
ResNet50	91.63%	68.27%	94.99%	99.79%	23.53 M	0.08 G
MobileNetV2/0.5	84.78%	56.32%	93.93%	99.70%	0.70 M	$<$ **0.01** G
MobileNetV2/1.0	89.07%	63.69%	94.85%	99.75%	2.24 M	0.01 G
MobileNetV2/1.25	90.60%	65.24%	95.05%	99.77%	3.47 M	0.01 G
MobileNetV2/2.0	91.02%	67.44%	95.26%	99.75%	8.72 M	0.02 G
Convolutional Networks (Designed for CIFAR)						
ResNet56[16]	94.63%	74.81%	95.25%	99.27%	0.85 M	0.13 G
ResNet110[16]	95.08%	76.63%	95.32%	99.28%	1.73 M	0.26 G
ResNet164-v1[17]	94.07%	74.84%	–	–	1.70 M	0.26 G
ResNet164-v2[17]	94.54%	75.67%	–	–	1.70 M	0.26 G
ResNet1k-v1[17]	92.39%	72.18%	–	–	10.33 M	1.55 G
ResNet1k-v2[17]	95.08%	77.29%	–	–	10.33 M	1.55 G
ResNet1k-v2*[17]	95.38%	–	–	–	10.33 M	1.55 G
Proxyless-G[5]	97.92%	–	–	–	5.7 M	–
Vision Transformers						
ViT-12/16	83.04%	57.97%	93.61%	99.63%	85.63 M	0.43 G
ViT-Lite-7/16	78.45%	52.87%	93.24%	99.68%	3.89 M	0.02 G
ViT-Lite-6/16	78.12%	52.68%	93.09%	99.66%	3.36 M	0.02 G
ViT-Lite-7/8	89.10%	67.27%	94.49%	99.69%	3.74 M	0.06 G
ViT-Lite-6/8	88.29%	66.40%	94.36%	99.73%	3.22 M	0.06 G
ViT-Lite-7/4	93.57%	73.94%	95.16%	99.77%	3.72 M	0.26 G
ViT-Lite-6/4	93.08%	73.33%	95.14%	99.74%	3.19 M	0.22 G
Compact Vision Transformers						
CVT-7/8	89.79%	70.11%	94.50%	99.70%	3.74 M	0.06 G
CVT-6/8	89.50%	68.80%	94.53%	99.74%	3.21 M	0.05 G
CVT-7/4	94.01%	76.49%	95.32%	99.76%	3.72 M	0.25 G
CVT-6/4	93.60%	74.23%	95.00%	99.75%	3.19 M	0.22 G
Compact Convolutional Transformers						
CCT-2/3×2	89.75%	66.93%	94.08%	99.70%	**0.28** M	0.04 G
CCT-4/3×2	91.97%	71.51%	94.74%	99.73%	0.48 M	0.05 G
CCT-6/3×2	94.43%	77.14%	95.34%	99.75%	3.33 M	0.25 G
CCT-7/3×2	95.04%	77.72%	95.16%	99.76%	3.85 M	0.29 G
CCT-6/3×1	95.70%	79.40%	95.41%	99.79%	3.23 M	1.02 G
CCT-7/3×1	96.53%	80.92%	**95.56**%	**99.82**%	3.76 M	1.19 G
CCT-7/3×1*	**98.00%**	**82.72%**	–	–	3.76 M	1.19 G

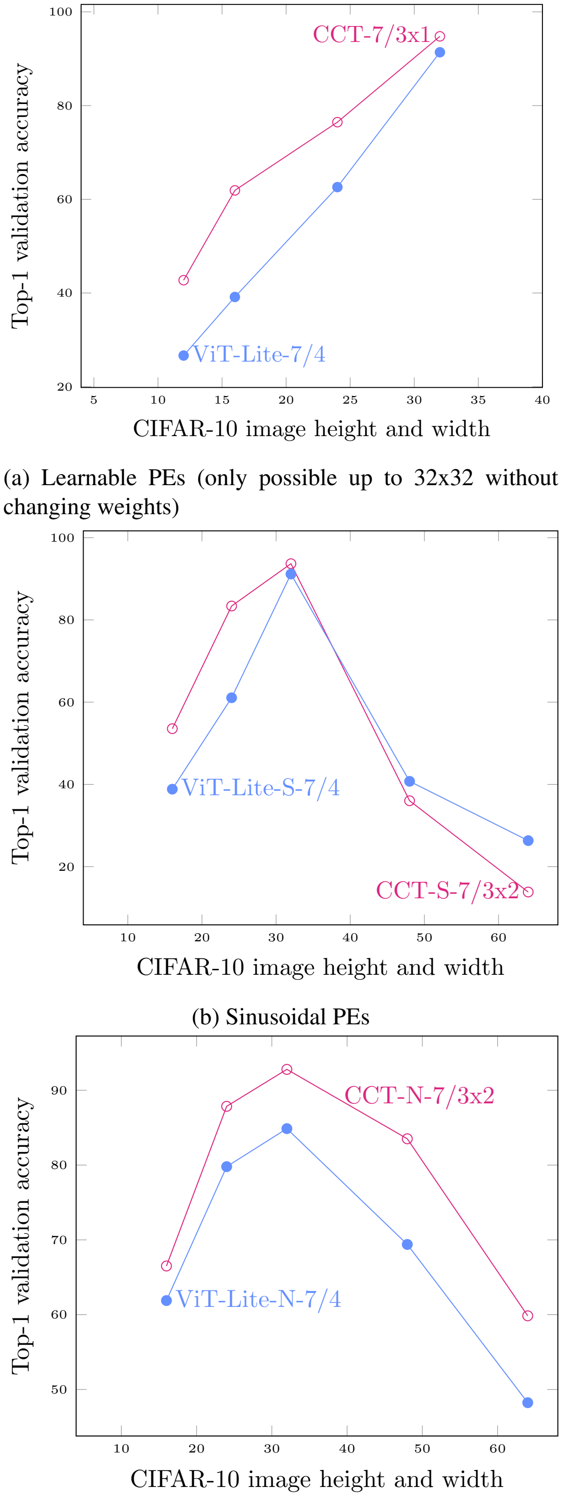


(a) Learnable PEs

(b) Sinusoidal Positional Embedding

(c) No Positional Embedding

Figure 6: CIFAR-10 resolution vs top-1% validation accuracy (training from scratch). Images are square.



(a) Learnable PEs (only possible up to 32x32 without changing weights)

(b) Sinusoidal PEs

(c) None

Figure 7: CIFAR-10 resolution vs top-1% validation accuracy (inference only). Images are square.