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, along with their unprecedented size and amounts of training data, many have come to believe that they are not suitable for small sets of data. This trend leads to great concerns, including but not limited to: 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 dispel the myth that transformers are "data hungry" and therefore can only be applied to large sets of data. We show for the first time that with the right size and tokenization, transformers can perform head-to-head with state-of-the-art CNNs on small datasets. Our model eliminates the requirement for class token and positional embeddings through a novel sequence pooling strategy and the use of convolutions. We show that compared to CNNs, our compact transformers have fewer parameters and MACs, while obtaining similar accuracies. Our method is flexible in terms of model size, and can have as little as 0.28M parameters and achieve reasonable results. It can reach an accuracy of 95.29 % when training from scratch on CIFAR-10, which is comparable with modern CNN based approaches, and a significant improvement over previous Transformer based models. Our simple and compact design democratizes transformers by making them accessible to those equipped with basic computing resources and/or dealing with important small datasets. Our method works on larger datasets, such as ImageNet (80.28% accuracy with 29% parameters of ViT), and NLP tasks as well.

1. Introduction

Transformers have rapidly been increasing in popularity and become a major focus of modern machine learning research. With the advent of Attention is All You Need[1],

---

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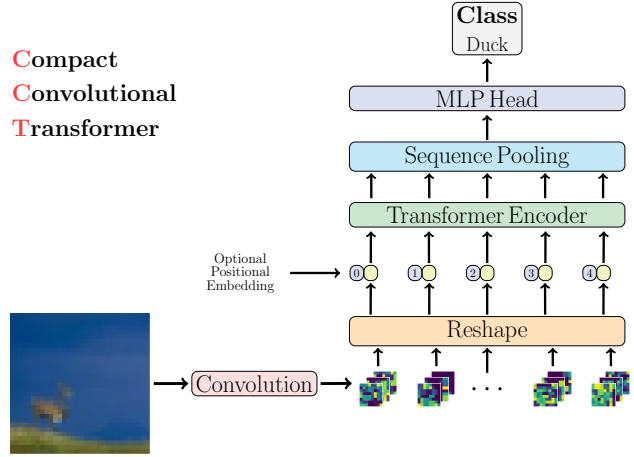

Figure 1: Overview of **CCT**: the convolutional variant of our compact transformer models. CCT can be quickly trained from scratch on small datasets, while achieving high accuracy (in under 30 minutes one can get 90% on an NVIDIA 2080Ti GPU or 80% on an AMD 5900X CPU on CIFAR-10 dataset). With CCT the class token becomes unnecessary and positional embedding becomes optional, resulting in only slightly different accuracy.

the community saw an explosion in transformer 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) [2] was the first major demonstration of a pure transformer backbone being applied to computer vision tasks. With this explosion in research, we've also seen an explosion in model sizes and datasets. It has become a common understanding that transformers are data hungry models and that they need sufficiently large datasets to perform as well or better than their convolutional counterparts. ViT 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.*".

This "data hungry" paradigm has made transformers 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. 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. 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 has low bias and is appropriate for conventional deep 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 [3]. Additionally, it has been shown that performing strongly on ImageNet does not necessarily mean that the model will perform strongly in other domains, such as medicine [4]. A non-"data hungry" transformer would allow these domains to utilize the advantages of long range interdependence that self-attention provides within their research and applications. This makes it important to build more efficient models that can be effective in less data intensive domains and allow for datasets that are orders of magnitude less than those conventionally seen in computer vision and NLP problems.

Additionally, the requisite of large data results in a requisite of large computational resources and this subsequently prevents many researchers from being able to provide insight. This not only limits the ability to apply models in different domains, but also limits reproducibility, making our field vulnerable to the reproducibility crisis. Verification of state of the art machine learning algorithms should not be limited to those with large infrastructures and computational resources. Throughout the history of research, we have learned that the speed of scientific advancement is directly proportional to the number of researchers, making it essential to enable researchers to participate in all areas of research.

On the other end of the spectrum, Convolutional Neural Networks (CNNs) [5] became the standard for computer vision, since the success of AlexNet [6]. AlexNet showed that convolutions are adept at vision based problems due to their invariance to spacial translations as well as having low relational inductive bias. He *et al*. [7] 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* [8]. Translation equivariance and translational invariance are due to the convolutions and pooling layers, respectively [8, 9]. They allow CNNs to leverage natural image statistics and subsequently allow models to have higher sampling efficiency [10, 10]. Conventionally, CNNs have commonly been used for works on smaller datasets because they have been shown to be the best performers in this regime. Additionally, CNNs are more efficient, both computationally and in terms of memory, when compared to transformers. They require less time and data to train while also requiring a lower number of parameters to accurately fit data. The above properties lead them to be effective for these areas, but they come with limitations in long range interdependence that attention mechanisms provide.

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, and provide competitive results.

In this paper we introduce ViT-Lite, a smaller and more compact version of ViT, which can obtain over 90% accuracy with correct patch sizing. We expand on ViT-Lite by adding using our novel sequence pooling and forming the Compact Vision Transformer (CVT), and further iterate by adding convolutional blocks to the tokenization step and thus creating the Compact Convolutional Transformer (CCT). Both of these additions add to significant increases in performance, leading to a top-1%accuracy of 94.72% on CIFAR-10. This is only slightly behind ResNet1001-v2, which obtains an accuracy of 95.38% but with $2.7\times$ model size and $1.6\times$ GMACs comparing to ours. Our model outperforms all previous transformer based models within this domain. 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, we achieve 80.28% accuracy while still maintaining a small number of parameters and reduced computation. Our model outperforms ViT while containing less than a third of the number of parameters with about 10% of the computational complexity (MACs). This demonstrates the scalability of our model while maintaining

compactness and computational efficiency.

The main contributions of this paper are:

- Dispelling the myth of "data hungry" transformers by introducing ViT-Lite, which can be trained from scratch effectively and achieve high accuracy on small datasets such as CIFAR-10.

- Introducing Compact Vision Transformer (CVT) with a novel sequence pooling stratgey, which removes the need of the conventional class token design in vision transformers and make it more accurate.

- 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 extremely 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 we can train on the majority of GPUs, even if researchers do not have access to top of the line hardware. Through these efforts, we aim to democratise transformer research and make it more accessible to the average person.

2. Related Works

Since the advent of deep neural networks, CNNs have predominantly been used for visual recognition. In NLP based research, attention mechanisms [11, 12, 13] gained popularity for their ability to weigh different features within sequential data. In the past, many researchers leveraged a combination of attention and convolutions in neural networks for visual tasks [14, 15, 16, 17, 18]. Ramachandran *et al*. [19] introduced the first a vision model that relies purely on an attention mechanism. Following this success, transformers have been used in a wide variety of tasks, including: machine translation [20, 21, 22], visual question answering [23, 24], action recognition [25, 26], and the like. Dosovitskiy *et al*. [2] introduced the first stand-alone transformer based model (ViT) for image classification. In the following subsections we briefly revisit ViT and several related works.

2.1. Vision Transformer

Vision Transformers (ViT) were introduced as a way to compete with CNNs on image classification tasks and utilize the benefits of attention within these networks. The motivation was because attention has many desirable properties for a network, specifically its ability to make long range connections. Dosovitskiy *et al*. showed 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 the 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 and orders them from top left to bottom right. 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, using a linear layer to obtain the token embeddings. This simple patching method has a few limitations, in particular information is lost along the boundary regions.

Positional Embedding: Positional embedding is a method that adds spatial information into the network. Since the model does not actually know anything about the spatial relationship between tokens we must add some extra information. 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.

Classification Token: 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 the latent information that is later used for classification. This `[class]` token design originated from BERT [20].

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 (MSA) 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: Similar to BERT [20], the position of the `[class]` token yields the location of the image representation. A linear layer is then applied to this representation to obtain a confidence score for each class.

2.2. Data-Efficient Transformers

In an effort to reduce the dependence upon data, Data-Efficient Image Transformers (DeiT) [27] use knowledge distillation to improve the classification performance of the vision transformer. This work builds off of Xu *et al*. [28], which optimizes transformers for smaller linguistic datasets by introducing a novel initialization technique inspired by Huang *et al*. [29]. DeiT's work focuses on reducing the

data dependence of vision transformers, showing that this model could perform well on mid-sized dataset like ImageNet, as well as fine tune on small datasets like CIFAR-10. This work pushes forward accessibility of transformers in smaller dataset domains but makes an assumption that pre-trained models are accessible and that the smaller datasets share enough attributes for fine tuning to work. However, if a small dataset happens to be sufficiently novel, then the pre-trained model will not help train on that domain and the model will not be appropriate for that dataset.

Yuan *et al.* [30] proposed Tokens-to-token ViT (T2T-ViT), which adopts a layer-wise tokenization strategy in which overlapping image patches are flattened, sent through self-attention layers, and reshaped back into patches. This strategy allows their network to model local structures, including along the boundaries between patches. Additionally, this process can be repeated multiple times and the final outputs are flattened into a sequence of vectors. This repetition of attention allows for a feature rich and learnable tokenization that allows T2T-ViT to obtain high accuracy 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.

2.3. Other Concurrent Works

Many concurrent works have been recently released on arXiv to improve vision transformers and eliminate the need to pretrain on super large datasets like JFT-300M [31] that is not accessible by the large community. ConViT [32], make an effort to combine the strength of CNNs and transformer based models. They introduce a *gated positional self-attention* (GPSA) that allows for a "soft" convolutional inductive bias within their model. This GPSA allows their network to have more flexibility with respect to positional information. Since their 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. Their *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) [33] uses a convolutions to extract low level features. They use an image-to-token module to extract this information, introduce a locally enhanced feed-forward layer that processes the spatial information form the extracted tokens, and utilize a layer-wise cross attention mechanism. This allows them to create a network that is competitive with DeiT on ImageNet. Convolutional vision Transformer (CvT) [34] also uses a convolutional layer for the tokenization process. Wu *et al.* uses these convolutional projections instead of the standard linear projections for their embeddings into the encoding layer of the transformer. This allows them to achieve competi-

tive results to DeiT. 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 but our work 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. Methodology

In order to dispel the myth that vision transformers are only applicable when dealing with large sets of data, we propose three different models: ViT-Lite, Compact Vision Transformers (CVT), and Compact Convolutional Transformers (CCT). ViT-Lite is near identical to the original ViT, but with more suitable smaller patch sizing for small datasets. In other words, *an image is **not always** worth 16×16 words* [2] and a suitable size matters. CVT builds on this by introducing a sequential pooling method, called SeqPool, that pools the sequential based information that results from the transformer encoder. SeqPool eliminates the need for the extra Classification Token(Sec 3.3). Lastly, we have CCT which introduces a convolutional based patching method, which preserves local information and is able to encode relationships between patches, unlike the original ViT (Apache License 2.0). A detailed modular-level comparison of these models can be viewed in Figure 2.

Our convolution based model, CCT, consist of convolutions with small strides that allow for efficient tokenization and preserves local spatial relationships. Furthermore, we eliminate the need for the [class] token in CCT (also eliminated in CVT), allowing for greater flexibility within our model. While simple in design, we show that our network can be quite effective if composed in the correct manner. To further differentiate the subtle differences between CVT and CCT we provide Figure 3 which outlines the similarities and differences of the networks. The components of our compact transformers are further discussed in the following subsections.

3.1. Convolutional Block

In order to introduce an inductive bias into the model, we replace the "image patching" and "embedding" layers in ViT with a simple convolutional block. Our convolutional
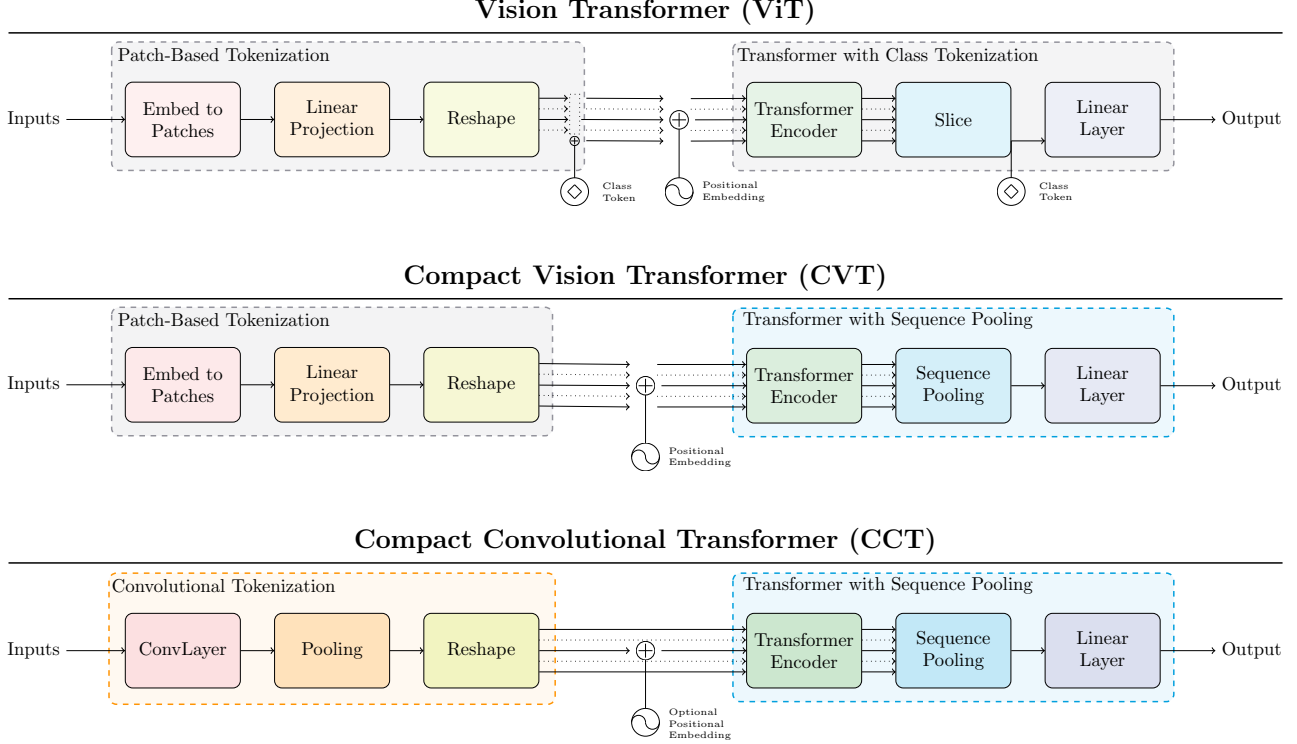
Figure 2: Comparing ViT (top) to CVT (middle) and CCT (bottom)

block follows conventional design, which consists of a single convolution layer, $ReLU$ activation, and a max pool operation. Given an image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$:

$$\mathbf{x}_0 = \text{MaxPool}(\text{ReLU}(\text{Conv2d}(\mathbf{x}))) \qquad (1)$$

where the Conv2d operation has d filters, same number as the embedding dimension of the transformer backbone. The number of these blocks can be to more than 1. When iterating this process, we use 64 filters for the earlier convolution operations, and d filters for the final one. By using this convolutional block we gain some added flexibility over models like ViT. While ViT patches the image, we seek to use convolutions to embed the image into a latent representation that will be more efficient for the transformer.

Changes in image size do not affect the number of parameters, but do affect the sequence length and subsequently the amount of computation needed. Although, ViT's patching requires that both the image height and width be divisible by the patch size. This allows us to take as input different size data without the need for cropping or padding. In other words, our CCT model does not have the requirement of evenly subdividing an image into patches, as ViT or CVT. Additionally, the convolution and max pool operations can be overlapping, which also increases the sequence length, but on the other hand, could increase performance by injecting inductive bias. Having these convolu-

tional patches allows our model to maintain locally spatial information. This also gives us more flexibility toward removing the positional embedding in our model, as it manages to maintain a very good performance. This is further discussed in Appendix A.1.

3.2. Transformer-based Backbone

In terms of model design, we follow the original Vision Transformer [2], and original Transformer [1], relatively closely. As mentioned in Section 2.1, the encoder consists of transformer blocks, each including a MSA layer and a MLP head. Similar to ViT, we apply layer norm after the positional embeddings are added. The transformer encoder uses Layer Normalization, $GELU$ activation, and dropout. The positional embeddings are learnable by default.

3.3. SeqPool

In order to map the sequential outputs to a singular class index, instead of using a class token (like BERT [20], ViT [2], and most other transformer-based classifiers), we propose **Sequence Pool**ing. As the name suggests, we pool the outputs of the transformer backbone across the sequence. We pool over the entire sequence of data since it contains relevant information across different parts of the input image, making our model compact. We can think of this operation as the mapping transformation $T : \mathbb{R}^{b \times n \times d} \mapsto \mathbb{R}^{b \times d}$.
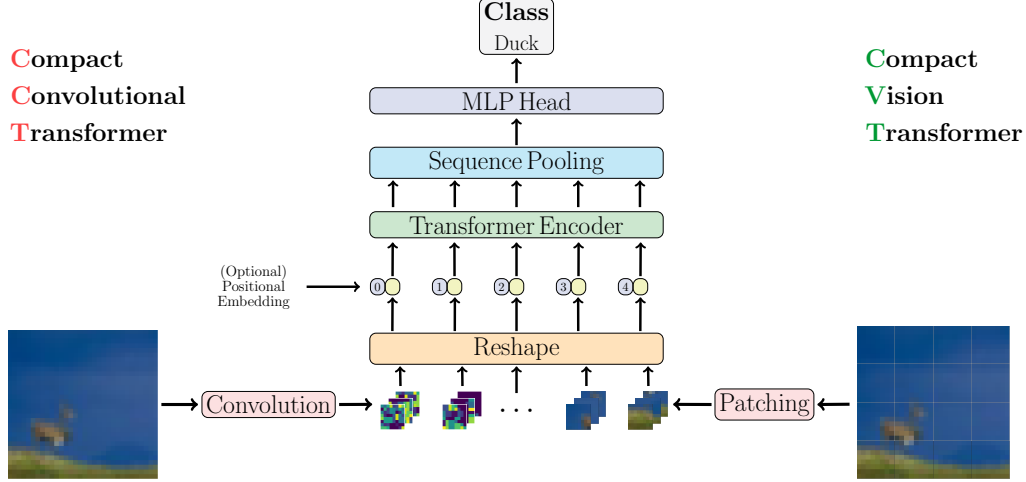
Figure 3: Overview of CCT vs CVT

Given:

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

where $\mathbf{x}_L$ or $\mathrm{f}(\mathbf{x}_0)$ is the output of an L layer transformer encoder, b is the mini-batch size, n is the sequence length, and d is the embedding dimension, we feed $\mathbf{x}_L$ to a linear layer $\mathrm{g}(\mathbf{x}_L) \in \mathbb{R}^{d \times 1}$ and apply softmax activation:

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

Then, we simply compute:

$$\mathbf{z} = \mathbf{x}'_L \mathbf{x}_L = \mathrm{softmax}\left(\mathrm{g}(\mathbf{x}_L)^T\right) \times \mathbf{x}_L \in \mathbb{R}^{b \times 1 \times d} \quad (2)$$

and by pooling the second dimension we are left with $z \in \mathbb{R}^{b \times d}$. Then this output can be sent through a linear classifier, like most other such networks.

This SeqPool allows for our network to weigh the sequential embeddings of the latent space produced by the transformer encoder and better 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. We have tested several variations of this pooling method, including learnable and static methods, and found that the learnable pooling performs the best. We believe that the learnable weighting is more efficient because each embedded patch does not contain the same amount of entropy. This allows our model to give higher weight to patches that contain more information relevant to the classifier. Additionally, the 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).

3.4. Small and Compact Models

We propose some smaller and more compact transformer models compared to ViT, and use them alongside the sizes that they've already proposed. In our notation, we use the number of layers to specify size, as well as the kernel size for our convolutional layers: for instance, CCT-12/7×2 means that the variant's transformer encoder has 12 layers, and uses 2 convolutional blocks with 7×7 convolutions. We follow the same format for other models: ViT-Lite, which is like ViT but with smaller patches and CVT which is like ViT but with SeqPool (making it compact). We further summarize these models in Appendix D.

4. Experiments

We conducted image classification experiments using our method on the following datasets: CIFAR-10, CIFAR-100 (MIT License), MNIST, and Fashion-MNIST. These 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. We also perform a smaller study on how our model performs on medium sized datasets by using ImageNet. Finally, we also include a study on NLP classification, shown in section 4.5.

In all experiments, we conducted a hyperparamter search for every different method and report the best results we were able to achieve. We provide a detailed report on the hyperparamter settings in Appendix B. All reported top-1% accuracy values are best out of 4 runs. Unless stated otherwise, all tests were run for 200 epochs with a batch size of 128, and the learning rate is reduced per epoch based on cosine annealing [35]. Label smoothing with a probability of 0.1 was also used during training. All methods are warmed up for 10 epochs. In all experiments (unless stated otherwise), AutoAugment [36] (MIT License) is used for CIFAR-10 and CIFAR-100.

Model	CIFAR-10	CIFAR-100	Fashion-MNIST	MNIST	# Params	MACs
Convolutional Networks (Designed for ImageNet)						
ResNet18	90.27%	63.41%	93.51%	99.18%	11.18 M	0.04 G
ResNet34	90.51%	64.52%	93.47%	99.24%	21.29 M	0.08 G
ResNet50	90.60%	61.68%	93.17%	99.18%	23.53 M	0.08 G
MobileNetV2/0.5	84.78%	56.32%	93.49%	99.08%	0.70 M	< 0.01 G
MobileNetV2/1.0	89.07%	63.69%	93.62%	99.28%	2.24 M	0.01 G
MobileNetV2/1.25	90.60%	65.24%	93.83%	99.25%	3.47 M	0.01 G
MobileNetV2/2.0	91.02%	67.44%	94.07%	99.38%	8.72 M	0.02 G
Convolutional Networks (Designed for CIFAR)						
ResNet56	94.63%	74.81%	95.25%	99.27%	0.85 M	0.13 G
ResNet110	95.08%	76.63%	95.32%	99.28%	1.73 M	0.26 G
ResNet164-v1[37]	94.07%	74.84%	–	–	1.70 M	0.26 G
ResNet164-v2[37]	94.54%	75.67%	–	–	1.70 M	0.26 G
ResNet1001-v1[37]	92.39%	72.18%	–	–	10.33 M	1.55 G
ResNet1001-v2[37]	95.08%	77.29%	–	–	10.33 M	1.55 G
ResNet1001-v2†**[37]**	95.38%	–	–	–	10.33 M	1.55 G
Vision Transformers						
ViT-12/16	76.42%	46.61%	83.88%	82.66%	85.63 M	0.43 G
ViT-Lite-7/16	76.52%	48.25%	82.53%	81.37%	3.89 M	0.02 G
ViT-Lite-6/16	76.47%	48.54%	82.06%	81.15%	3.36 M	0.02 G
ViT-Lite-7/8	87.49%	63.52%	90.73%	98.9%	3.74 M	0.06 G
ViT-Lite-6/8	87.50%	62.76%	90.44%	98.9%	3.22 M	0.05 G
ViT-Lite-7/4	91.38%	69.74%	94.39%	99.29%	3.72 M	0.24 G
ViT-Lite-6/4	90.94%	69.2%	94.35%	99.26%	3.19 M	0.21 G
Compact Vision Transformers						
CVT-7/8	89.66%	66.02%	92.47%	99.05%	3.74 M	0.06 G
CVT-6/8	89.03%	65.71%	92.52%	99.11%	3.22 M	0.05 G
CVT-7/4	92.43%	73.01%	94.55%	99.32%	3.72 M	0.24 G
CVT-6/4	92.58%	72.25%	94.54%	99.27%	3.19 M	0.20 G
Compact Convolutional Transformers						
CCT-2/3×2	89.17%	66.90%	93.90%	99.21%	0.28 M	0.03 G
CCT-4/3×2	91.45%	70.46%	93.99%	99.18%	0.48 M	0.05 G
CCT-6/3×2	93.56%	74.47%	94.23%	99.25%	3.33 M	0.24 G
CCT-7/3×2	93.65%	74.77%	93.95%	99.26%	3.85 M	0.28 G
CCT-7/7×1	92.57%	73.01%	93.53%	99.19%	3.74 M	0.25 G
CCT-7/3×1	94.47%	75.59%	94.40%	99.24%	3.76 M	0.95 G
CCT-7/3×1†	94.78%	77.05%	94.48%	99.21%	3.76 M	0.95 G
CCT-6/3×1†	94.81%	76.71%	94.60%	99.23%	3.17 M	0.81 G
CCT-6/3×1◇	95.29%	77.31%	94.44%	99.37%	3.17 M	0.81 G

Table 1: Top-1 validation accuracy comparisons. ResNet164 and ResNet1001 were reported directly from the paper, while the rest were trained and tuned using a single training script. † variants used a batch size of 64 instead of 128. ◇ variants used lower weight decay ($1e-4$) and were run for 500 epochs.

Model	Top-1	# Params	MACs	Training Epochs
ResNet152*[7]	78.57%	60.19 M	11.58 G	≈ 120
ViT-Base/16 [2] $\uparrow$ 384	77.91%	86.83 M	49.35 G	300
CCT-14/7×2$^\diamond$	80.04%	22.29 M	5.11 G	300$^\heartsuit$
CCT-16/7×2$^\diamond$	80.28%	25.32 M	5.69 G	300$^\heartsuit$

Table 2: ImageNet Top-1 validation accuracy comparisons. * uses 10-Crop testing. $\uparrow$ indicates that ViT first trained on image sizes of 224 and then fine-tuned on sizes 384 for better accuracy. $\diamond$ means the methods were trained with extra augmentations (see appendix for details); $\heartsuit$ means we trained for 300 epochs + 10 cooldown epochs.

4.1. Computational Resources

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

4.2. Performance Comparison

In order to argue 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 [7], which are still very useful CNNs for small to medium amounts of data, as well as to MobileNetV2 [38], which are very compact and small-sized CNNs. We also compare with results from [37] where He *et al.* designed very deep (up to 1001 layers) CNNs specifically for CI-FAR. The results are presented in Table 1. We highlight the top performers. CCT-6/3x1 achieves on par results with the CNN models while having significantly fewer parameters.

Other than CNNs, we also compare our method to the original ViT [2] in order to express the effectiveness of smaller sized backbones, convolutional layers, as well our pooling technique. It should be noted that only one row represents ViT according to the original paper, and the rest of the rows are simply "smaller" ViT variants according to our own models which are denoted as ViT-Lite. Following ViT's original notation, we denote ViT variants with their patch size, but use the number of layers instead of the nominal descriptions ("Base", "Large", "Huge"). For instance, ViT-12/16 is a ViT with a 12 layer transformer encoder with 16×16 patches. ViT-Base is equivalent to our model with size 12 in terms of the transformer encoder, and the number after the forward slash is the patch size (for our variants it's the convolution kernel size).

We also present ImageNet training results in Table 2. We should note that ResNet152 [7] used 10-crop testing, and ViT has more than 3 times as many parameters than our variants. We used the timm package [39] to train our models on ImageNet (see Appendix B). We also note that ViT first trains on images of size 224, similar to us, but fine-
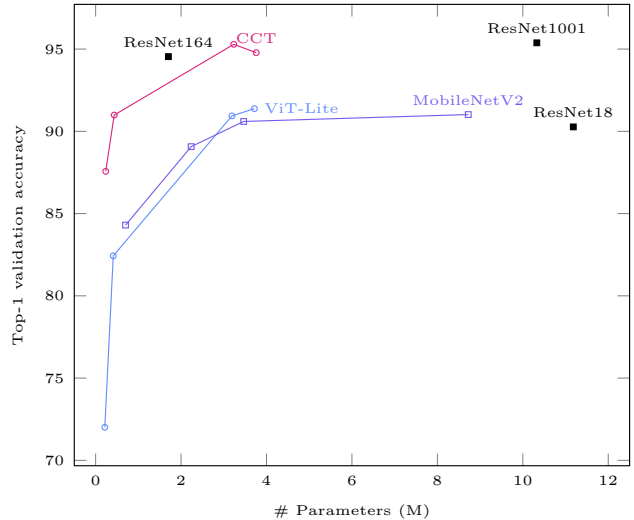

Figure 4: CIFAR-10 accuracy vs model size (sizes $<$ 12M).

tunes on images of size 384. They note that this increase in image size improves performance, likely due to a higher information density.

4.3. Model Sizing

We also evaluate smaller variants of our model. These variants can have as little as 284 K parameters and still achieve a considerably good performance on these datasets. These results are also presented in Table 1.

4.4. Ablation Study

We extend our previous comparisons by doing an ablation study on our method. We do so by progressively transforming the original ViT in structure into CCT, with intermediate models called ViT-Lite and CVT, and comparing the accuracy between these models. In this particular study, we report the results (best out of 4 runs) on CIFAR-10 and CIFAR-100 in Table 8 in the appendix.

The "Model" column refers to model size (see Table 6 for details). The column "Conv" specifies the number of convolutional blocks used, and "Conv Size" specifies the kernel size. "Aug" denotes the use of AutoAugment [36].

Model	AG News	TREC	SST	IMDb	DBpedia	# Params
Vanilla Transformer Encoders						
Transformer-2	93.28%	90.40%	67.15%	86.01%	98.63%	1.037 M
Transformer-4	93.25%	92.54%	65.20%	85.98%	96.91%	2.071 M
Transformer-6	93.55%	92.78%	65.03%	85.87%	98.24%	4.136 M
Vision Transformers						
ViT-Lite-2/1	93.02%	90.32%	67.66%	87.69%	98.99%	0.227 M
ViT-Lite-2/2	92.20%	90.12%	64.44%	87.39%	98.88%	0.264 M
ViT-Lite-2/4	90.53%	90.00%	62.37%	86.17%	98.72%	0.337 M
ViT-Lite-4/1	93.48%	91.50%	66.81%	87.38%	99.04%	0.416 M
ViT-Lite-4/2	92.06%	90.42%	63.75%	87.00%	98.92%	0.453 M
ViT-Lite-4/4	90.93%	89.30%	60.83%	86.71%	98.81%	0.526 M
ViT-Lite-6/1	93.07%	91.92%	64.95%	87.58%	99.02%	3.087 M
ViT-Lite-6/2	92.56%	89.38%	62.78%	86.96%	98.89%	3.160 M
ViT-Lite-6/4	91.12%	90.36%	60.97%	86.42%	98.72%	3.307 M
Compact Vision Transformers						
CVT-2/1	93.24%	90.44%	67.88%	87.68%	98.98%	0.227 M
CVT-2/2	92.29%	89.96%	64.26%	86.99%	98.93%	0.264 M
CVT-2/4	91.10%	89.84%	62.22%	86.39%	98.75%	0.337 M
CVT-4/1	93.53%	92.58%	66.64%	87.27%	99.04%	0.416 M
CVT-4/2	92.35%	90.36%	63.90%	86.96%	98.93%	0.453 M
CVT-4/4	90.71%	90.14%	61.98%	86.77%	98.80%	0.526 M
CVT-6/1	93.38%	92.06%	65.94%	86.78%	99.02%	3.087 M
CVT-6/2	92.57%	91.14%	64.57%	86.61%	98.86%	3.160 M
CVT-6/4	91.35%	91.66%	61.63%	86.13%	98.76%	3.307 M
Compact Convolutional Transformers						
CCT-2/1x1	93.40%	90.86%	**68.76 %**	88.95%	99.01%	0.227 M
CCT-2/2x1	93.38%	91.86%	67.19%	**89.13 %**	99.04%	0.264 M
CCT-2/4x1	**93.80 %**	91.42%	64.47%	88.92%	99.04%	0.337 M
CCT-4/1x1	93.49%	91.84%	68.21%	88.71 %	99.03 %	0.416 M
CCT-4/2x1	93.30%	**93.54 %**	66.42%	88.94%	**99.05 %**	0.453 M
CCT-4/4x1	93.09%	93.20%	66.57%	88.86%	99.02%	0.526 M
CCT-6/1x1	93.73%	91.22%	66.59%	88.81%	98.99%	3.087 M
CCT-6/2x1	93.29%	92.10%	65.02%	88.74%	99.02%	3.160 M
CCT-6/4x1	92.86%	92.96%	65.84%	88.68%	99.02%	3.307 M

Table 3: 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.

"Tuning" specifies a minor change in dropout, attention dropout, and/or stochastic depth. ViT, by default, uses a dropout for the MLP heads only with a probability of 0.1, and no attention dropout. Conversely, we do not use MLP dropout and only use attention dropout ($p = 0.1$). We also use stochastic depth with a drop probability of 0.1.

The first row in Table 8 are essentially ViT with the same settings from the paper. In the second row, the optimizer has been changed to weighted Adam, which we found more useful for training vision transformers. 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.

4.5. 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 [40], TREC [41], SST [42], IMDb [43], DBpedia [44]. The results are summarized in Table 3. As can be seen our model out performs the transitional transformer encoder, demonstrating that the techniques we use here also help similarly with NLP tasks. The network is slightly modified from the vision CCT. We use GloVe (Apache License 2.0) [45] 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 this we can see that 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 each of them.

5. Conclusion

The myth of "data hungry" transformers has persisted since they were introduced. We have shown within this paper that this is not necessarily true, and that with proper configuration, a transformer can be just as efficient as state-of-the-art CNN image classifiers. We democratise the use of Transformers by providing a framework that allows for these SOTA results on small datasets and minimal hardware. Our method is flexible in size, and the smallest of our variants can be easily loaded on even a minimal GPU, or even a CPU. While the trend of research has been bigger and bigger transformers, we show that there is still much research to be done to make efficient networks that work on small datasets and less efficient hardware. This kind of research is extremely important for many scientific domains where data is far more limited that the conventional datasets we use for general research. Continuing research in this direction will help open research up to more people and domain, helping democratize machine learning research.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008, 2017. 1, 5

[2] 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, 3, 4, 5, 8

[3] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *CoRR*, abs/1911.02685, 2019. 2

[4] 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*, April 2021. 2

[5] 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. 2

[6] 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. 2

[7] 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. 2, 8

[8] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. MIT press Cambridge, 2016. 2

[9] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015. 2

[10] Daniel L Ruderman and William Bialek. Statistics of natural images: Scaling in the woods. *Physical review letters*, 73(6):814, 1994. 2

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

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

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

[14] 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

[15] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3464–3473, 2019. 3

[16] 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

[17] 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

[18] 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

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

[20] 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

[21] 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

[22] 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

[23] 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

[24] 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

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

[26] 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

[27] 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

[28] Peng Xu, Wei Yang, Wenjie Zi, Keyi Tang, Chengyang Huang, Jackie Chi Kit Cheung, and Yanshuai Cao. Optimizing deeper transformers on small datasets: An application on text-to-sql semantic parsing. *arXiv preprint arXiv:2012.15355*, 2020. 3

[29] 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. 3

[30] 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, 14

[31] 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

[32] 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

[33] 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

[34] 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

[35] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *ICLR*, 2017. 6

[36] 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. 6, 8

[37] 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, 14

[38] 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

[39] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. 8, 14

[40] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*, 2015. 10

[41] Xin Li and Dan Roth. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002. 10

[42] 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. 10

[43] 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. 10

[44] 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. 10

[45] 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. 10

[46] 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. 14

[47] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 14

[48] 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. 14

[49] 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. 14

A. Additional analyses

Within this appendix we present some additional performance analyses that we have done. We focus this work on the positional embedding and the relationship between performance and the dataset size.

A.1. Positional Embedding

To determine the effects of the convolution layers and SeqPool, we perform an ablation study just for the purpose of positional embedding, seen in Table 4. In this study we look at ViT, ViT-Lite, CVT, and CCT, and investigate the effects of: a learnable positional embedding, the 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 B). 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 can be safely removed much impact in accuracy. We also tested our CCT model without SeqPool, in place using the standard class token, and found that there was little to no effect from having a positional encoder or not, depending on model size. This demonstrates that the convolutions are the feature that helps provide spatially sparse information to the transformer as well as helps our 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 that we can increase the overall performance and maintain a low dependence on positional information. This demonstrates that our model is more robust to sparse data and that users may not even be required to positionally embed their datasets at all.

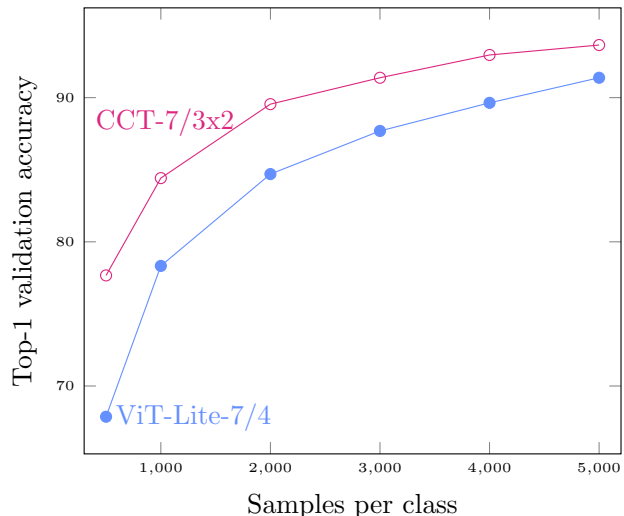### A.2. Performance vs Dataset Size

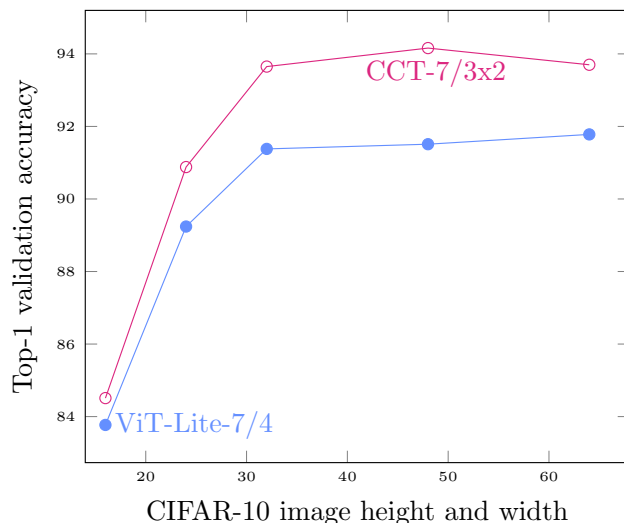Figure 5: Reduced # samples / class (CIFAR-10)

Figure 6: Image Size vs Accuracy (CIFAR-10)

In this experiment we reduce the size of CIFAR-10 to determine the relationship between our model's performance and the number of samples within a dataset. This experiment disassociates the dimensionality of the data from the size of the dataset, measuring which metric is the "data hun-

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	93.65% *(+0.86%)*	74.77% *(+1.34%)*
	Sinusoidal	93.67% *(+0.88%)*	74.49% *(+1.06%)*
	None	92.79% *(baseline)*	73.43% *(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% *(+1.4%)*
	Sinusoidal	89.84% *(+0.92%)*	64.71% *(−0.46%)*
	None	88.92% *(baseline)*	65.17% *(baseline)*

Table 4: Top-1 validation accuracy comparison when changing the positional embedding method. The numbers reported are best out of 4 runs with random initializations. † denotes model trained with extra augmentation and hyperparameter tuning.

gry" aspect of transformers. To do this we remove samples, uniformly, from each class in CIFAR-10 and measure the performance of both ViT and CCT. In Figure 5 we see the comparison of each model's accuracy vs the number of samples per class. We show how the model runs when we have 500, 1000, 2000, 3000, 4000, or 5000 (original) samples per class, meaning the total data set ranges from one tenth the size to full. We can see here that our model 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. This allows us to conclude that our model is not data hungry and can still effectively learn on very small datasets, obtaining over 84% accuracy with a dataset that has only 10k samples and a total of 10 classes.

A.3. Performance vs Dimensionality

In order to determine if transformers need highly dimensional data, as opposed to the number of samples (explored in Appendix A.2) we again use CIFAR-10 and downsample or upsample the images to determine this relationship. In Figure 6, we show the image dimensionality vs the performance of our networks, where we trained both CCT and ViT-Lite with images of sizes ranging from 16×16 to 64×64. We can see that CCT performs better on all image sizes, with a widening difference as the number of pixels increases. From this we can infer that our model is able to better utilize the information density of an image, where ViT does not see continued performance increases after the standard 32x32 size.

We go into further detail about the effects of positional embeddings in Appendix C, where it should be noted that

the difference seen here is exaggerated once positional embeddings are removed. Additionally, we show the difference between training a model with these parameters and inferring on a model that was trained on 32×32 sized images, denoting the generalizability of the models. We note that the number of parameters for these competing models is roughly the same, see Table 8.

B. Hyperparameter tuning

We tuned the hyperparamters per experiment and arrived at the following for each table in Sec. 4. We determined these hyper parameters through a parameter sweep. For ResNet (all models except ResNet164 and ResNet1001) and MobileNetV2 (all variants) we used a learning rate of 0.1, a weight decay of $1e-4$ and optimized using SGD with momentum (0.9). It should be noted that we directly report the numbers from the paper "Identity mappings in deep residual networks" [37] for ResNet164 and ResNet1001.

Hyper Param	ViT-12	ViT-7	CCT
Optimizer	AdamW	AdamW	AdamW
LR	1e-4	5e-4	5e-4
Weight Decay	0	3e-2	3e-2

Table 5: ViT and CCT hyperparamters.

Methods using AdamW used the default beta values $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

For CIFAR runs we use a learning rate of $5e-4$, weight decay of $3e-2$, a batch size of 128 (unless stated otherwise), and trained for 200 epochs. We note that our *best* performance modified these parameters slightly. For some models, a batch size of 64 performs slightly better than 128. By lowering weight decay to $1e-4$, using a batch size of 128 and increasing to 500 epochs, we achieve the best results on CIFAR-10. We found that reducing the weight decay slightly increases performance but greatly reduces to rate of convergence, taking approximately 380 epochs to achieve similar performance. This demonstrates that weight decay is still a useful parameter to investigate when hyperparameter tuning.

For our ImageNet runs, we used a learning rate of $5e-4$, weight decay of $5e-2$, batch size of 512, and trained for 300 epochs (+10 cool down epochs) on $8 \times$ NVIDIA RTX A6000 GPUs. We also followed [30] in applying CutMix [46], Mixup [47], Randaugment [48], and Random Erasing [49]. We used timm's ImageNet training script [39]. As for ViT-Base and ResNet152, we report their results directly from the original papers.

C. Dimensionality Experiments

Within this appendix we extend the analysis from Appendix A.3, showing the difference in performance when using different types of positional embedding. Figure 7 shows the difference of the accuracy when models are being trained from scratch. On the other hand, Figure 8 shows the performance difference when models are only used in inference and pre-trained on the 32×32 sized images. We note that in Figure 8(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 7(c) and Figure 8(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.

D. CCT Variants

Variants of our model are presented in Table 6.

Model	# Layers	# Heads	Ratio	Dim
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-14t	14	6	3	384
CCT-16t	16	6	3	384

Table 6: Transformer backbones used for CCT.

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

Table 7: Difference between **tuned** and not tuned runs in table 8

E. 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. Refer to section 4.4 for further discussion. We again note the substantial increase in performance moving from CVT to CCT.

| Model	Pool	# Conv	Conv Size	Aug	Tuning	CIFAR-10	CIFAR-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	0.24 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	0.24 G
CVT-7/4	SP	✗	✗	✓	✗	91.72%	69.59%	3.72 M	0.24 G
CVT-7/4	SP	✗	✗	✓	✓	92.43%	73.01%	3.72 M	0.24 G
CVT-7/2	SP	✗	✗	✗	✗	84.8%	57.98%	3.76 M	0.94 G
CCT-7/7×1	SP	1	7×7	✗	✗	87.81%	62.83%	3.74 M	0.25 G
CCT-7/7×1	SP	1	7×7	✓	✗	91.85%	69.43%	3.74 M	0.25 G
CCT-7/7×1	SP	1	7×7	✓	✓	92.29%	72.46%	3.74 M	0.25 G
CCT-7/3×2	SP	2	3×3	✓	✓	93.65%	74.77%	3.85 M	0.28 G
CCT-7/3×1	SP	1	3×3	✓	✓	**94.47%**	**75.59%**	3.76 M	0.95 G

Table 8: CIFAR Top-1 validation accuracy when transforming ViT into CCT step by step.
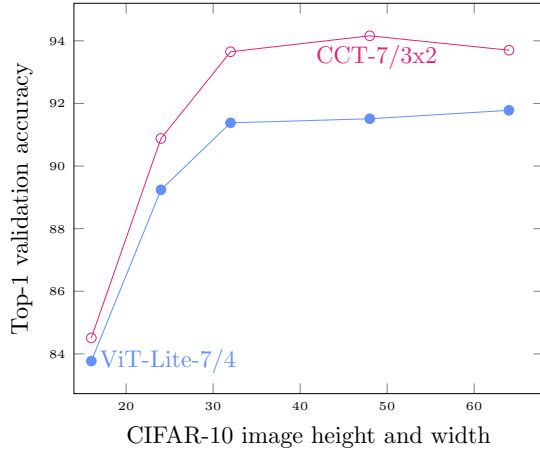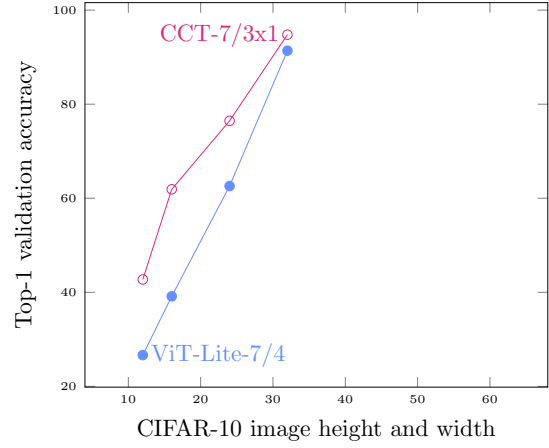
(a) Learnable PEs
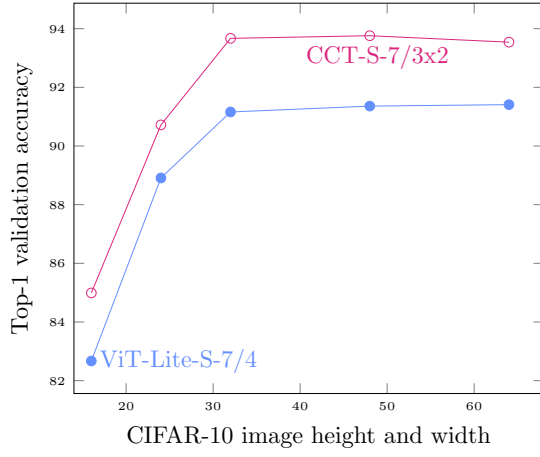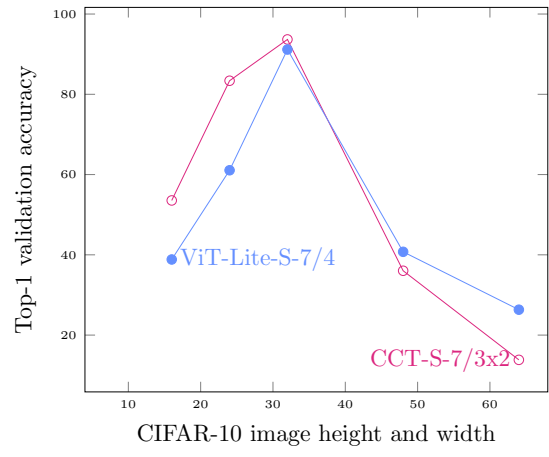

(b) Sinusoidal Positional Embedding
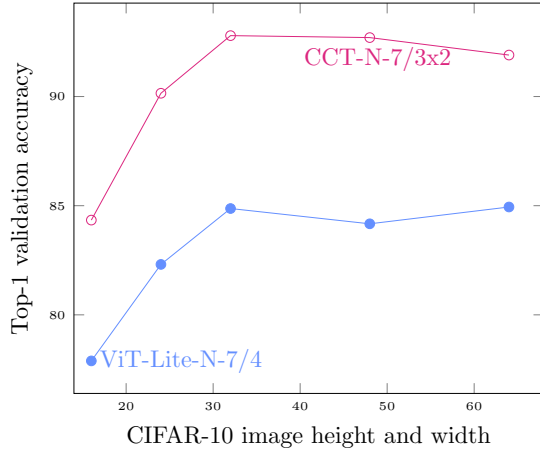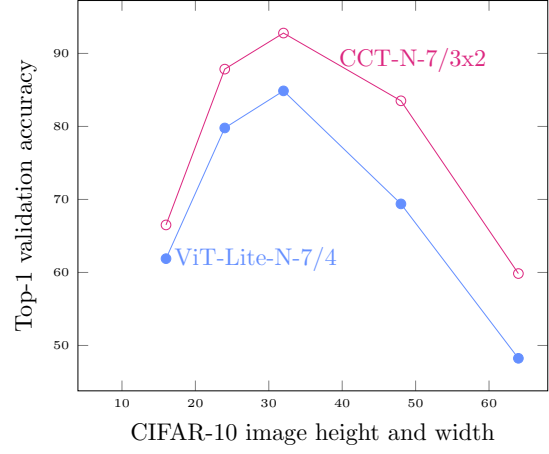

(c) No Positional Embedding

Figure 7: CIFAR-10 resolution vs top-1% validation accuracy (training from scratch). Images are square.

(a) Learnable PEs

(b) Sinusoidal PEs

(c) None

Figure 8: CIFAR-10 resolution vs top-1% validation accuracy (inference only). Images are square.