# Escaping the Big Data Paradigm with Compact Transformers - Paper Implementation

Shreyas Bhat

August 15, 2021

**Abstract**

In the recent past, Transformers have advanced in computer vision and brought competitive results in standard computer vision tasks. While this is great, it comes with its constraints. The major bottleneck being to train a transformer from scratch with small datasets that are accessible to the public. The authors in the paper make changes in the original Vision Transformer which competes with the state of the art CNNs on small datasets for classification tasks and so on. In this project I try to implement the main claims of the paper and record its performance.

## 1 Introduction

The Vision Transformer has seen great success ever since its introduction in the paper by **Alexey Dosovitsky et al.**, However, The authors of the paper - **Hassani et al.** dealt with this issue in transformers which is that it requires high training data and is computationally expensive. Transformers also lack some of the inductive biases which come out during the convolutional procedures such as preserving local information and therefore are not suitable when trained on insufficient data. Datasets such as ImageNet which are considered to be large scale are no longer considered to be and datasets with more data such as JFT-300M curated by Google is not publicly available and requires multiple TPUs for training. The authors revisit the vision transformer and proposes the following variants.

ViT-Lite is a smaller Vision Transformer which uses smaller patch-size of 6x6 as compared to the patch-size 16x16 which is mentioned in the original paper and has lesser MLP heads(2) and lesser transformer layers. In Convolutional Vision Transformer, A novel sequence pooling is added on top of this which maps sequential outputs to a single class token. This replaces the original class tokens used to classify images. Further adding do this, there is the final iteration known as the Convolutional Compact Transformer, where they not only use sequence pooling method but also the patch embeddings are replaces by convolutional embedding, convolutions are introduced to create inductive bias which helps in effective tokenization and preserves local spatial relationships.

CCT achieves better accuracy than CVT and ViT-lite. Convolutions have been very successful in computer vision like in AlexNet and ResNet due to their invariance to spatial translations and have low relational inductive bias. They provide good inductive priors where each convolution cares about the next convolution and so on. This is very intuitive as this is how human beings perceive images, they observe one part of the image and then start expanding to the neighboring pixels. Therefore, we bias the tokenization part by using convolutional blocks which help the model to become better at that specific task.

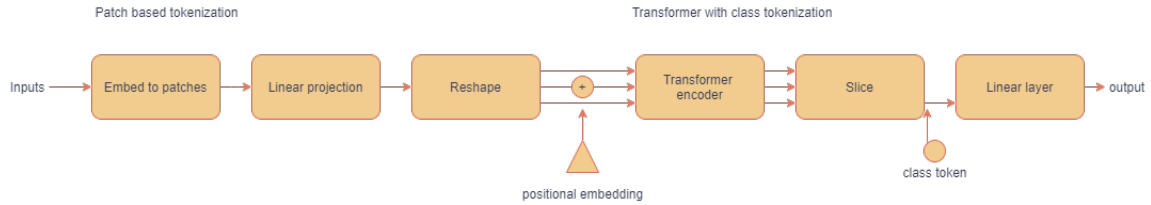


Figure 1: Here is the schematic diagram of the Vision Transformer which uses conventional class tokens and positional embedding.

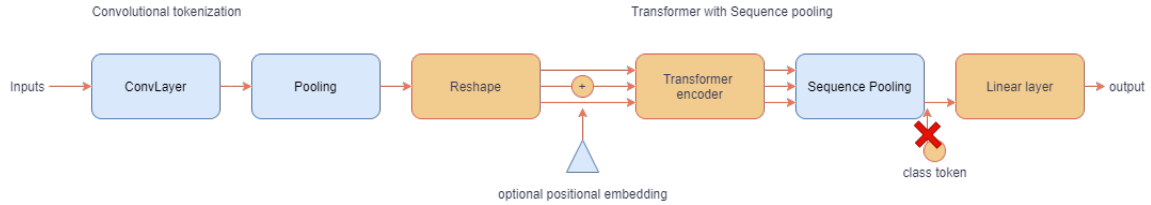


Figure 2: Here is the schematic of the Convolutional Compact Transformers. The blocks in blue are the changes made from the original transformer. We can observe how the architecture has been changed.

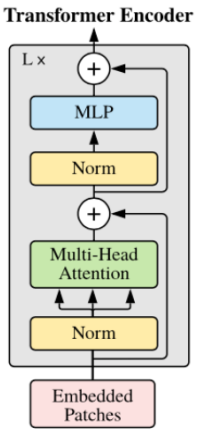


Figure 3: This is the architecture for a transformer encoder. Source: 2

# 2 Implementation of the Paper

The dataset I used to train the model on is CIFAR10 which has number of classes as 10 with shape of images as (32x32x3).
The images then undergoes data augmentation where it is first rescaled be 1/255 i.e inputs of [0, 255] is mapped to [0, 1] range. The images also undergo random cropping and random flipping.
We can start the implemenation from ViT-Lite and then gradually progress to CCT

**ViT-Lite is implemented as follows** -
1) We compute the patches with each patch having dimensions of patch-size which is rolled out into a sequence.
2) each patch is encoded by passing it through a linear embedding layer which is essentially gives out a vector and can be attached to a positional embedding.
3) This passes to the transformer encoder(Fig 3) which is finally passes to an MLP layer with GELU activation for classification.

**Convolutional Vision Transformer** -
1.This has the same architecture as ViT-Lite except for additional features.
2.The authors introduce a novel sequence pooling strategy that pools essential information that results from the transformer encoder and eliminates the need for an extra class token for classification.
3.This allows better correlation of outputs from the transformer encoder, the weights in the sequence pool are learnt during training and are better than static weights as each patch does not contain the same information.
4.This also allows the models to utilize information across sparse data as compared to the classical MLP head present only on the left most output of the transformer encoder.

**Convolutional Compact Transformer** -
1.In this variant, there is a change in the tokenization which introduces convolutional blocks to produce the tokens.
2.This helps in preserving the local spatial relationships between the patches and therefore doesn't depend on the positional embedding.

**Regularization methods used** -
1.Stochastic depth dropout - randomly skipping set of layers while training. This is similar to Dropout but this operates on block of layers.

2.Layer normalization - This normalizes the activation of the previous layer between 0 and 1.

3.Dropout for MLP heads

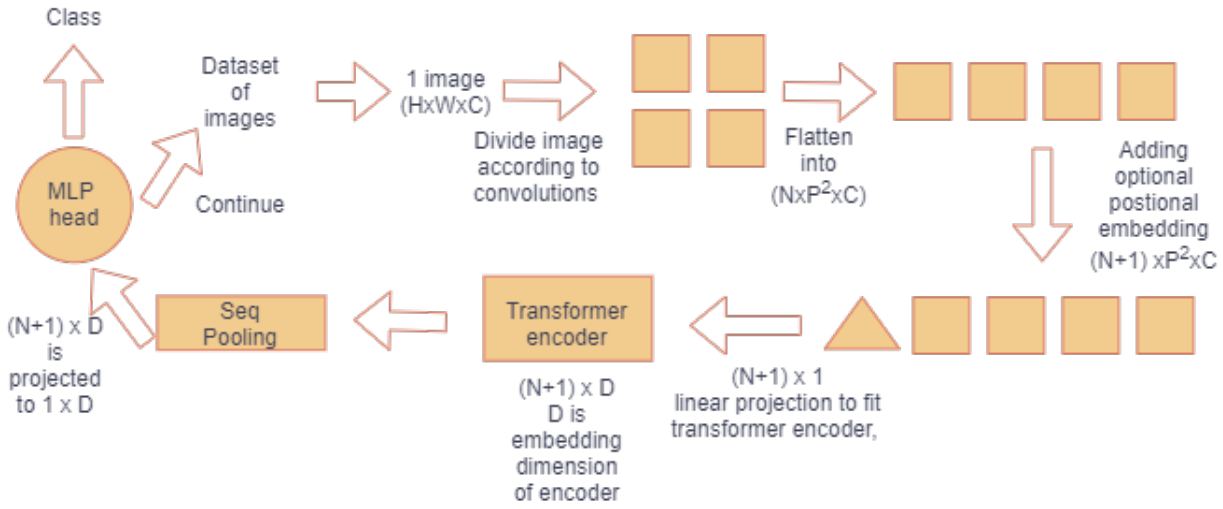


Figure 4: Here is the workflow of how the Convolutional Compact Transformer functions from end to end.

# 3 Results

In this section I have presented the results and hyperparameters used in the three transformer variants as described in the paper on CIFAR10 dataset.
Hyperparameters -
Optimizer = AdamW
learning rate = 0.001
weight decay = 0.0001
batch size = 256
num epochs = 30
image size = 32
patch size = 6
projection dim = 64
num heads = 4
transformer layers = 8
mlp head units = [2048, 1024]

The training of ViT-Lite took about 120 minutes on Google Colaboratory's TPU and got an accuracy of 59.18 and validation accuracy of 60. CVT took 100 minutes to train with an accuracy of 61 and validation accuracy of 60 and a test accuracy of 59.29 The training of CCT took 200 minutes for training on Google and got an accuracy of 84.24 and validation accuracy of 77.64 and a test accuracy of 77.15.

# 4 Experiments

While doing the literature review for vision transformers, I came across an interesting paper on a model that uses 'MLP-mixer' and I tried to incorporate this method instead of a vanilla

MLP layer. This is good because the of the skip connection which helps in creating a bias in the model and each type of feature(channel) goes through a forward pass which helps in making a correlation between the feature vectors.

I used different positional embedding like sinusoidal positional embedding but it did have much difference. In general CCT is less sensitive to positional embedding thanks to the convolutional embedding.

# 5 Discussion

Why does sequence pooling improve performance?
While this is not explicitly explained in the paper, I have a few conjectures. Sequence pooling is nothing but matrix multiplication of the softmax of attention weights of the encoded patches. And these weights are learnt on the fly this means that it may learn biases and assign higher weights to those patches. The sequence pool like an attention layer in transformers gravitate towards specific regions of the image, as do humans naturally gravitate towards desired features rather than auxiliary features.
This is beneficial for out approach as we do not want to generalize our approach and focus more on localized information. The compensation for training of smaller data set maybe in the quality of the image

# 6 Drawback

Convolutional Transformers may not perform well for other tasks and cannot be generalized. If we have a lot of data, a biased model will perform worse as the estimator will not be a perfect representation of the underlying function for the generalization of multiple tasks.

# 7 Future Scope

1) Further changes in the attention mechanism can be made to make the patches better represented such as using attention gradient rollout which ignores regions with low attention. This will help in increasing the bias in the model towards the task.

2) It will be interesting to use involutions instead of convolutions in the tokenization step and see how the performance has changed.

3)We can use different type of loss functions such as patch-wise contrastive loss, patch-wise mixing loss etc.which helps improving the stability of the model and reduces information loss

# 8 Citation

@article hassani2021escaping,
title = Escaping the Big Data Paradigm with Compact Transformers,
author = Ali Hassani and Steven Walton and Nikhil Shah and Abulikemu Abuduweili and Jiachen Li and Humphrey Shi,
year = 2021,
url = https://arxiv.org/abs/2104.05704,
eprint = 2104.05704,
archiveprefix = arXiv,
primaryclass = cs.CV

# 9 Annexure

Figure 5: Result of attention map of a Vision Transformer

# 10 References

1) Escaping the Big Data Paradigm with Compact Transformers - Hassani et al. (CVPR 2021)
2)An image is worth 16x16: Transformers for image recognition at scale. A. Dosovitsky et al.(ICLR 2020)
3) MLP-Mixer: An all-MLP architecture for Vision, Tolstikhin et al.(CVPR 2021)
4)Quantifying Attention Flow in Transformers(2020) Samira Abnar and Willem Zuidema
5)Vision Transformers with Patch Diversification(CVPR 2020), Gong et al.
6)https://keras.io/examples/vision/image_classification_with_vision_transformer/
7)https://keras.io/examples/vision/mlp_image_classification/