Funnel Vision Transformer for image classification

Yujing Zhang
Stanford University
yujingz@stanford.edu

Abstract

Vision Transformer(ViT) [6] adopts the Transformer architecture on the image classification tasks and outperforms the state-of-the-art convolutional networks with substantially fewer computational resources. However, it's still expensive to train Transformer either on a very large pre-training dataset or with a large model size. So model efficiency is still an important area to explore. Spatial compression is a common technique widely used in convolutional networks for image classification tasks, which indicates the spatial information redundancy for classification tasks. In addition, inspired by the success of Funnel-Transformer [4] in NLP, this project examines a similar idea on the ImageNet dataset that gradually shrink the image patch length dimension of Vision Transformer as the layers go deeper, in order to save the computational resources (Funnel-ViT). The results show that with with a small pre-training accuracy compromise ($< 1\%$), we can save 40% memory, get 37.5% speedup with three funnel blocks, and get 0.6% fine-tuning accuracy improvement. The saved resources can even be re-invested to a wider and deeper Funnel-ViT model to further reduce the pre-training accuracy loss to 0.1%.

1. Introduction

The unsupervised pre-training has been widely adopted in computer vision tasks. It has been observed that with a larger pre-training dataset or a larger model size, the model performance consistently improves. However, it would require much more computational resources during pre-training and even fine-tuning subsequently. It limits the model adoptions in real world. There has been lots of efforts to improve the model efficiency. For computer vision tasks, lots of work have been done (e.g. the pooling technique, the inception module introudced by GoogLeNet [11]) to improve the efficiency of convolutional networks given it's dominance in computer vision tasks. One common effective approach is to compress the spatial dimension for image classification tasks.

Inspired by the success of the Transformer-based model architecture in NLP tasks, Vision Transformer applies the Transformer architecture on the image classification tasks. It entirely replaces the convolutions with self-attentions, in order to benefit from the computational efficiency and scalability of Transformer. Vision Transformer outperforms the state-of-the-art convolutional networks with much fewer computational resources, especially on a very large pre-training dataset. This work arises the popularity of the Transformer architecture in computer vision.

Given the popularity of Transformer architecture and the spatial information redundancy, this project aims to remove the spatial redundancy in Vision Transformer to improve the training efficiency for image classification tasks. Inspired by the success of Funnel-Transformer in NLP pre-training, this project examines a similar approach to gradually shrink the patch length dimension of Vision Transformer hidden states (Funnel-ViT). The reduced patch length dimension could largely reduce the required memory and computation FLOPs.

This project conducts all experiments on the ImageNet dataset. The inputs are an image and a class label. They are feed into the Vision Transformer architecture and its variants to predict a classification label. The results show that with a very small sacrifice on the pre-training accuracy ($< 1\%$), we could save 40% memory, get 37.5% speedup and even better fine-tuning accuracy (0.6% improvement). It demonstrates the redundant information in the patch sequence dimension of Transformer layers. In addition, this project also explores different ways to re-invest the saved resources to model capacity in order to further improve the model quality. It turns out that a deeper and wider Funnel-ViT can be easily overfitting on the training data. After tuning the model depth and width, the overfitting issue is mitigated and Funnel-ViT can almost recover the pre-training accuracy of full-length ViT. The results also show that it's helpful to have shallow layers wider and deep layers thinner.

2. Related Work

Image classification is a popular classic computer vision task. There has been a significant progress made over the past few years. Most advanced model architectures are built on top of deep convolutional networks, like [7] [8] [10]. One common pattern is that, as the layers goes deeper, the network downsamples the feature representations in the spatial dimensions through either convolutional layers or pooling layers. It reduces the computation and memory cost of maintaining a full spatial size during CNN training and also indicates the information redundancy in the full-sized spatial representation for convolutional networks. The convolutional layers shares the model weights across different image positions which largely reduces the number of weights to train compared to fully connected layers. Moreover, the convolution operation is applied within a small image region, so it's good at capturing local structures. However, it's hard to capture dependencies with a long distance. Another drawback is that the convolution operations treat different receptive fields equally.

Self-attention [12] mechanism is dominate in NLP. It's good at capturing distant-dependencies compared to convolutional layers and takes token importance into consideration. Although we compute the attention scores of token pairs over the whole sequence, the scores over different positions can be computed in parallel. Given its strong power, there have been efforts which combine convolutional networks and self-attentions for computer vision tasks, like Visual Transformer [13]. It first uses a convolutional layer to extract low-level features. Then, it uses a tokenizer to group pixels into semantic tokens and feeds them into Transformer. There have also been efforts on replacing convolution layers with self-attentions completely. In order to make the model scalable with the high resolutions, some adjustments are made on top of self-attentions. For example, Image Transformer [9] computes attention scores only on neighbors of each query pixel. Sparse Transformer [2] introduces factorized self-attention to save computation. Visaul Transformer, Image Transformer and Sparse Transformer all outperform the state-of-the-art convolution-based models on ImageNet. However, either the convolutional layers or additional adjustments on self-attentions make users not fully benefit from self-attentions, e.g. the Transformer scaling law.

[3] completely replaces the convolutional layers and uses native Transformer layers on a sequence of image patches for image classification tasks. The model architecture is much simpler than Visaul Transformer, Image Transformer, Sparse Transformer, etc. It shows that attention layers can express any convolutional layers given enough attention heads. But the patch size with 2x2 pixels makes the model only applicable to images with small resolutions. Vision Transformer(ViT) [6] uses larger patches and demonstrates the strong power of Transformer in large-scale pre-training for computer vision tasks. With a large pre-training dataset, Vision Transformer beats the inductive bias inherit to CNNs and outperforms the state-of-the-art CNN architectures with much fewer resources. Compared to convolution-based ResNet, it provides higher inference speed and allows a large batch size for training. Furthermore, it can be easily scaled to a wider and deeper model and yields better model quality. But it's still expensive to pre-train and fine-tune a large Vision Transformer with more data. The paper also shows that "decreasing the patch size and thus increasing the effective sequence length shows surprisingly robust improvements". However, a longer sequence length means computationally more expensive. Thus, training efficiency is important for Vision Transformer to be widely adopted.

Funnel-Transformer [4] proposed an efficient Transformer architecture for NLP pre-training at a lower cost through gradually shrinking the hidden states in sequence dimension. It effectively removes the redundancy in the sequence dimension and largely reduces the computation complexity. It can further improve the model quality by re-investing the saved FLOPs and memory to model capacity. It outperforms the standard Transformer on varieties of NLP tasks, especially on sentence-level predication tasks. A similar idea can be extended to Vision Transformer for image classification tasks.

3. Methods

3.1. Baseline/ViT

Vision Transformer [6] applies the standard Transformer architecture directly to images by splitting the spatial pixels of an image (H, W) into a sequence of fixed-size 2D image patches with a smaller spatial dimension (P, P). The number of patches N would be HW/p^2. The self-attentions are applied on patches instead of pixels, which makes the attention computation more scalable with different image resolutions. The patches are flattened and fed into a trainable linear projection layer to get patch embeddings. It also adds learnable 1D position embeddings to the patch embeddings to retain location information of the patches. At the beginning of the sequence, ViT adds an extra learnable [class] token as classification head. The Transformer output of this token serves as a image-level representation for class prediction.

The Transformer encoder consists of a stack of Transformer layers with the same configurations. Within each Transformer layer, it has two blocks: self-attention block and MLP block, and applies residual connection for each block in order to make a deep neural network easier to train. The self-attention block consists of a layer normalization followed by a multi-head attention layer. The multi-head self-attention allows to learn different kinds of dependencies of different patch pairs over the patch dimension. The MLP block consists of a layer normalization followed by a feed-forward MLP layer.

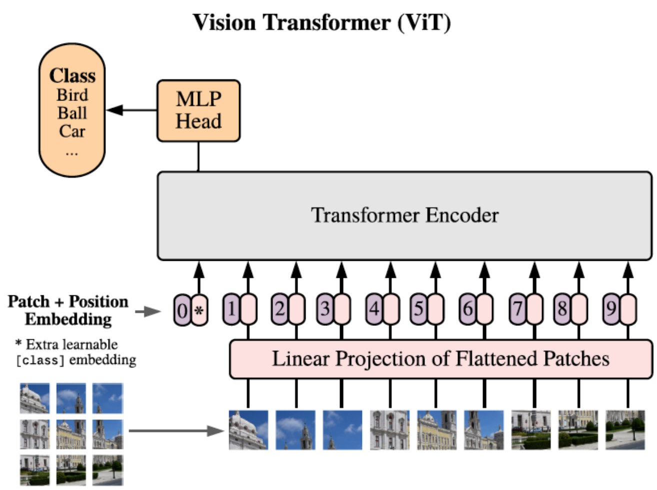


Figure 1. Vision Transformer (copied from [6])

$$Z_0 = [X_{class}; X_p^1 E; X_p^2 E; \cdots; X_p^N E] + E_{pos}$$

$$Z_l' = Self{-}Attention(LayerNorm(z_{l-1})) + z_{l-1}, l = 1 \cdots L$$

$$Z_l = MLP(LayerNorm(z_l)) + z_l', l = 1 \cdots L$$

$$y = LayerNorm(Z_L^0)$$

For the image classification tasks, it takes the outputs of the last layer on the [class] token as the encoder outputs and uses it together with labels to compute a categorical cross entropy loss.

During fine-tuning, the image resolution is usually higher than the resolution at the pre-training stage. ViT keeps the same patch size, resulting in a longer patch sequence. It makes the position weights not loadable directly. To deal with arbitrary patch length, ViT performs 2D interpolation of the pre-trained position embeddings. The new position embeddings are based on the locations in the image of a pre-training resolution.

3.2. Funnel-ViT

Funnel-Transformer [4] splits the stacked Transformer layers into several blocks. Different blocks have same layer configuration except for sequence length. Within each block, the sequence length of hidden states keeps the same. But the sequence length is cut by half across the block boundary by a pooling layer. The pooling is applied on patch tokens and ignores the first [class] token to preserve full image-level feature representation.

$$h' = concat(h[0], Pooling(h[1:]))$$

.

Pretraining	Finetuning
ViT	ViT
Funnel-ViT	Funnel-ViT
Funnel-ViT	ViT
ViT	Funnel-ViT

Table 1. Pretraining and Finetuning setup.

For the first transformer layer after pooling, it uses the "pool-query-only" technique [4], which takes the pooled hidden states h' as query and unpooled states h as key and value.

$$Attn(Q = h', KV = h)$$

In this way, more information can be carried to the compressed representation compared to naively taking pooled states as key and value.

$$Attn(Q = h', KV = h')$$

For the token-level prediction tasks, Funnel Transformer could recover a full-length representation by adding the last full-length representation from the first block and an up-sampled representation of the last layer.

Funnel-ViT combines ViT and Funnel-Transformer. It adopts the same approach as ViT to process an image as a sequence of patch embeddings and positional embeddings. Then, feed the embeddings to Funnel-Transformer architecture. Image classification is a sequence-level task, so we only need to a compressed representation (image-level) and don't need to recover a full-length representation (patch-level).

Compared to Funnel-Transformer, Funnel-ViT explores different re-investing strategies. The configurations of Transformer layers can vary across different blocks in Funnel-ViT. For example, different blocks can have different number of layers, hidden dimensions and MLP dimensions. Funnel-ViT uses a projection layer to re-size the hidden states.

In addition, ViT and Funnel-ViT have the same weight shapes. So Funnel-ViT can load pre-trained ViT weights, and ViT can load pre-trained Funnel-ViT weights. This project also examines the effectiveness of Funnel-ViT in both pre-training and fine-tuning with setups as shown in Table 1.

I use the ViT implementation provided by TensorFlow official models [1]. On top of it, I implemented the Funnel Transformer architecture and 2D interpolation of the pre-trained position weights for fine-tuning.

[1] available at https://github.com/tensorflow/models/tree/master/official/projects/vit

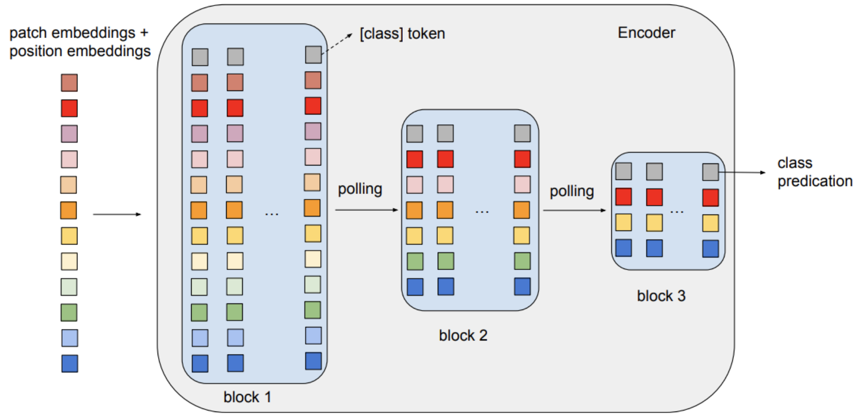


Figure 2. Funnel ViT

4. Dataset and Features

In this project, I use the standard ImageNet [5] dataset for both pre-training and fine-tuneing. It has 1000 classes and contains 1,281,167 training images, 50,000 validation images and 100,000 test images. The input resolution is 224 for pre-training and is 384 for fine-tuning. I convert the dataset to tfrecord in order to use the ViT implementation provided by TensorFlow official models.

5. Experiments and Analysis

5.1. Experiments

Model configs

- patch size = 16, patch length = 14 x 14 + 1 = 197 (yield better accuracy compared to patch size = 32)
- Transformer with 12 layers, 12 attention heads, hidden dimension=768, MLP dimension=3076 (standard base-size Transformer)
- Pooling with stride = 2, window = 2 (suggested by Funnel Transformer paper)
- Adam optimizer with learning rate = 0.003, weight decay = 0.3 for pre-training ; SGD optimizer with a momentum of 0.9 for fine-tuning (same as the Vision Transformer paper).

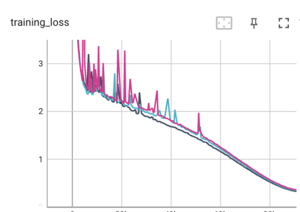
Evaluation metrics

- model quality: top1 accuracy and top5 accuracy
- model resource usage: memory usage, steps/sec

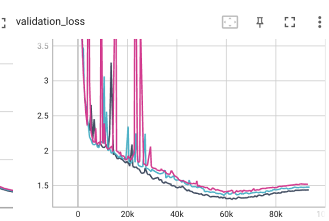
5.2. Results and Discussion

5.2.1 Compression

Pre-training Table 2 shows the results of different block layouts for pre-training on ImageNet. 'Bn(t)' means that there are n Transformer layers in a block with patch length t. The accuracy of Funnel-ViT only reduces slightly after cutting the hidden states by half in the sequence dimension. It demonstrates the assumption that there is spatial information redundancy in the deeper Transformer layers of ViT. It takes more than 12 hours to pre-train the ViT-base model on ImageNet. However, with a small accuracy compromise, we can save 25.8% memory and get 23% speedup with two funnel blocks (0.22% accuracy loss), and save 40% memory and get 37.5% speedup with three funnel blocks (0.7% accuracy loss). My experiments also show that mean pooling performs better than max pooling.

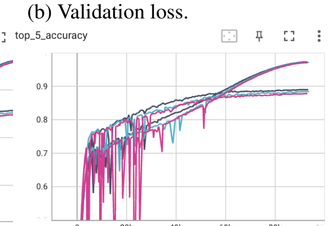


(a) Training loss.



(b) Validation loss.

(c) Top1 accuracy.



(d) Top5 accuracy.

Figure 4. Loss and accuracy pretraining curves of ViT and Funnel-Vit
(black: ViT; blue: Funnel-ViT B6(197)-6(99); red: Funnel-ViT B4(197)-4(99)-4(50))

block layout	train top1	val top1	train top5	val top5	memory usage	steps/sec time
B12(197) (ViT)	92.4%	71.41%	97.28%	89%	12.38G	2.0
B6(197)-6(99), max	92.05%	70.42%	97.15%	88.41%	9.18G	2.6
B6(197)-6(99), mean	92.12%	71.19%	97.17%	88.9%	9.18G	2.6
B4(197)-4(99)-4(50), max	91.66%	69.71%	97.01%	87.8%	7.38G	3.2
B6(197)-6(50), max	92.02%	70.42%	97.14%	88.26%	8.04G	3.1

Table 2. Different block layouts of pre-training on ImageNet.

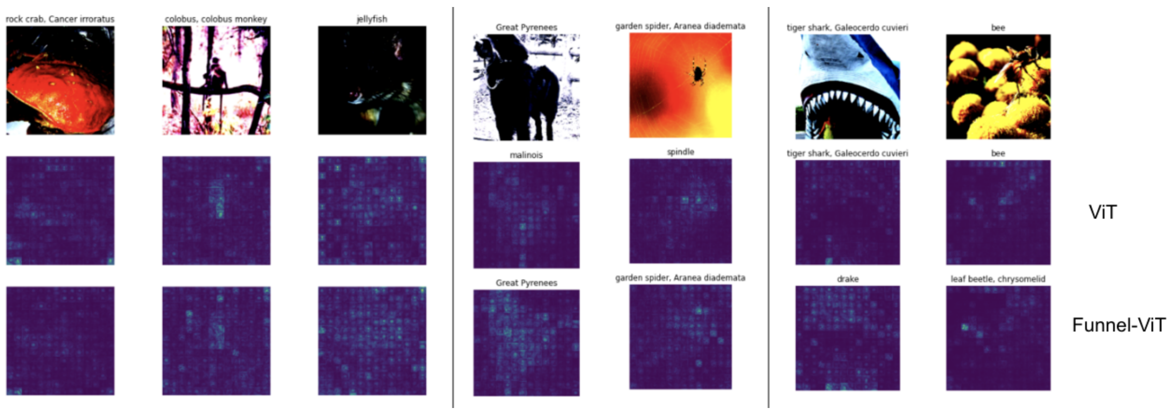


Figure 3. Saliency maps

From the training curves in Figure 4, the model converges at the same number of steps. So the speedup mainly comes from more steps per second. Moreover, if we compress the spatial dimensions too much each time (e.g. stride size = 4), it would make the training more unstable. Deeply compressed features are more sensible to gradients update at the initial training stage.

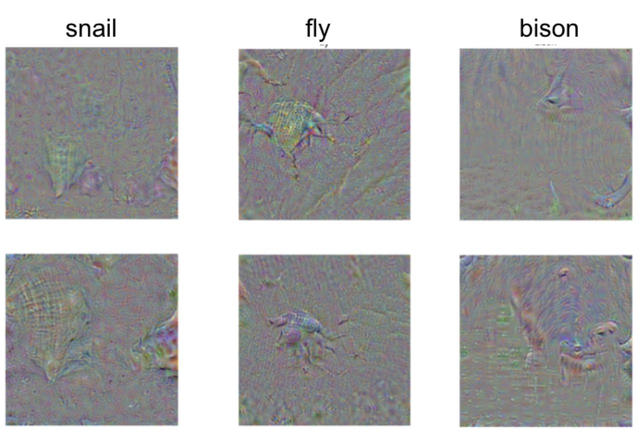


Figure 5. Class visualization

For a given class, the class visualization generated by Funnel-ViT is similar to ViT. Funnel-ViT is able to capture similar class features as ViT. For example, for class snail, both of the models are able to learn the spiral shells of snails. For class fly, both of the models capture the feature of long legs.

The saliency maps of the same image with ViT and Funnel-ViT are almost the same as shown in Figure 3. The regions of pixels that have a large effect on the classification score are similar. It re-demonstrated the spatial information redundancy in Transformer layers. My experiments show that Funnel-ViT misclassifies some images which are classified correctly by ViT, while correcting a few classifications. This can be explained by the saliency maps. The pooling layer of Funnel-ViT slightly enlarges regions with a large effect on the classification score. The noise introduced by the additional pixels would disturb the feature extraction, thus affecting class prediction.

Fine-tuning Form Table 3, we can get a better accuracy with a pre-trained Funnel-ViT when fine-tuning on ImageNet with a higher resolution, no matter whether the fine-tuning task uses ViT or Funnel-ViT. We can get 0.6% improvement on top-1 accuracy with a pre-trained Funnel-ViT, compared to a pre-trained ViT. We can also save 26.5% memory and get 24% speedup. Furthermore, The results shows that fine-tuning ViT with a pre-trained Funnel-ViT yields better accuracy than the baseline (ViT for both pre-training and fine-tuning). However, fine-tuning Funnel-ViT with a pre-trained ViT yields worse accuracy than the baseline. It makes sense since Funnel-ViT learns to produce

pre-training	fine-tuning	train top1	val top1	train top5	val top5	memory usage	steps/sec time
ViT	ViT	73.84%	70.95%	90.06%	89.45%	7.32G	4.4
Funnel-ViT	**Funnel-ViT**	75.28%	71.53%	90.8%	89.97%	5.38G	5.8
Funnel-ViT	ViT	75.18%	71.35%	90.74%	89.8%	7.32G	4.4
ViT	Funnel-ViT	73.09%	70.44%	89.7%	89.14%	5.38G	5.8

Table 3. Fine-tuning performance on ImageNet.

block layout	train top1	val top1	train top5	val top5	memory usage	steps/sec time
B12(197) (ViT)	92.4%	71.41%	97.28%	89%	12.38G	2.0
B6(197)-6(99)	92.12%	71.19%	97.17%	88.9%	9.18G	2.6
B6(197)-6(99)-4(50)	93.7%	70.88%	97.76%	88.32%	10.19G	2.3
B6(197, h1024)-6(99, h1024)	93.45%	70.7%	97.69%	88.42%	10.54G	1.76
B6(197, m4096)-6(99, m4096)	93.29%	70.27%	97.62%	88.11%	10.04G	2.35
B6(197)-6(99)-4(50, m1536)	93.04%	71.17%	97.53%	88.79%	9.78G	2.4
B6(197)-6(99)-2(50, m1536)	92.65%	71.13%	97.36%	88.69%	9.44G	2.5
B6(197, m4096)-6(99)-2(50, m1536)	93.05%	71.31%	97.51%	88.69%	10.97G	2.1

Table 4. Different re-investment configurations of pre-training on ImageNet.
(h: hidden dim; m: MLP dim)

a image-level representation and ViT learns to produce a patch-level representation. When the resolution changes during fine-tuning, a patch-level representation is relatively not so meaningful.

5.2.2 Re-investment

Tables 4 shows the results of different ways to re-invest saved resources. In general, either a deeper or a wider model could improve the accuracy on the training dataset. But it reduces the accuracy on the validation dataset, resulting in overfitting. The overfitting issue can be mitigated by tuning the model width and depth at the same time. Empirically, while adding more layers, increase the model width of shallow layers and reduce the model width of deep layers. However, tuning the model width and depth cannot fully solve the overfitting issue. Funnel-ViT with a larger model size doesn't outperform ViT. The reason is that a larger model usually requires more training data. From the ViT paper, Large-ViT (24 layers) yields a worse accuracy on ImageNet but a higher accuracy on ImageNet-21k. ImageNet-21k contains 21k classes and 14M images which is much larger than ImageNet. So it's expected to see overfitting on ImageNet when increasing the model capacity.

While tuning the model width, it's better to adjust the MLP dimension instead of the hidden dim given limited pre-training data. A larger hidden dim would make the model easier to overfit on the training data and largely slows down the training speed at the same time.

6. Conclusion and Future work

In this project, I explored the Funnel-Transformer architecture introduced for NLP tasks on top of Vision Transformer (base size) for image classification tasks with ImageNet dataset. The results show that with a small pre-training accuracy loss, we can save 25.8% memory and get 23% speedup with two funnel blocks (0.22% accuracy loss), and save 40% memory and get 37.5% speedup with three funnel blocks (0.7% accuracy loss). It largely reduces the Transformer-based pre-training time (more than 12 hours with ViT), making the Transformer architecture more applicable for image classification tasks. The accuracy and saliency maps both demonstrates the redundant information in deeper full-length Transformer layers. And the pooling layer would slightly enlarge the image regions which affects the prediction scores. Although, Funnel-ViT yields a worse pre-training accuracy, but it helps learn a better image-level representation during fine-tuning, producing better fine-tuning accuracy. With limited time, I only run fine-tuning experiments on the ImageNet dataset. It's worth experimenting on other datasets (e.g. CIFAR-10, CIFAR-100) to consolidate the conclusion.

The project also explored different ways to re-invest the saved resources to a deeper and wider model in order to improve the accuracy. Due to the limited computation resource, I have only experimented on the ImageNet dataset, which is small for large-Transformer. So a deeper and wider model would overfit on the training data. But Funnel-ViT can mitigate the overfitting issue and almost recover the pre-training accuracy of ViT by tuning the model depth and width. In general, it's helpful to increase the width of shallow layers and reduce the width of deep layers. It would be interesting to explore the re-investment further on a larger training dataset like ImageNet-21k, which allows us to experiment with larger models.

7. Contributions

The model is implemented in TensorFlow [1]. I use the ViT implementation provided by TensorFlow official models [1] as the baseline.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Z. Chen, Craig Citro, Gregory S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian J. Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Józefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Gordon Murray, Christopher Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul A. Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda B. Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *ArXiv*, abs/1603.04467, 2016. 7

[2] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *ArXiv*, abs/1904.10509, 2019. 2

[3] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. *ArXiv*, abs/1911.03584, 2020. 2

[4] Zihang Dai, Guokun Lai, Yiming Yang, and Quoc V. Le. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. *ArXiv*, abs/2006.03236, 2020. 1, 2, 3

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 4

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021. 1, 2, 3

[7] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 2

[8] Alex Krizhevsky. Convolutional deep belief networks on cifar-10. 2010. 2

[9] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam M. Shazeer, Alexander Ku, and Dustin Tran. Image transformer. *ArXiv*, abs/1802.05751, 2018. 2

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015. 2

[11] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. 1

[12] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017. 2

[13] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Péter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *ArXiv*, abs/2006.03677, 2020. 2

[1] available at https://github.com/tensorflow/models/tree/master/official/projects/vit