REPORT OF COURSEWORK FOR COMP6248 AN IMAGE IS WORTH 16X16 WORDS

Xin Chen, Xiaoqiu Zhou, Huaiyi Fang
Electronics and Computer Science
University of Southampton
{xc5n21,xz2y21,hf5n21}@soton.ac.uk

ABSTRACT

The emergence of the Vision transformer attempts to demonstrate that the transformer, a model from NLP, can also achieve state of the art results in the field of computer vision. To verify the reproducibility of vision transformer, we reproduce some of the experiments in Beyer et al. (2021). The experimental results are similar to those of the authors. In addition, for the inductive bias we have also give some analysis based on experimental reproduction.

1 INTRODUCTION

Transformer, one of the most famous self-attention models, is a typical model of natural language processing, but its application in computer vision is more of an auxiliary CNN and other structures. In Beyer et al. (2021), the Google team attempts to demonstrate that computer vision's reliance on CNNs is unnecessary, and their new model, vision transformer (ViT), can achieve comparable results to CNNs. It solved a problem of inputting images into Transformer by dividing images into patches and making a sequence of linear embedding as the input of the Transformer encoder.

In this report, we will try to reproduce the ViT model. Since ViT requires a huge amount of pre-training data, we will directly import the parameters of the pre-trained model for testing instead of completing the pre-training ourselves. We will compare the pre-trained ViT with ResNet on some small datasets, and visualise the position embedding and attention map, to verify whether ViT performs as expected and the authenticity of the experiments done by other authors.

We have attached our build and results to our Github repository[1].

2 REPRODUCTION OF VISION TRANSFORMER

The invention of Vision Transformer was inspired by Transformer in the field of NLP. As a new attempt in the field of computer vision, the paper mentions several times that the structure of the model mirrors the original Transformer in Vaswani et al. (2017) as closely as possible. Therefore, our reproduction follows this principle rather than making equivalent changes. The implementation of this section can be viewed in *models.py* of our repository.

First, the difficulty that tries to be solved in the paper is the input of flattened images is too long compared to sentences in NLP. So, the core idea of ViT is to break up the picture as patches thus each patch can be small enough to input into the Transformer. Another problem is the lack of inductive bias of the Transformer, so the positional embedding is required. The paper also imports the class token, which was actually drawn on the Bert of Devlin et al. (2018). We first implemented linear projection of flattened patches and positional embedding in the class *LPFP()* and *PatchAndPosEmb()*.

Second, the Transformer encoder is nearly identical to the original Transformer. It can be clearly observed that there is a slight difference between the encoder in Figures 1 and Figure 2 and that

[1] Github repository site: https://github.com/COMP6248-Reproducability-Challenge/Vision-Transformer-COMP6248CW

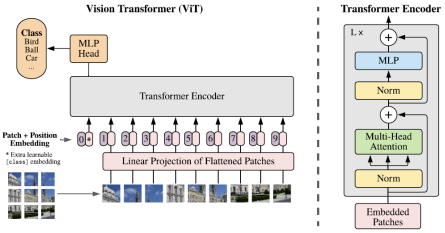

Figure 1: The structure of Vision Transformer

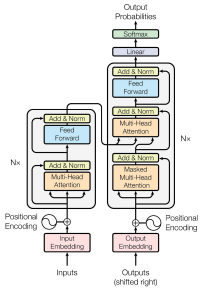

Figure 2: The structure of original Transformer

is the order of the layer norm. Other than that, the structure is exactly the same as that of the original Transformer. We implemented transformer encoder in the class *TransformerEncoder()* and *TransformerEncoderLayer()*.

Finally, there is an MLP classifier head after the Transformer encoder. Only the output of the class token will be input into the MLP head. At this point, the class token has been affected by all patches, so it is similar to a pooling operation that can give a global view. Actually, subsequent research and others' implementations usually choose to use the pooling layer to do the equivalent operation, but based on the principle of following the paper exactly, we still use the class token here. Our implementation can be found in the class *MLP()*.

The whole structure of Vision Transformer was completed in the class *ViT()*. We also implemented the download of pre-trained models from *torchvision* and automatically transfer it to our model.

3 Experiments and further analysis of Vision Transformer

Training ViT from scratch can be not possible for us due to the limitation of computing resources. So, we decide to do some in-depth experiments using transfer learning. The package *torchvision* provided some pre-trained models of ViT. **We implemented automatic download and transfer of pre-trained models from *torchvision*, so our subsequent experiments will be based on our build of ViT with transferred pre-trained parameters of *torchvision*.**

3.1 Transfer and fine-tune pre-trained ViT to new datasets

The package *torchvision* provided pre-trained models[2] of ViT-B/16, ViT-B/32, ViT-L/16 and ViT-L/32. Vision Transformer has a huge number of parameters and has strong requirements for computing equipment. Due to the limited computing resource, we decided to take the following measures to save graphic memory usage and training time:

- The pre-trained ViT-B/16 model was chosen for experiments because it has a relatively fewer amount of parameters among ViT models. For comparison, we chose ResNet-152x1 which was published in the research of He et al. (2015) instead of ResNet-152x2 or ResNet-152x4 in Kolesnikov et al. (2020). Because ResNet-152x1 is most comparable to ViT/B-16 in terms of number of parameters, ResNet-152x2 or ResNet-152x4 are too large models.
- The number of parameters for fine-tuning should continue to be reduced since 86M parameters of ViT-B/16 are still tricky. So, when fine-tuning the ViT, only the multi-head attention layers and MLP head were unfrozen. The research of Touvron et al. (2022) showed that only fine-tuning the multi-head attention layers can save a lot of computational cost without appreciably degrading the accuracy.
- The batch size was reduced to 64, while Beyer et al. (2021) used 512.

[2]Models provided by *torchvision* were pre-trained on ImageNet ILSVRC-2012, so the fine-tuning results may not be as good as those state of the art results of models pre-trained on huge datasets such as the ImageNet-21k or the JFT-300M. But it is still worth comparing since the experiments of the paper also give partial results that are pre-trained on ImageNet ILSVRC-2012 and transferred to other datasets.

Table 1 shows the comparison of ViT-B/16 and ResNet-152 after fine-tuning on various datasets. All fine-tuning of ViT was applied with a cosine decay learning rate schedule with linear warm-up. The training time for fine-tuning on each dataset using GTX1080Ti ranged from 0.5 to 2 hours.

	ViT-B/16	ResNet-152x1
CIFAR-10	97.97	96.67
CIFAR-100	87.41	82.67
Oxford Flowers-102	89.12	89.02
Oxford-IIIT Pets	93.65	93.24

Table 1: Accuracy (in %) of ViT-B/16 and ResNet-152x1 after fine-tuning on different datasets

It can be found that the accuracy of the two models is relatively close to each other in different datasets. The accuracy of ResNet seems to be a little bit lower at a similar training scale. Additionally, these experimental results are also close to the results claimed by their authors which proves that the research of this paper are reproducible.

3.2 RGB FILTER AND POSITION EMBEDDING

The patch embedding layer of ViT makes a linear projection of flattened patches. The authors visualised the filters of the initial linear embedding of RGB values of ViT-L/32 because of the well-trained networks often show nice and smooth filters in the field of computer vision. The Figure 3 below shows our visualisation of the first 28 principal components of the learned embedding filters. These filters appear to form a set of basis functions that represent the texture structure of the patch.

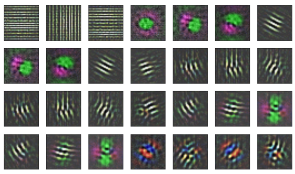

Figure 3: RGB embedding filters (first 28 principal components)

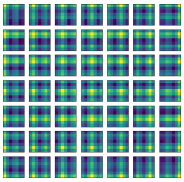

Figure 4: The visualisation of the position embedding

In addition, after the low-dimensional linear projection, a trainable position embedding is added to the patch representations. The authors also visualised the learned positional encoding in this paper and found that similar blocks of images had more similar positional encoding and similar peer or column positional encoding. We also verified this also by visualising. We obtained the Figure 4 by taking the position embedding of each patch and finding the cosine similarity with the embedding at the other positions.

In Figure 4, the colour of each block represents the similarity between the current patch and the corresponding patch in the row and column indexes, the closer the colour is to green, the lower the similarity, and the closer it is to yellow, the higher the similarity. For the patch in the first column of the first row, we zoom in and see that the top leftmost dot is yellow, representing the highest similarity to the patch in the first column of the first row (since the first column of the first row is its own position, the similarity is the highest). Secondly, the similarity between the first row and the first column is also higher, indicating that it is more similar to the patch in the current row and the current column. The other positions are less similar. And so on, giving a similarity profile for 7*7 patches.

3.3 ATTENTION VISUALISATION

In the paper, the author extracts the attention weight of each layer and uses these weights for attention rollout from Abnar & Zuidema (2020). The result of attention rollout is the focus of the model on the original image, which can be visualised by overlaying it on the original image as a mask.

The attention layer in the source code does not return weight by default. Two methods have been tried to obtain the attention weight: 1. Set the global variable list, get the attention weight and put it in the list, and call the list directly when using it; 2. Return the processing results of the attention weight and the input image to the model layer by layer output. The ViT model developed by ourselves is used, as well as the ViT in timm and torchvision, to conduct experiments (modify the source files in the library), and the results of the two methods are the same.

According to the method of attention rollout, the extracted attention matrix is synthesised into a tensor and the identity matrix is added for residual connection, and the result is normalised to generate raw attention. Since the information of each layer is propagated by the paths between nodes, the final information is the multiplication of all the weights of the paths between nodes. The output is added as a mask to the original image to get attention visualisation and the result is shown in Fig.5.

(a) hen (b) daisy (c) sea snake

Figure 5: attention visualisation

4 Conclusions

We reproduced the ViT model in Beyer et al. (2021), imported the pre-trained parameters, and used this model to test some important experimental content in the paper, such as the fine-tuning comparison between ViT and ResNet, the visualisation of RGB filter and position embedding, attention visualisation. In the original paper of Vision Transformer, the explanation of the model structure as well as the experimental procedures and results are clear, and the results of experiments are roughly the same as the results we reproduced.

References

Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020.

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. pp. 21, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. URL `http://arxiv.org/abs/1512.03385`. arXiv: 1512.03385.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big Transfer (BiT): General Visual Representation Learning. *arXiv:1912.11370 [cs]*, May 2020. URL `http://arxiv.org/abs/1912.11370`. arXiv: 1912.11370.

Hugo Touvron, Matthieu Cord, Alaaeldin El-Nouby, Jakob Verbeek, and Herve Jegou. Three things everyone should know about Vision Transformers. pp. 22, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. pp. 11, 2017.