Vision Transformer final-report

Taehun Kim[1]
[1] Department of Computer Science and Engineering, Pusan National University.

1 Introduction

The Vision Transformer[2] is a model that directly applies the Transformer[3] architecture to image. this model achieves excellent results when pre-trained at large scale and transferred to tasks with fewer datapoints.

In this report, We test the Vision Transformer, visualize its attention matrix. We use pre-trained Vision Transformer from TIMM(pytorch image model)[4] with 16×16 patch size and 224×224 input image size. The Vision Transformer implementation is from [5].

2 Test data

We use the Santorini image from [1]. the image is resized to 224×224, and normalized with mean of 0.5 and a standard deviation of 0.5.

3 Vision Transformer

In this section, we examine the details of the Vision Transformer by implementing a small part of the model. The architecure of the Vision Transformer is shown in Figure 2 and Figure 3.

3.1 Patch Embedding

We use 224×224 image, and divide it into patches of 16×16 size. This creates a grid size of 14×14, resulting in 196 patches. We use convolutional layers to embed each patch into a 1×768.

this convolutional layer has 3 input channels, 768 output channels, 16×16 kernel size, and 16×16 stride. After each patch passes through the convolutional layer, the embedded vectors are flattened from $14 \times 14 \times 768$ to 196×768.

3.2 Position Embedding

In this report, We use pre-trained position embedding. To visualize the position embedding, We calculate the cosine similarity between one patch and all other patches and reshape the similarity vector back to the original patch grid shape. The visualization is shown in Figure 5.

In the visualization, brighter areas in each patch indicate a higher similarity between the position embeddings corresponding to the rows and columns of that patch. These embeddings are more similar when aligned horizontally than vertically.

3.3 Transformer Input

We generate inputs for transformer using pre-trained patch and position embedding. After concatenating class token to the patch embedding, The position embedding is added to patch embedding, which finally produce input for transformer. The class token is already concatenated to position embedding. The input shape is 197×768.

The formula is

$$\mathbf{z}_0 = [\mathbf{x}_{class}; \mathbf{x}_1^p\mathbf{E}; \mathbf{x}_2^p\mathbf{E}; \cdots; \mathbf{x}_N^p\mathbf{E}] + \mathbf{E}_{pos}, \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}, \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D}$$

, which P is patch size, C is input channel, D is output channel(embedding dimension), and N is number of patches. In this lab, P is 16, C is 3, D is 768, N is 196.

3.4 Transformer Encoder

The transformer input is fed to L(=12) series Transformer Encoders for multihead self-attention. The embedded vector input is expanded to $197 \times (3 \times 768)$ by a fully connected layer. this vector is divided into 3 197×768 vectors: **q**,**k**, and **v**.

After that, Each vectors is divided into 12 197×64 vectors, one for each attention head. The output from the attention heads is concatenated and reshaped to match the encoder's input shape. After applying the attention heads, the result passes through layer normalization and two fully connected layers to produce the encoder's output. We use pre-trained transformer encoder in this lab.

The formula for attention head is

$$A = \mathbf{softmax}(\mathbf{q}\mathbf{k}^T / \sqrt{D_h})$$

$$SA(\mathbf{z}) = A\mathbf{v}$$

$$MSA(\mathbf{z}) = [SA_1(z); SA_2(z); \cdots; SA_k(z)]U_{msa}, U_{msa} \in \mathbb{R}^{k \cdot D_h \times D}$$

, which A is the attention matrix, k is number of self-attention operation, D_h is dimension of each head(typically set to D/k), and U_{msa} is projection matrix. In this lab, k is 12, D_h is 64.

The shape of **q** is 197×64, and $\mathbf{k}^T$ is 64×197. Therefore, the shape of attention score is 197×197. The softmax layer makes the sum of each row to 1.

3.4.1 Visualizing attention matrix

In this section, We visualize attention matrix of the Santorini image. Typically, Layer normalization is applied after self-attention(i.e., after multiplication of Attention matrix and **v**). However, We applied layer normalization to a Attention matrix for visualizing. Therefore, The attention matrix for visualizing is:

$$A_{visual} = \mathbf{layernorm}(\mathbf{softmax}(\mathbf{q}\mathbf{k}^T / \sqrt{D_h}))$$

The visualization of the attention matrix of 3rd head is shown in Figure 6. Additionally, the visualization of the 100th row of attention matrix of 0-7th heads is shown in Figure 8. Each 196 columns(except class token) is reshaped to 14×14. In the attention matrix visualization, the bright regions indicate that the vision transformer pays more attention to those parts of the image.

3.5 MLP for classification

Output of the transformer encoder is fed to MLP head, which is fully connected layer that maps 768 features to 1000 classifications. MLP head layer is also pre-trained. We only check the first rows of the output of the MLP head, corresponding to the class token index. The classification result is shown in Figure 7.

4 conclusion

We implemented the part of Vision Transformer[2], and examine and tested the pre-trained Vision Transformer model. We visualize the position embedding and attention matrix. Additionally, we comfirmed that the pre-trained ViT could classify the Santorini image correctly. As a result, We verified that the Transformer[3] can be directly applied to a image, attend to crucial parts of the image, and classify the image correctly.

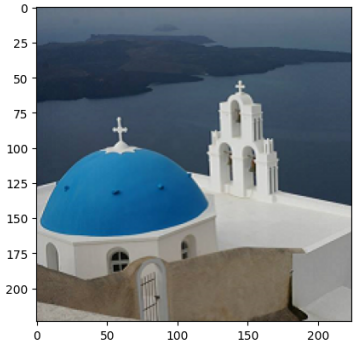

Figure 1: Santorini image for testing ViT. The pre-trained ViT model classifies this image as 'church' and 'church building.'

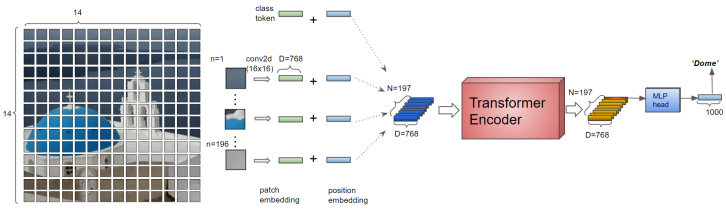

Figure 2: The architecture of Vision Transformer. This model divides the image into $16x \times 16$ patches(196 patches from 224×224 image), embedding each patch to 1×768 by $16 \times 16 \times 768$ convolutional layers. After that, this embedded patches are added to each position embedding. This added embedded vector is fed into the Transformer Encoder. The output of the encoder is passed to an MLP head, which finally produces the classification result.

[1] Github - hirotomusiker/schwert_colab_data_storage: schwert's data storage repo (mainly images) to load from colab — github.com. https://github.com/hirotomusiker/schwert_colab_data_storage/tree/master. [Accessed 21-11-2024].

[2] 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, 2021. URL https://arxiv.org/abs/2010.11929.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

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

[5] Ross Wightman. Vision transformer from pytorch image models. https://github.com/huggingface/pytorch-image-models/blob/main/timm/models/vision_transformer.py, 2019.

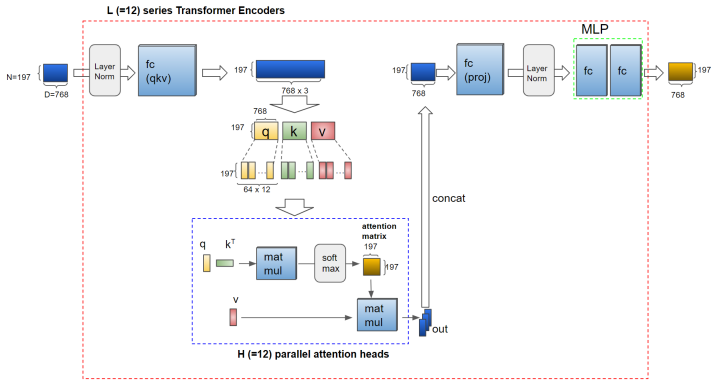

Figure 3: The architecture of Transformer Encoder. This encoder takes 197×768 embedded vector as input. After applying the layer normalization to the input, this input is fed to a fully connected layer, which produce $197 \times (3 \times 768)$ vector. this vector is divided into 3 197×768 vectors, which are the q,k,v vectors. For multi-head attention, these vector divided into 12 197×64 each, and each divided vector is changed to attention matrix through matrix multiplication and softmax. The output from the attention heads is concatenated and reshaped to match the same shape as the encoder's input. After applying the attention heads, the result passes through layer normalization and two fully connected (FC) layers to produce the encoder's output.

Figure 4: Visualization of patches.

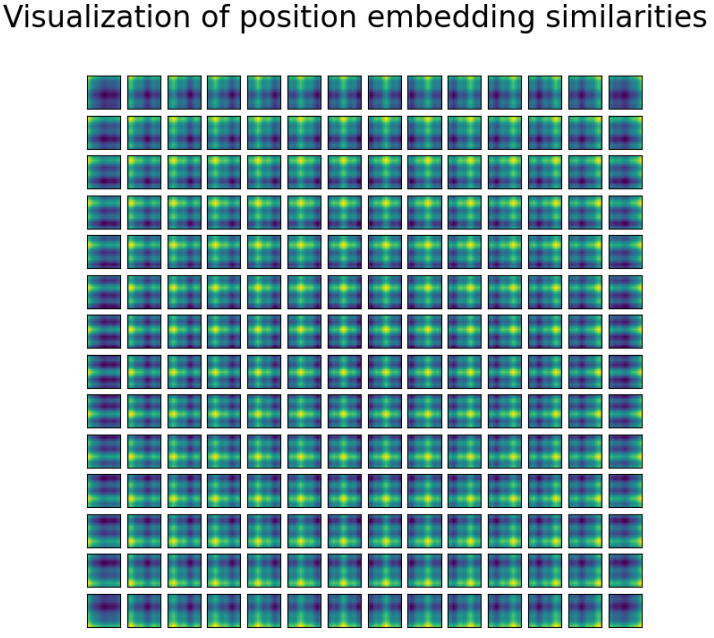

Figure 5: Visualization of position embedding similarities.

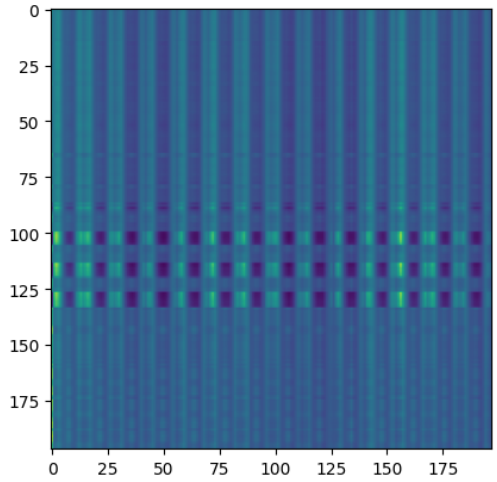

Figure 6: Visualization of attention matrix of 3rd head. The input image is the Santorini image. the size of attention matrix is 197×197

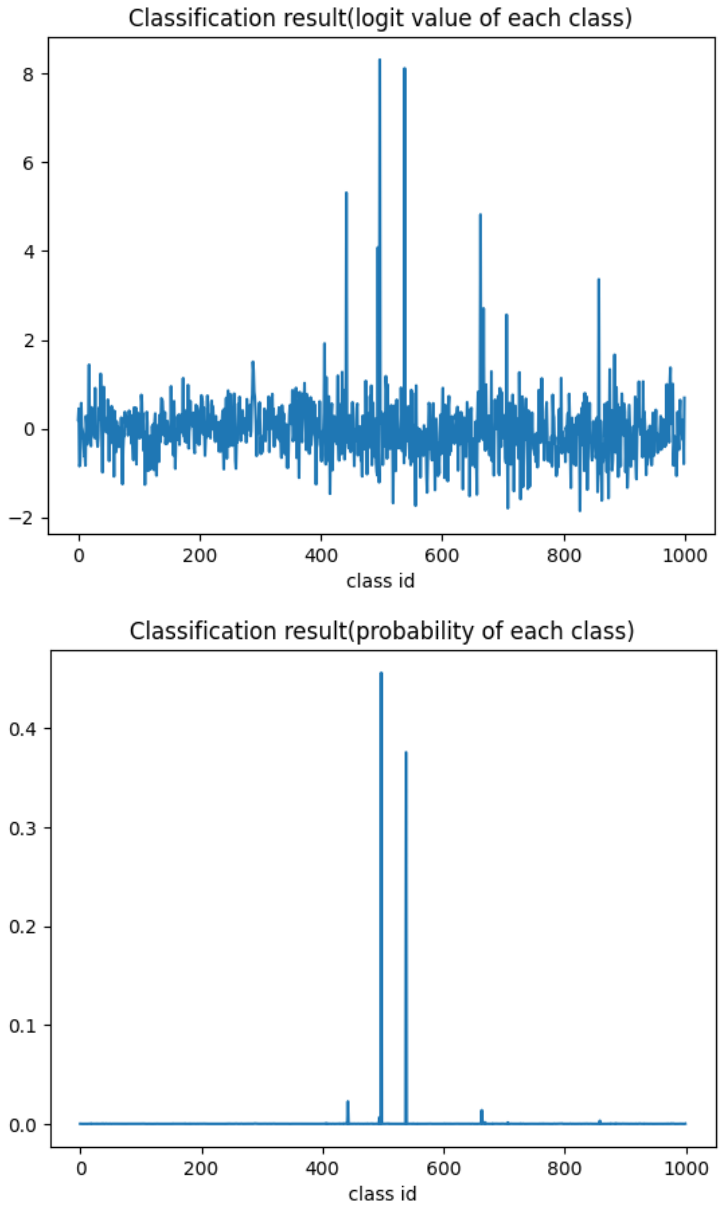

Figure 7: The classification result of the Santorini image using Vision Transformer. top is the logit values by class, and bottom is the probability by class. The highest class index is 497, which is corresponded to Church(and Church-building)

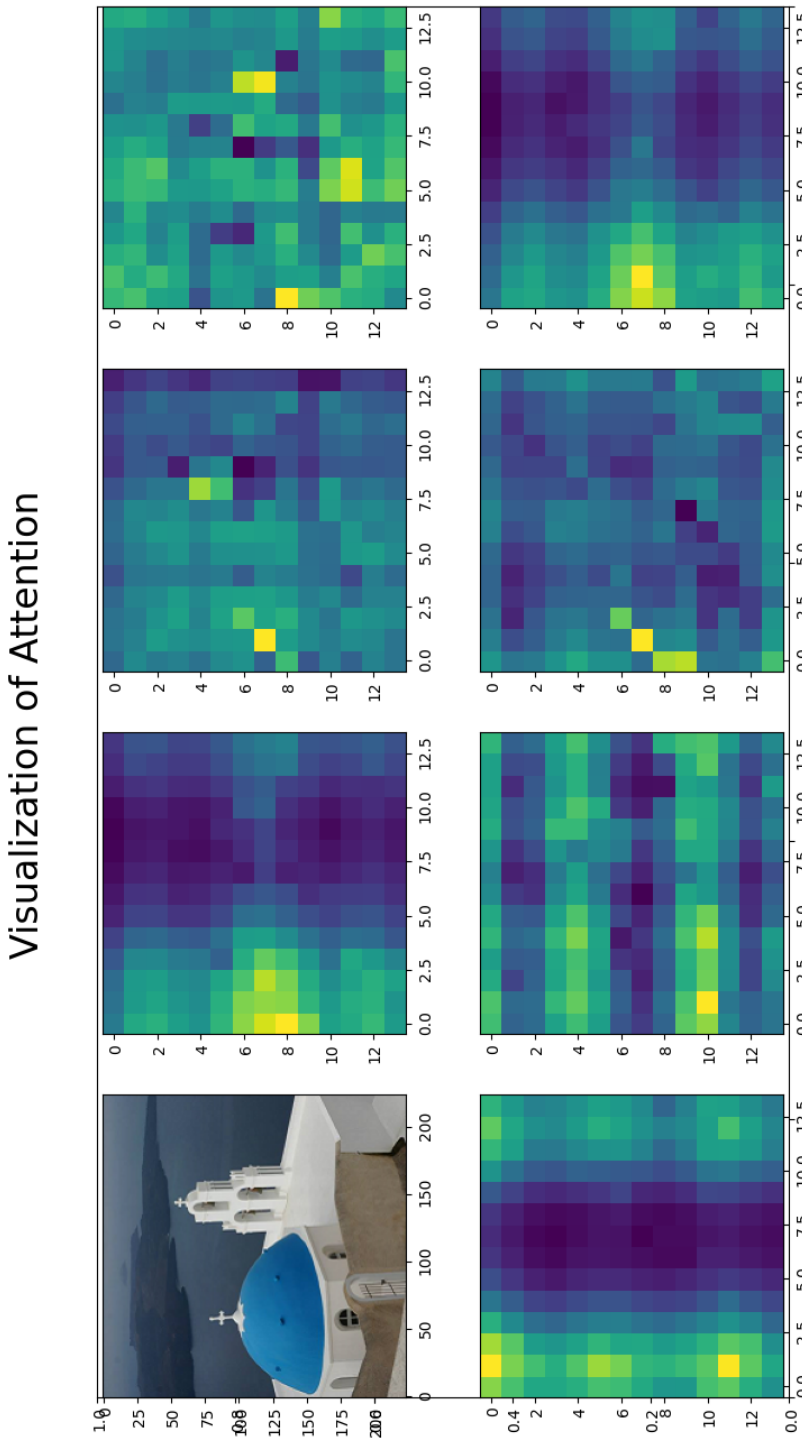

Figure 8: Visualization of 100th row of attention matrix of 0-7 heads. The bright regions indicate that the vision transformer pays more attention to those parts of the image.