DLCV HW3

R09521603

December 13, 2021

1 Problem 1

1.1 Report accuracy on validation data

a Model Discussion

There's eight types of pretrained model in the github of Pytorch-Pretrained-ViT. Due to the limit of time i chose the best model with the following hyperparameter and chose the best model architecture.

learning rate = 0.00001
batch_size = 16
optimizer = AdamW (with default setting)

Other settings include resizing all the image into 384x384, in training data i randomResizedCrop the image, RandomAffine with parameter 30, RandomHorizontalFlip with probability 50, ColorJitter with all parameter in 0.5, Normalize into (0.5, 0.5).
For the Validation data part, i only resize all the image into 384x384 and Normalize into (0.5, 0.5).

I chose the best model depends on the best validation accuracy, the following graph is all the curve of validation accuracy.

Figure 1: B_16.jpg

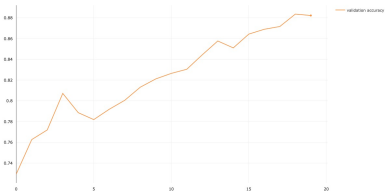


Figure 2: B_32.jpg

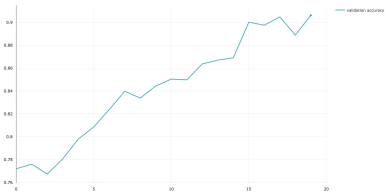


Figure 3: L_32.jpg

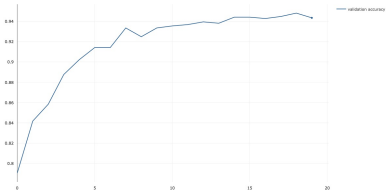


Figure 4: B_16_imagenet1k.jpg

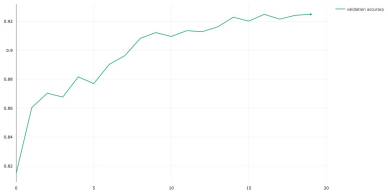


Figure 5: B_32_imagenet1k.jpg

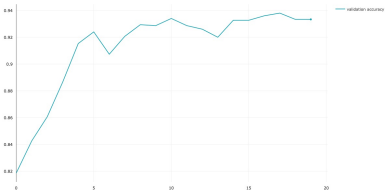


Figure 6: L_32_imagenet1k.jpg

The following table is the best validation score of each model

Model Name	Validation Accuracy
B_16	0.9155
B_32	0.8823
L_32	0.9062
B_16_imagenet1k	0.9433
B_32_imagenet1k	0.9249
L_32_imagenet1k	0.9383

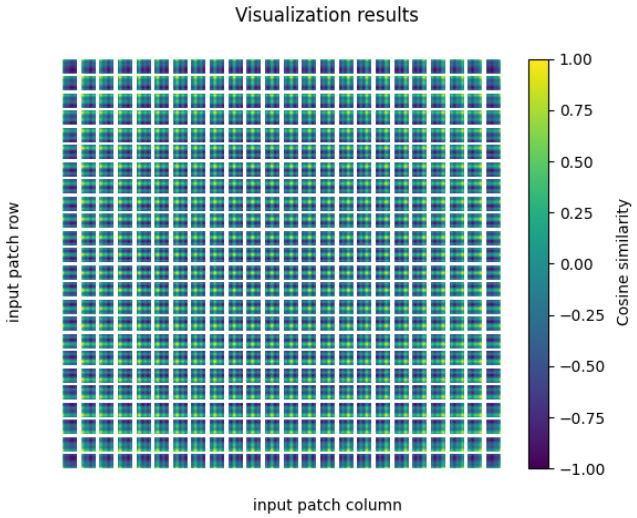
As the table point out B_16_imagenet1k has the higher accuracy, I chose the B_16_imagenet1k model with pretrained model as my final prediction model.

b Accuracy Report

Accuracy for validation data : 0.94333

1.2 Position Embeddings Visualization

a Graph



b Analysis

For the source code of Pytorch-Pretrained-ViT, they only implement 1D positional encoding. But the 1D positional encoding is received by training and initialized to zero at the begining. For my graph we can inspect that the positional encoding parameter do learned how to represent their own position in each patch. For example, for the patch that is closer to the center, their positional encoding similarity will be higher when closing to the center. Another inspection is that the positional encoding similarity not only have high value close to their position but also have high value on their vertical or horizontal axis.

1.3 Attention Map Visualization

a Graph

Figure 7: 26_5064.jpg

Figure 8: 29_4718.jpg

Figure 9: 31_4838.jpg

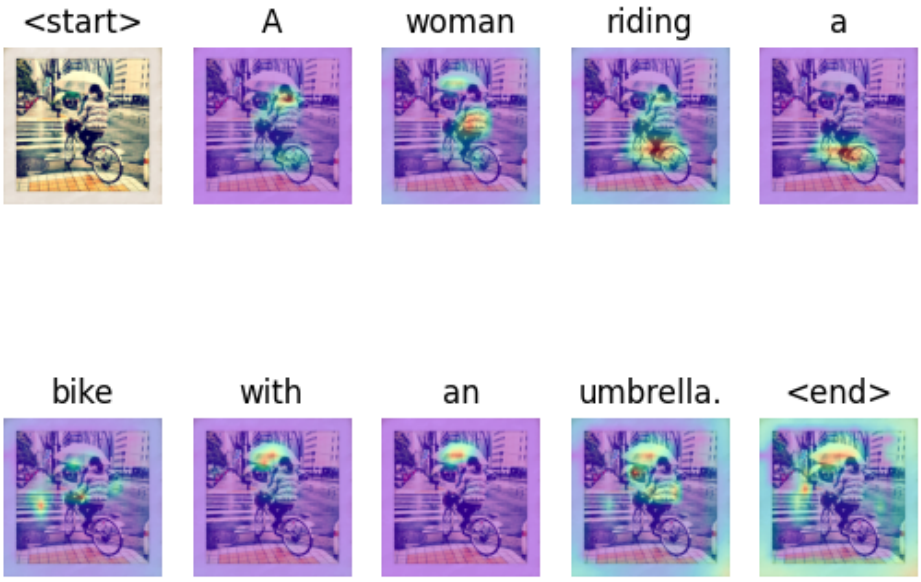
b Analysis

For each figure we can observe that in the region of animal we can get higher attention. But the result is not as delicate as the figure TA gave us. The reason of this result is that in my model i get each patch of image by 16x16 size, which highly reduce the sequence size of transformer. In my implementation i only receive attention map of 24x24 size. In the origin output attention map it look like the graph below. To receive more accommodate result i use linear interpolation with bilinear mode to upsample the origin attention map. Though the graph seems to extract useful information between each patch, to get more accurate attention map we need to clip the image with smaller patch size.

2 Problem 2

2.1 Report

a Analysis



I choose the image of bike.jpg as my report analysis since i think this image has get most interpretable result. We can find out that for most word the attention map has given great output. For example, for two words "a" the attention has nearly one center of attention. Another example is that on the "riding" part the attention has attended most on the bike. One of the amazing outcome is that at the "woman" part the attention has attend on the person patch, but how can it predict correctly that the person is a woman is what makes this amazing. Another inspection is that across all the image we usually get similar output on "end " part, which is most attention gather at the edge of the image. But i still think there one part that is hard to explain, which is the part "bike" word. It should attend most on the bike patch but instead it attend on the road, so how model predict correctly still have some mystery that is hard to explain.

b Difficulty

In this part the most hard part is to understand the whole architecture of the code of ViT. But once i look through all the code, this problem is easy. I didn't recalculate the attention map by myself but simple extract the second element of output of multiheadAttention in the decoder part.

References