DLCV – HW3

台科資工 M11015Q02 柯元豪

Problem 1

1. *Report accuracy of your model on the validation set.*

a. analyze the results with different settings

```
### All Setting ###
(L_16_imagenet1k)
Image_size : 384
optimizer : SGD
lr : 1e-4
weight decay: 3e-4
momentum = 0.9
Epoch: 10
early stop: 2
training data transform:
    transforms.Compose([
        transforms.RandomResizedCrop((image_size,image_size)),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize([0.5],[0.5])
    ])
Epoch 5 (best acc): 95.67%
----------------------------
(L_16_imagenet1k)
Image_size : 384
optimizer : SGD
lr : 1e-5
weight decay: 3e-4
momentum = 0.9
Epoch: 20
early stop: 2
training data transform:
    transforms.Compose([
        transforms.RandomResizedCrop((image_size,image_size)),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize([0.5],[0.5])
    ])
Epoch 19 (best acc): 95.67%
========================================
```

```
(L_32_imagenet1k)
Image_size : 384
optimizer : SGD
lr : 1e-4
weight decay: 3e-4
momentum = 0.9
Epoch: 10
early stop: 2

training data transform:
    transforms.Compose([
        transforms.RandomResizedCrop((image_size,image_size)),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize([0.5],[0.5])
    ])

Epoch 10 (best acc): 94.13%
###################
```

Discussion:

實作前就有先看過 paper 哪個模型的 performance 最好，因此一開始就直接使用 *L_16_imagenet1k*，由於 pretrained 好的模型本身表現已經很好，將 output class 改成 37 個去 finetune 模型收斂速度很快，5 個迭代左右就收斂了，後來又再嘗試調小 learning rate，雖然收斂速度比較慢，但到了 20 個 epoch 左右收斂的 performance 也是差不多。後來有再嘗試過 L_32 (patch size=32*32)的模型與各種 data 的 augmentation，表現大概也都在 baseline 上下而已，因此最終結果還是採用了 *L_16_imagenet1k*，參數如 (b) 所敘。

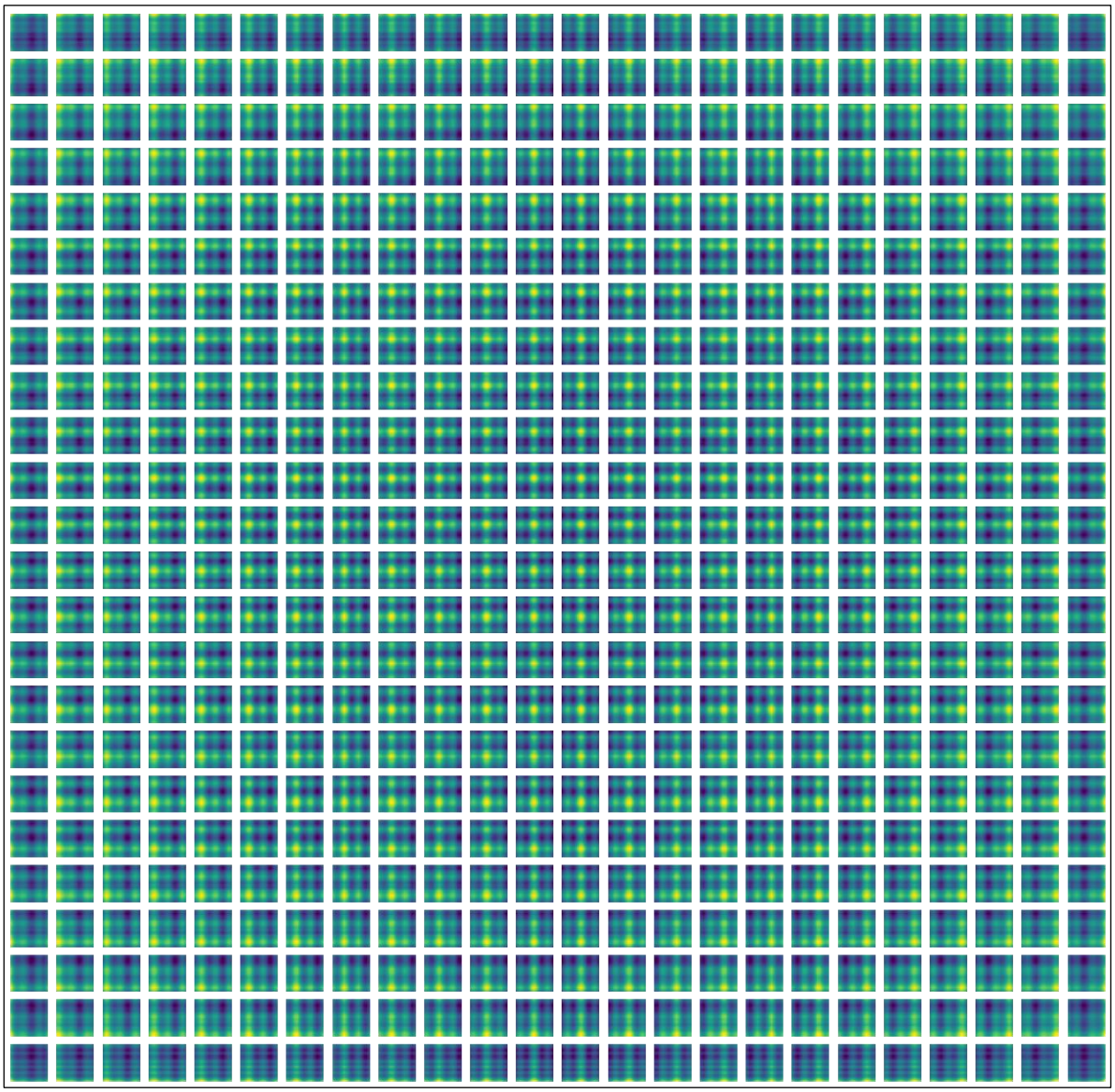
b. Clearly mark out a single final result

```
## final result's setting ##
(L_16_imagenet1k)
Image_size : 384
optimizer : SGD
lr : 1e-4
weight decay: 3e-4
momentum = 0.9
Epoch: 10
early stop: 2
training data transform:
    transforms.Compose([
        transforms.RandomResizedCrop((image_size,image_size)),
```

```
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize([0.5],[0.5])
    ])
##########################
```

2. Visualize position embeddings

a. Visualize cosine similarities from all positional embeddings



b. Discuss or analyze the visualization results

取 positional embedding 有用過兩種方法，畫出來結果都一樣。一個是改 source code，把 model.py 裡面 PositionalEmbedding1D 的 self.pos_embedding 拉出來就是結果了；另一個方法則是去 register hook，把 PositionalEmbedding1D (A) & PatchEmbedding (B) 的 forward 結

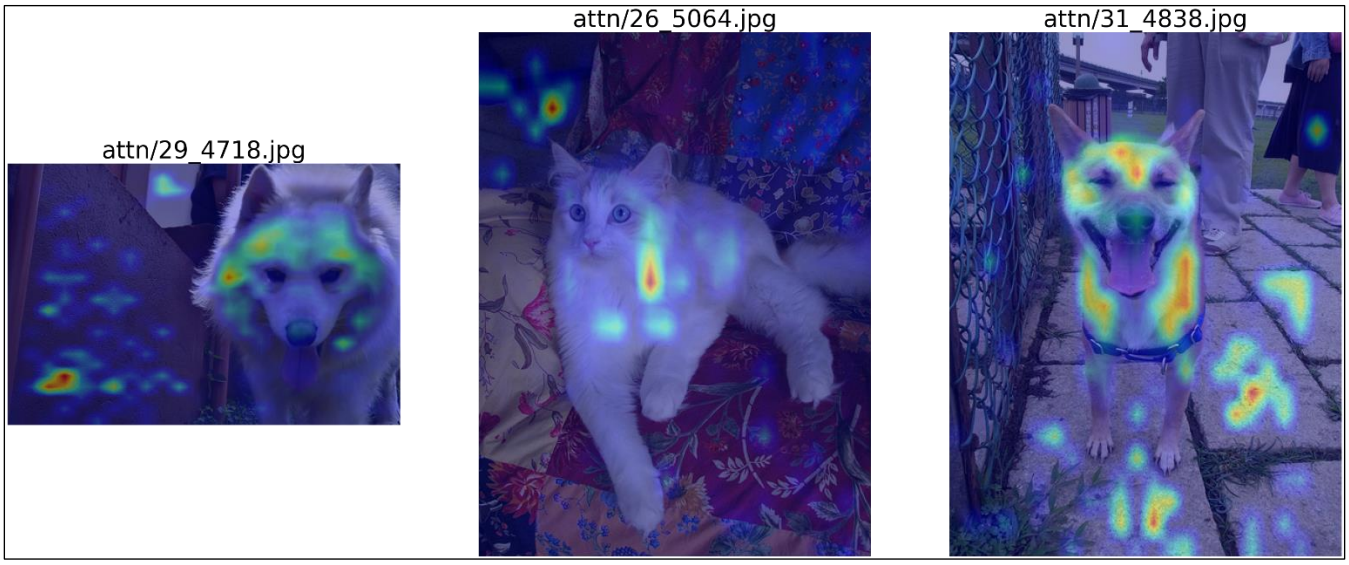
果拉出來，將 A – B 即是 PositionalEmbedding1D 中的 self.pos_embedding。

視覺化前先將 positional embedding 跟自己的 transpose 做 cosine similarity，再將每個 patch 畫出來就是上圖那樣了，仔細看也能看出 position embedding 的資訊。

(Image size = 384, 因此有 24*24 個 patch)

*3. Visualize *attention map of 3 images (p1_data/val/26_5064.jpg, p1_data/val/29_4718.jpg, p1_data/val/31_4838.jpg)*

a. Visualize the attention map

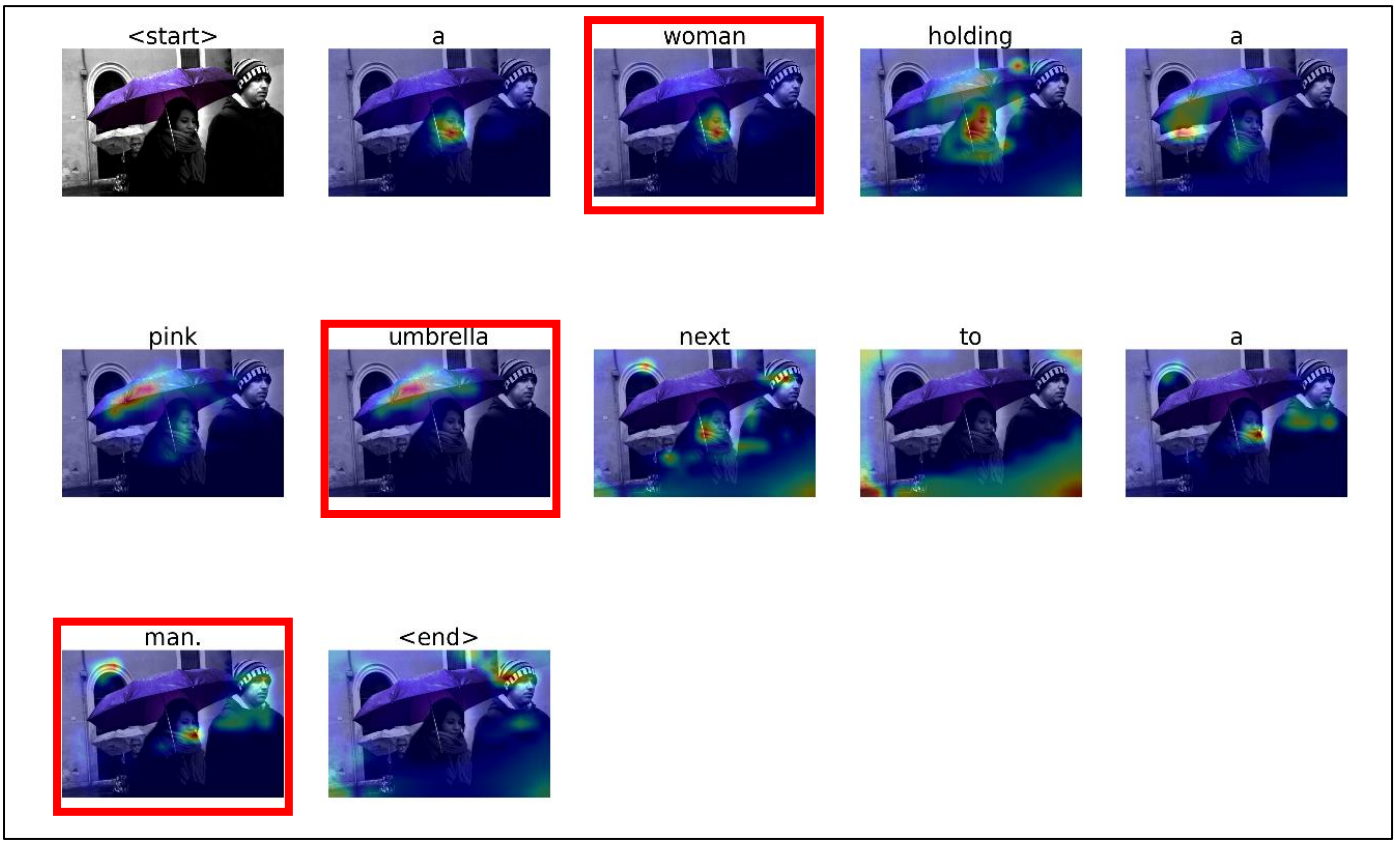


b. Discuss or analyze the visualization results

這次選擇的模型準確率有 95.6%左右，從繪製結果來看，可以看出三張圖片大部分注意力較集中的地方都是在我們要辨識的目標身上，只有少部分會聚焦在其他不重要的區塊。我嘗試過拿 performance 較差的模型繪製 attention map，同樣的圖片，注意力集中的部分會沒那麼準確。因此我認為從準確率高低來看，目前選擇的模型還是有學習好重點，以至於後面的分類也有好的表現。

Problem 2

Choose one test image and show its visualization result in your report.



a. **Analyze the predicted caption and the attention maps for each word. Is the caption reasonable? Does the attended region reflect the corresponding word in the caption?**

這次分析採用的是 catr 的 v2 model 繪製倒數第二層 decoder layer 的 cross-attention，從繪製結果可以看出產生的 image caption 是很準的，且 attention maps 也都有對應到正確的單字。採用其他版本的 pretrained model 預測出來的 caption 有的會與 ground truth 有一些誤差，且繪製的 attention maps 也會有些單字無法正確匹配；且取 decoder 的倒數第二層效果也較最後一層好一點。

b. **Discuss what you have learned or what difficulties you have encountered in this problem.**

這題最困難的部分我覺得是去找出正確的 attention maps，即使架構看懂也明白題目需求，但還是需要慢慢 trace code，將幾個特定 layer 輸出結果的 dimension dump 出來看，藉此去找

出包含 image patches 與 positional embedding 資訊的 output，最後得到的 attention map dimension 會是 (max positional embedding size , image patches size)，便可將對應位置的 word 與 attention map 視覺化出來。因為這題是使用統一的模型架構，所以實作過程也很容易知道自己繪製出來的 attention map 是對是錯，若取得的 attention map 是正確的，我認為應該要和簡報呈現的差不多，這也讓自己的實作過程多了一個衡量標準，可以不斷從錯誤中改進，以找出正確的結果。

這題實作雖然不需要再去訓練模型，但還是讓我更加了解 transformer 的架構，對於 image cation 相關的技術與題目也變得更有興趣。

Reference

https://github.com/lukemelas/PyTorch-Pretrained-ViT

https://github.com/saahiluppal/catr

Collaborators

M11015Q12 黃柏翰