作業 part1:

最一開始照著範例的 YT 影片操作，使用 Incepton_v3 有遇到一些問題

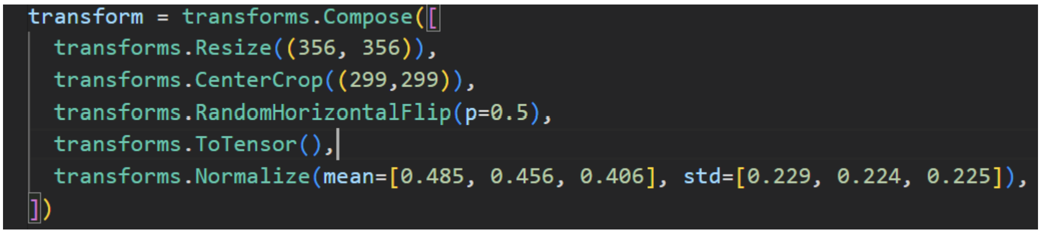
Aux_logits 一直無法使用，還有 imgs,captions = dataL.next()也有點小問題

一一排除掉後就可以開始 Run 了，本來用 google colab 的免費資源跑，後來真的太慢了，才使用本機端訓練模型。

第一次訓練結果:

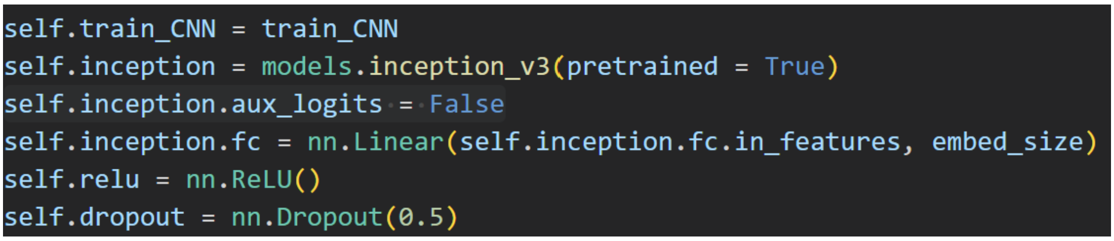
因為跑太久了，想說先訓練一 epoch 試看看，結果慘不忍睹，

Data augmentation:



```
transform = transforms.Compose([
    transforms.Resize((356, 356)),
    transforms.CenterCrop((299,299)),
    transforms.RandomHorizontalFlip(p=0.5),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

encoderCNN:



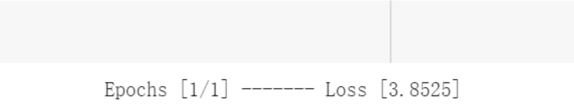
```
self.train_CNN = train_CNN
self.inception = models.inception_v3(pretrained = True)
self.inception.aux_logits = False
self.inception.fc = nn.Linear(self.inception.fc.in_features, embed_size)
self.relu = nn.ReLU()
self.dropout = nn.Dropout(0.5)
```

embed_size = 256

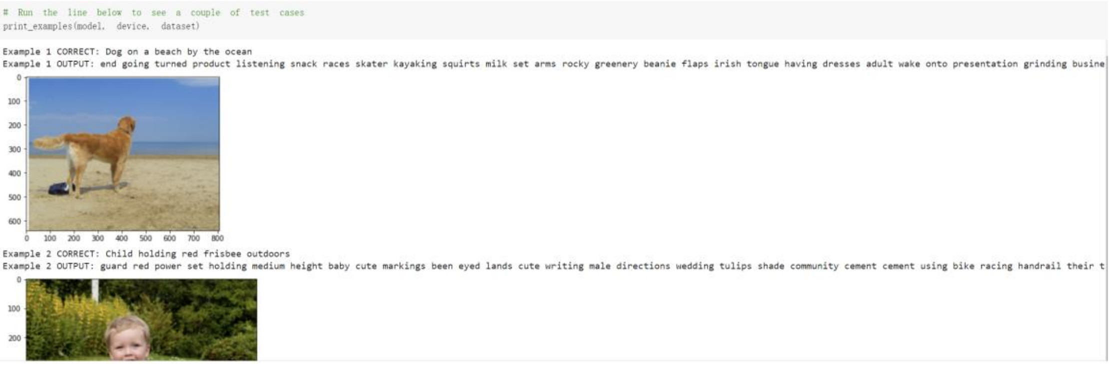
hidden_size = 256

learning rate = 3e-4

結果:



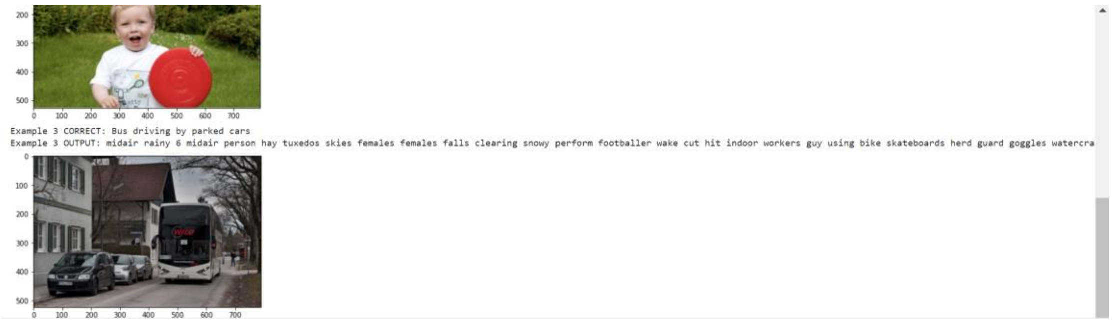
```
Epochs [1/1] ------- Loss [3.8525]
```



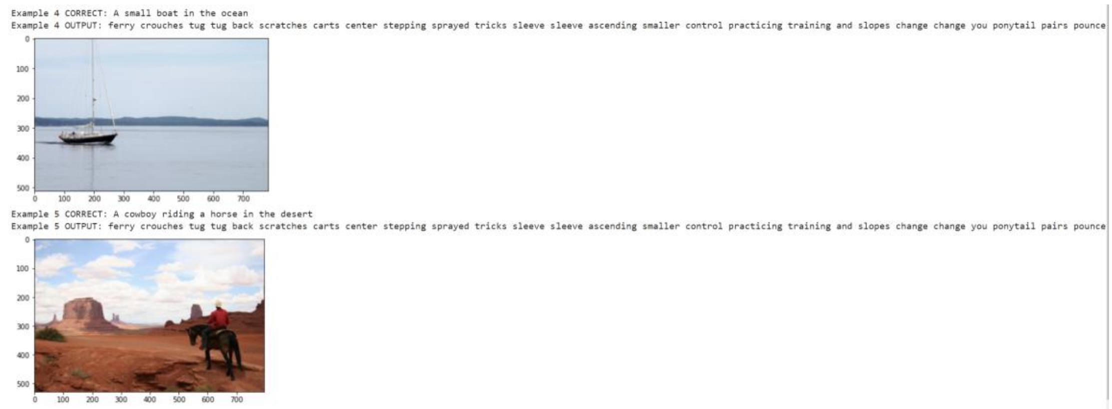
```
# Run the line below to see a couple of test cases
print_examples(model, device, dataset)
```

```
Example 1 CORRECT: Dog on a beach by the ocean
Example 1 OUTPUT: end going turned product listening snack races skater kayaking squirts milk set arms rocky greenery beanie flaps irish tongue having dresses adult wake onto presentation grinding busine
```

```
Example 2 CORRECT: Child holding red frisbee outdoors
Example 2 OUTPUT: guard red power set holding medium height baby cute markings been eyed lands cute writing male directions wedding tulips shade community cement cement using bike racing handrail their t
```



```
Example 3 CORRECT: Bus driving by parked cars
Example 3 OUTPUT: midair rainy 6 midair person hay tuxedos skies females females falls clearing snowy perform footballer wake cut hit indoor workers guy using bike skateboards herd guard goggles watercra
```

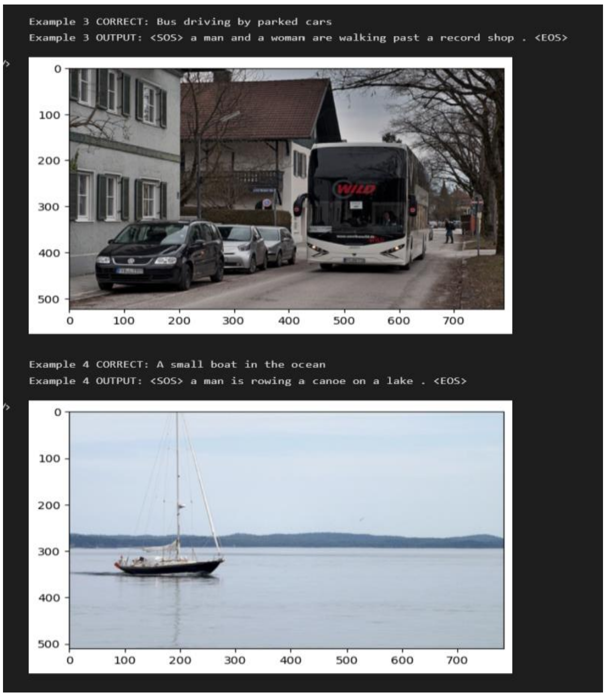


```
Example 4 CORRECT: A small boat in the ocean
Example 4 OUTPUT: ferry crouches tug tug back scratches carts center stepping sprayed tricks sleeve sleeve ascending smaller control practicing training and slopes change change you ponytail pairs pounce
```

```
Example 5 CORRECT: A cowboy riding a horse in the desert
Example 5 OUTPUT: ferry crouches tug tug back scratches carts center stepping sprayed tricks sleeve sleeve ascending smaller control practicing training and slopes change change you ponytail pairs pounce
```

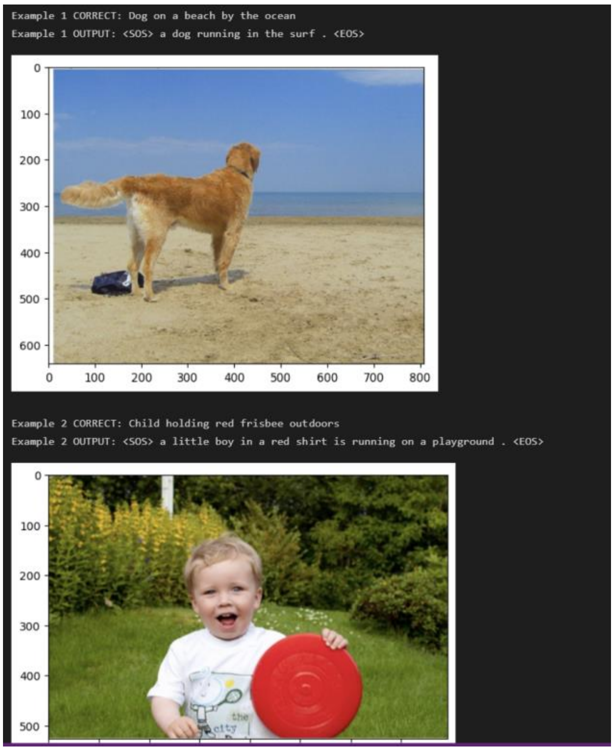
第二次訓練:

是從本機端跑的，大致上與第一次訓練一樣，只是我把 epoch 數量增加到 20



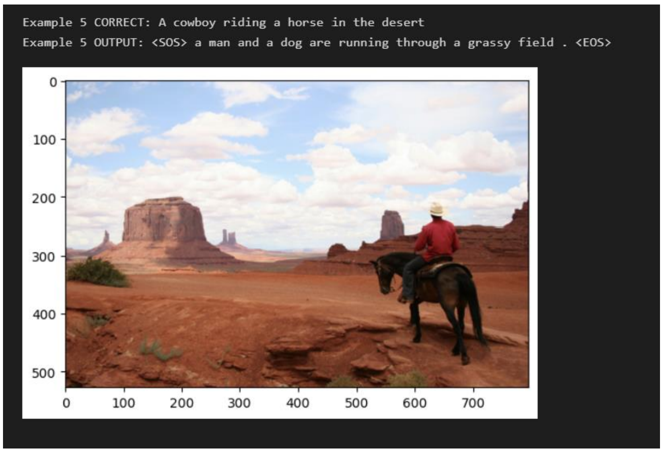
```
Example 3 CORRECT: Bus driving by parked cars
Example 3 OUTPUT: <SOS> a man and a woman are walking past a record shop . <EOS>
```

```
Example 4 CORRECT: A small boat in the ocean
Example 4 OUTPUT: <SOS> a man is rowing a canoe on a lake . <EOS>
```



```
Example 1 CORRECT: Dog on a beach by the ocean
Example 1 OUTPUT: <SOS> a dog running in the surf . <EOS>
```

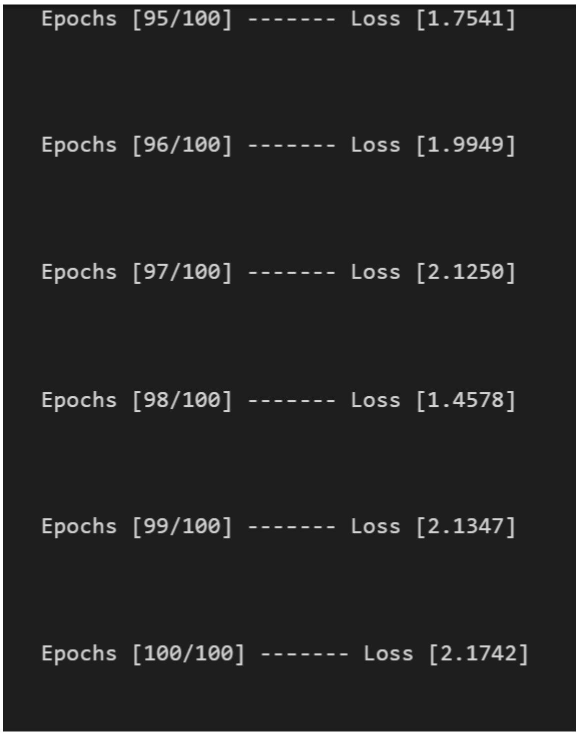
```
Example 2 CORRECT: Child holding red frisbee outdoors
Example 2 OUTPUT: <SOS> a little boy in a red shirt is running on a playground . <EOS>
```



```
Example 5 CORRECT: A cowboy riding a horse in the desert
Example 5 OUTPUT: <SOS> a man and a dog are running through a grassy field . <EOS>
```

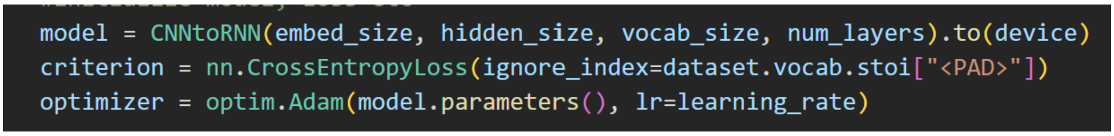
第三次訓練:

我執行了 100epoch，但是 Loss 還是卡在 2 左右



```
Epochs [95/100] ------- Loss [1.7541]

Epochs [96/100] ------- Loss [1.9949]

Epochs [97/100] ------- Loss [2.1250]

Epochs [98/100] ------- Loss [1.4578]

Epochs [99/100] ------- Loss [2.1347]

Epochs [100/100] ------- Loss [2.1742]
```

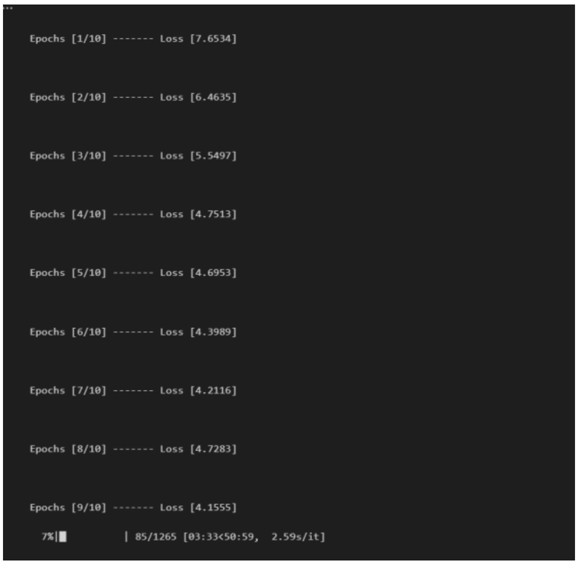
Hyperparameters 的部分都使用一樣的:



```
model = CNNtoRNN(embed_size, hidden_size, vocab_size, num_layers).to(device)
criterion = nn.CrossEntropyLoss(ignore_index=dataset.vocab.stoi["<PAD>"])
optimizer = optim.Adam(model.parameters(), lr=learning_rate)
```

第四次訓練

Lr 設定為 3e-6

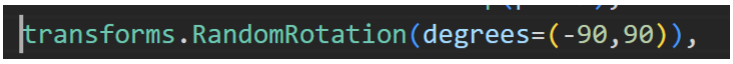


```
...
    Epochs [1/10] ------- Loss [7.6534]

    Epochs [2/10] ------- Loss [6.4635]

    Epochs [3/10] ------- Loss [5.5497]

    Epochs [4/10] ------- Loss [4.7513]

    Epochs [5/10] ------- Loss [4.6953]

    Epochs [6/10] ------- Loss [4.3989]

    Epochs [7/10] ------- Loss [4.2116]

    Epochs [8/10] ------- Loss [4.7283]

    Epochs [9/10] ------- Loss [4.1555]
  7%|█         | 85/1265 [03:33<50:59,  2.59s/it]
```

看到這結果，就不想讓他跑完了

最後一次訓練:

我增加了圖片旋轉



```
transforms.RandomRotation(degrees=(-90,90)),
```

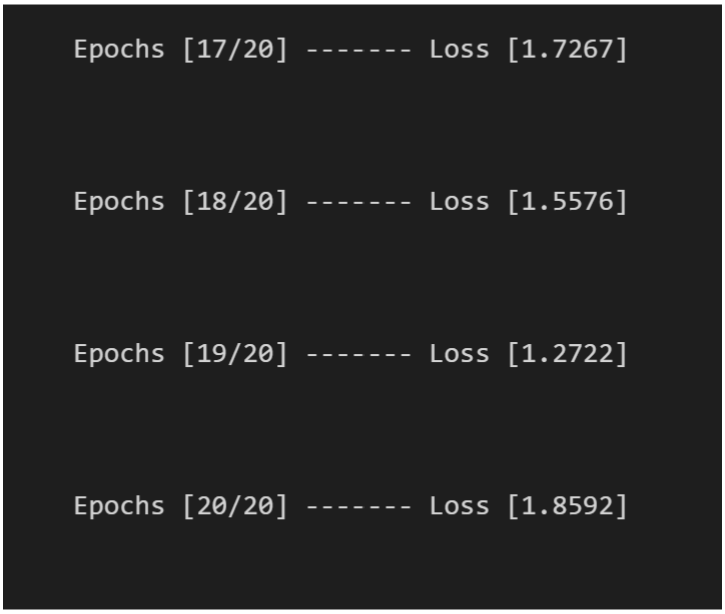
embed_size = 1024

hidden_size = 1024

learning rate = 3e-4

epoch = 20

結果如下圖，感覺好像好一些



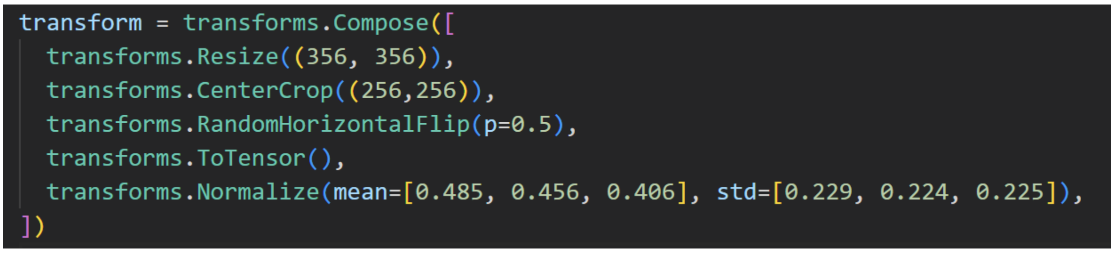
```
    Epochs [17/20] ------- Loss [1.7267]

    Epochs [18/20] ------- Loss [1.5576]

    Epochs [19/20] ------- Loss [1.2722]

    Epochs [20/20] ------- Loss [1.8592]
```

Part2

第一次訓練:

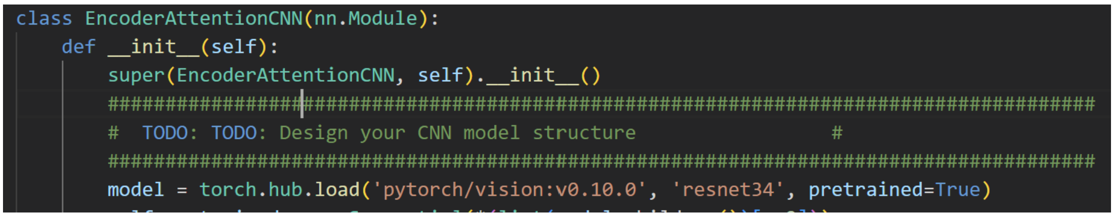
使用 resnet18

Data augmentations:



```
transform = transforms.Compose([
    transforms.Resize((356, 356)),
    transforms.CenterCrop((256,256)),
    transforms.RandomHorizontalFlip(p=0.5),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

CNN encoder:



```
class EncoderAttentionCNN(nn.Module):
    def __init__(self):
        super(EncoderAttentionCNN, self).__init__()
        #######################################################################################
        #  TODO: TODO: Design your CNN model structure                            #
        #######################################################################################
        model = torch.hub.load('pytorch/vision:v0.10.0', 'resnet34', pretrained=True)
```

Hyperparams:

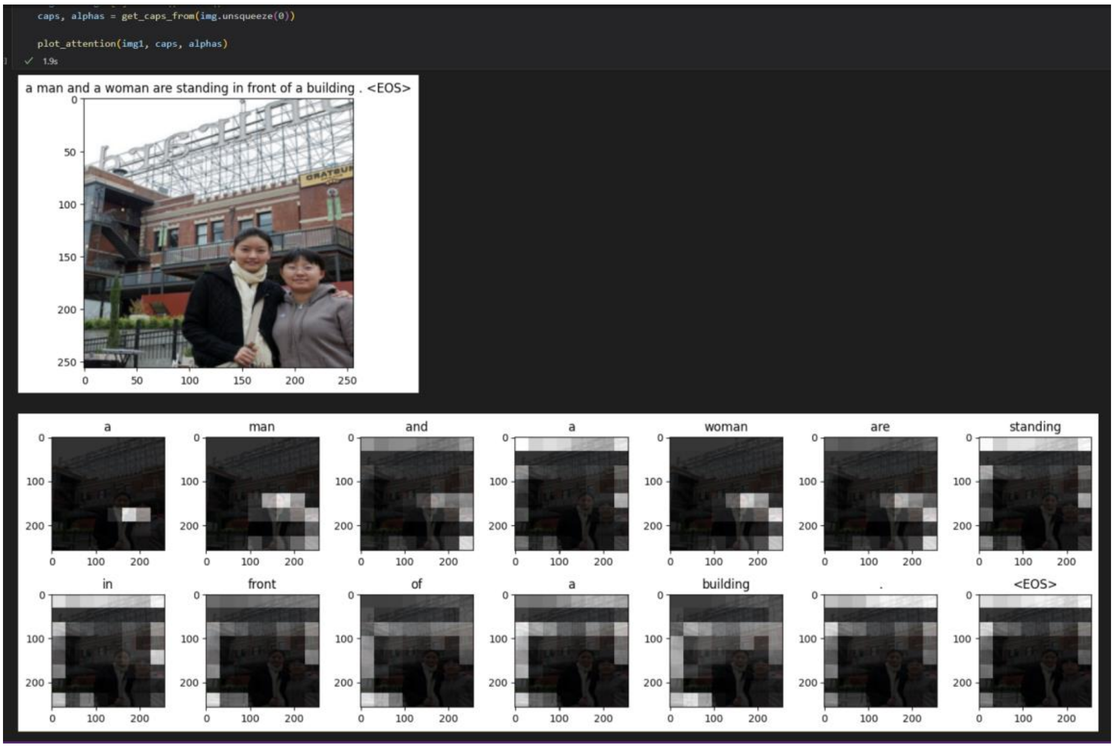
Emb_size = 300

Attention_dim = 256

Encoder_dim = 2048

Decoder_dim = 512

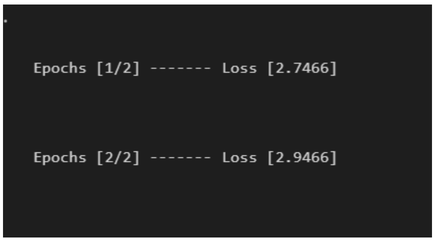
Lr = 3e-5



```
caps, alphas = get_caps_from(img.unsqueeze(0))

plot_attention(img1, caps, alphas)
```

✓ 1.9s

a man and a woman are standing in front of a building . <EOS>



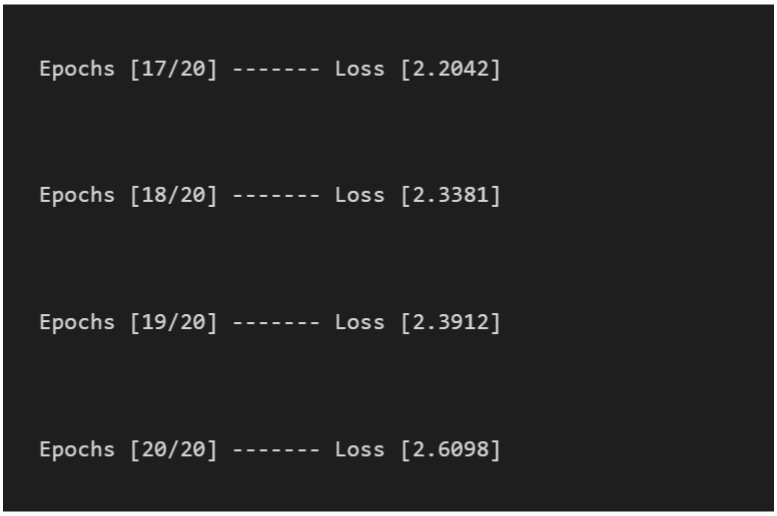
```
Epochs [1/2] ------- Loss [2.7466]


Epochs [2/2] ------- Loss [2.9466]
```

第二次訓練:

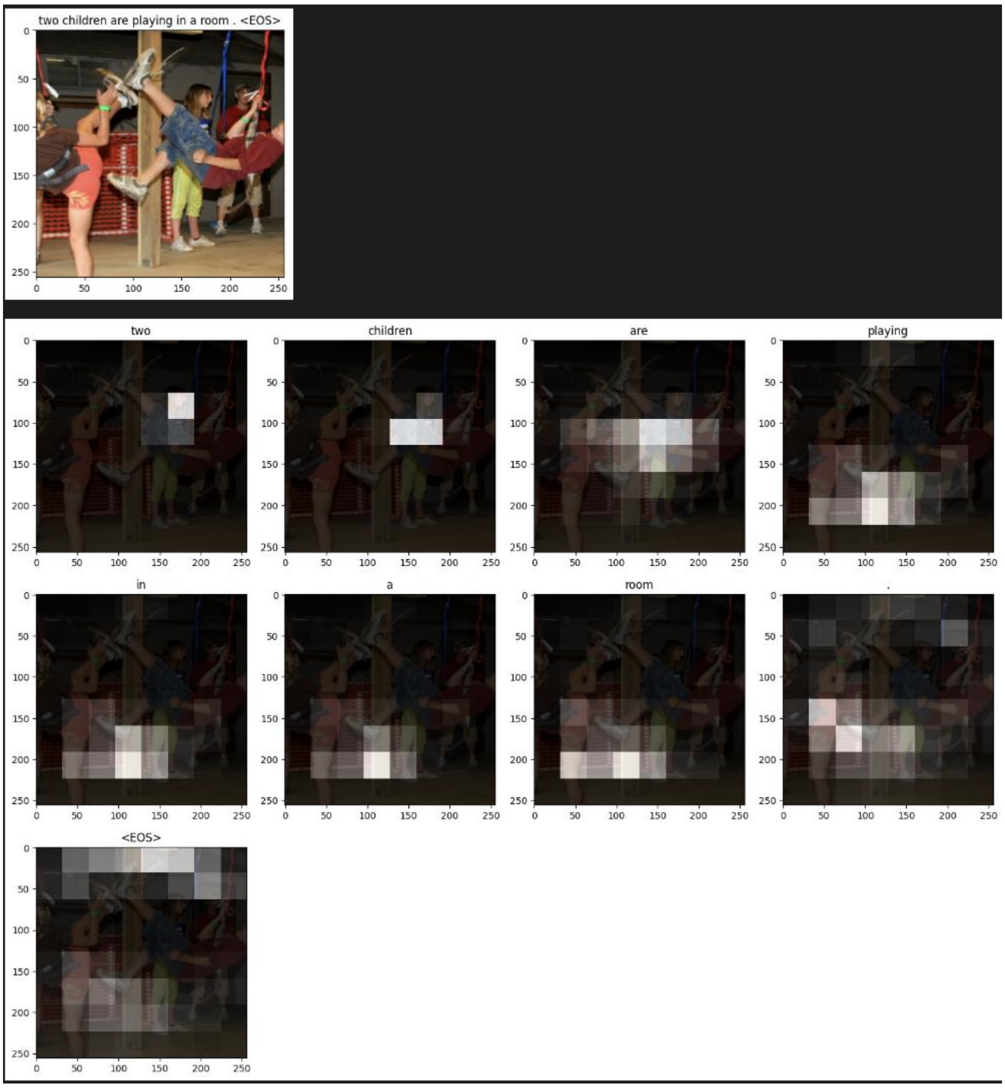
將 epoch 數增加到 20

其他沒有做更動

結果如下:



```
Epochs [17/20] ------- Loss [2.2042]


Epochs [18/20] ------- Loss [2.3381]


Epochs [19/20] ------- Loss [2.3912]


Epochs [20/20] ------- Loss [2.6098]
```



結果沒有很大的差異，模型應該還可以做其他調整，只是真的跑太久了!!!