# Deep Learning for Computer Vision

NTU, Fall 2023, homework3
電機所碩一 謝宗翰 r12921a10

## ● Problem 1: Zero-shot Image Classification with CLIP

1. *Methods analysis (3%)*

   ◆ *Previous methods (e.g. VGG and ResNet) are good at one task and one task only, and requires significant efforts to adapt to a new task. Please explain why CLIP could achieve competitive zero-shot performance on a great variety of image classification datasets.*

   因為CLIP在大量的圖像和對應的文字標題上進行訓練，並且CLIP的預訓練目標是最大化配對的圖像文本樣本的相似性，同時最小化未配對的樣本。

   它的目標是讓配對的圖片和文字之間的相似度最大，而未配對的圖片和文字之間的相似度最小。這種方法與以前的方法有所不同，因為CLIP並不直接優化特定的任務，而是在自然語言的指導下進行訓練。

   更具體地說，透過caption來構建一個線性分類器，而無需任何標記數據，然後它的強大的視覺表示可以以競爭性的性能執行任務。

2. *Prompt-text analysis (6%)*

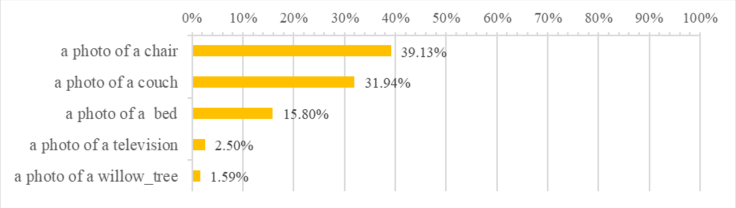
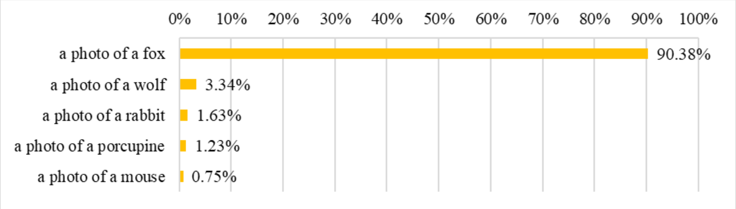
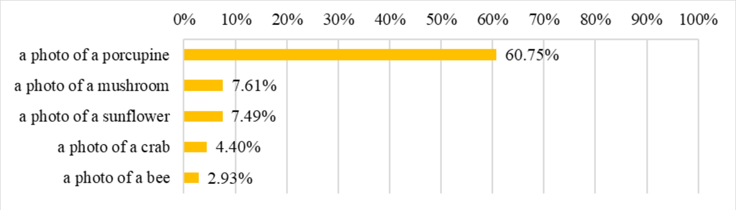
   ◆ *Please compare and discuss the performances of your model with the following three prompt templates:*

   i. *"This is a photo of {object}"*
   ii. *"This is not a photo of {object}"*
   iii. *"No {object}, no score."*

| Caption | Accuracy |
|---|---|
| This is a photo of {object} | 67.84% |
| This is not a photo of {object} | 69.52% |
| No {object}, no score. | 45.72% |

3. *Quantitative analysis (6%)*

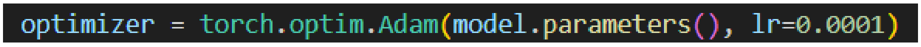
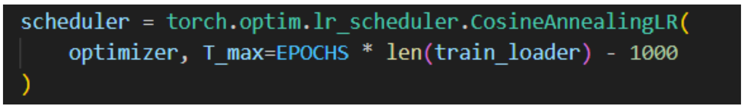
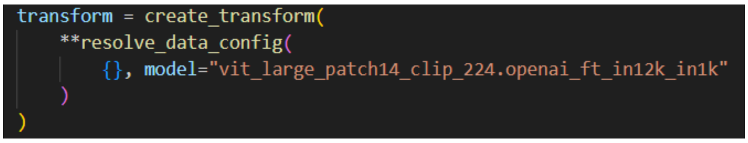
◆ *Please sample three images from the validation dataset and then visualize the probability of the top-5 similarity scores.*

| Picture | Top-5 similarity Scores |
|---|---|
| | <br> |
| | <br> |
| | <br> |

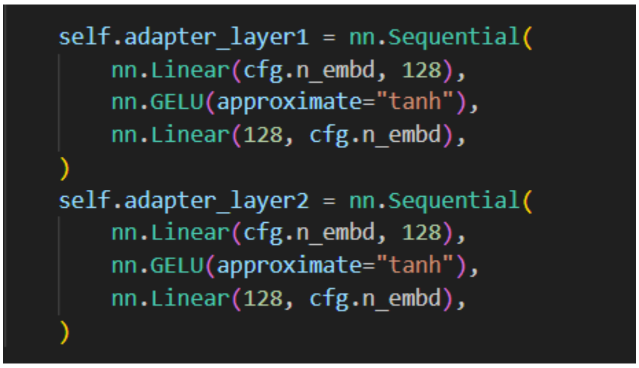
## Problem 2: PEFT on ViT Model for Image Captioning

1. *Evaluation metrics report*

◆ *Report your best setting and its corresponding CIDEr & CLIPScore on the validation data.*

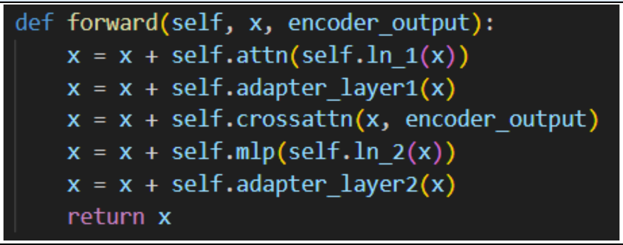
| Model |
|---|
| vit_large_patch14_clip_224.openai_ft_in12k_in1k |
| Optimizer |
| <br>`optimizer = torch.optim.Adam(model.parameters(), lr=0.0001)` |
| Scheduler |
| <br>`scheduler = torch.optim.lr_scheduler.CosineAnnealingLR(`<br>`    optimizer, T_max=EPOCHS * len(train_loader) - 1000`<br>`)` |
| Transform |
| <br>`transform = create_transform(`<br>`    **resolve_data_config(`<br>`        {}, model="vit_large_patch14_clip_224.openai_ft_in12k_in1k"`<br>`    )`<br>`)` |

Adapter 架構



```
self.adapter_layer1 = nn.Sequential(
    nn.Linear(cfg.n_embd, 128),
    nn.GELU(approximate="tanh"),
    nn.Linear(128, cfg.n_embd),
)
self.adapter_layer2 = nn.Sequential(
    nn.Linear(cfg.n_embd, 128),
    nn.GELU(approximate="tanh"),
    nn.Linear(128, cfg.n_embd),
)
```

修改 decoder 的架構



```
def forward(self, x, encoder_output):
    x = x + self.attn(self.ln_1(x))
    x = x + self.adapter_layer1(x)
    x = x + self.crossattn(x, encoder_output)
    x = x + self.mlp(self.ln_2(x))
    x = x + self.adapter_layer2(x)
    return x
```

Best (加了 adapter)

| CIDEr | CLIP Score |
|---|---|
| 0.91032 | 0.71471 |

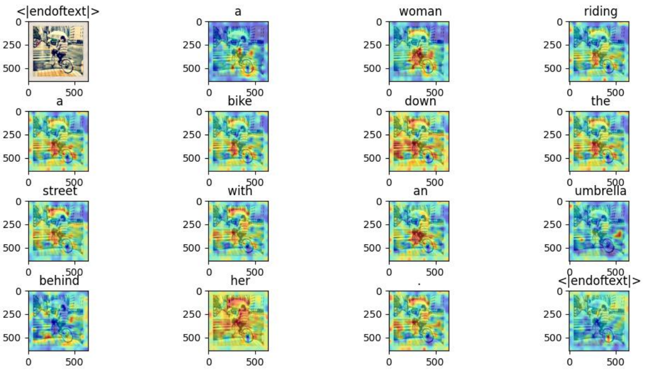
- *Report 3 different attempts of PEFT and their corresponding CIDEr & CLIPScore.*

|  | Adapter | Prefix | Lora |
|---|---|---|---|
| CIDEr | 0.921 | 0.884 | 0.893 |
| CLIPScore | 0.717 | 0.710 | 0.721 |

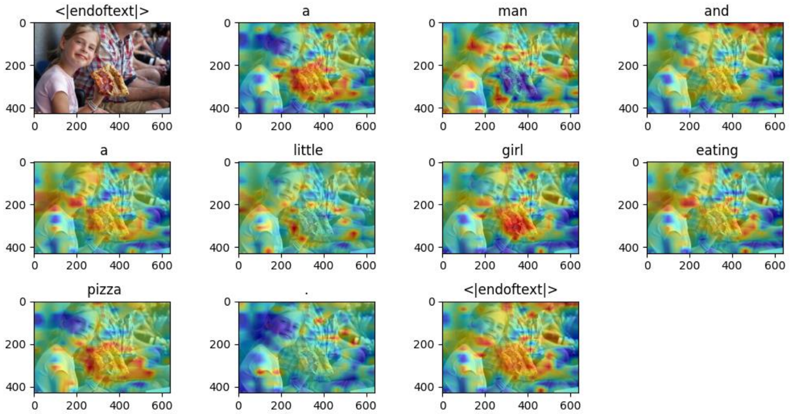
2. *Visualization of Attention in Image Captioning*

- *please visualize the predicted caption and the corresponding series of attention maps.*

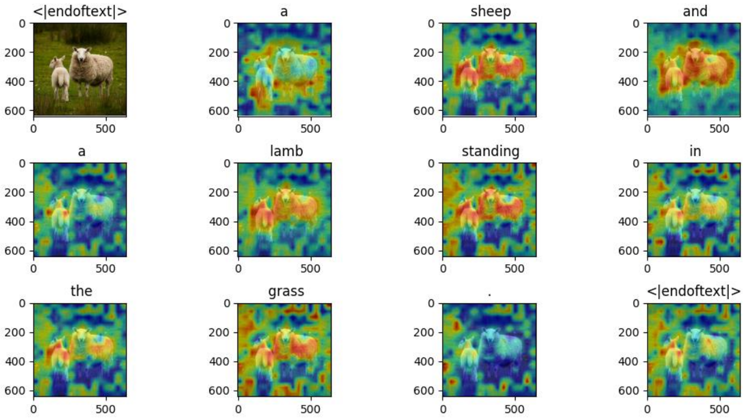
| Bike.jpg |
|---|



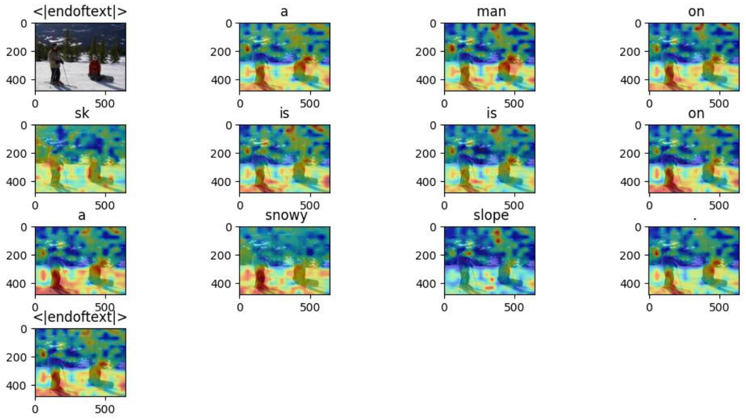
girl.jpg



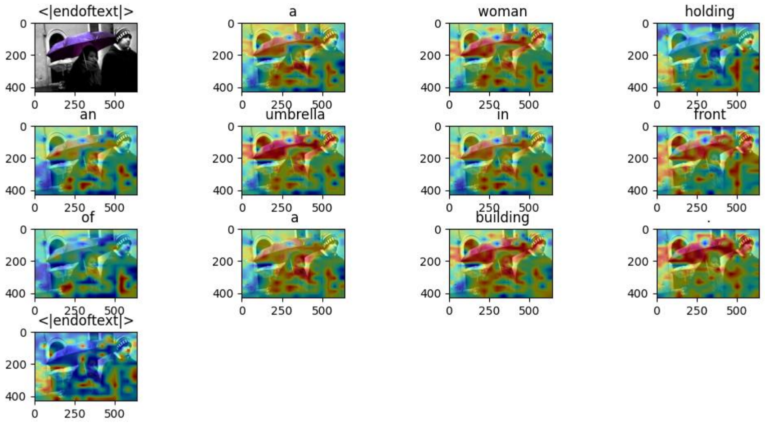
sheep.jpg



ski.jpg



umbrella.jpg

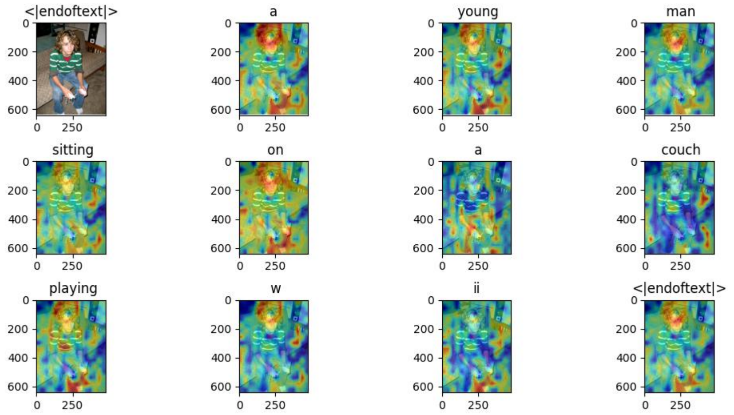
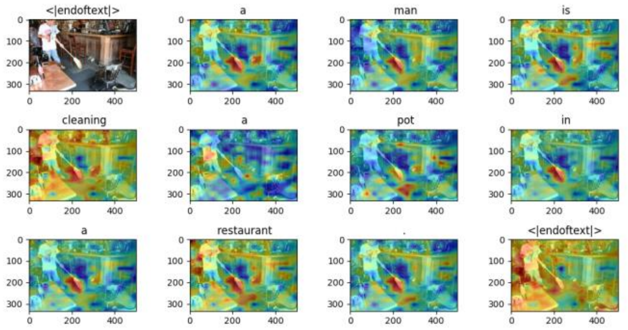


- *According to CLIPScore, you need to:*

  i. *visualize top-1 and last-1 image-caption pairs*

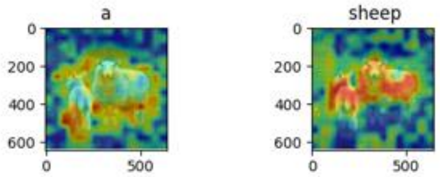
  ii. *report its corresponding CLIPScore in the validation dataset of problem 2.*

| Best | |
|---|---|
| 000000539189.png | CLIPScore = 1.03820 |

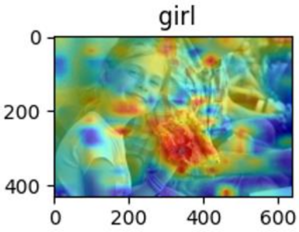
| | |
|---|---|
|  | |
| Worst | |
| 491987177.png | CLIPScore = 0.34393 |
|  | |

- *Analyze the predicted captions and the attention maps for each word according to the previous question.*
  - *Is the caption reasonable?*
  - *Does the attended region reflect the corresponding word in the caption?*

1. 大部分看起來都蠻合理的
2. 字詞都有對應到該注意的地方，而我覺得表現最好的幾個應該是 sheep、000000539189.png、umbrella。以 sheep 為例，只要有對應到 sheep 的名詞動詞形容詞都有很明顯的聚焦(紅色地方)，而遇到 a、the、.之類無關圖片的字詞，注意力就會散開來。



但還是有少部分怪怪的地方，例如 girl.png，他的注意力很常聚焦在 pizza 上，girl 字詞或 a 也都會聚焦在 pizza 上。



- **Reference**

大神：Chatgpt4

Beam Search：

[pytorch_beam_search/src/pytorch_beam_search/autoregressive/search_algorithms.py at master · jarobyte91/pytorch_beam_search (github.com)]