DLCV HW3 Report

NTU CSIE, R12922051
資工碩一 陳韋傑

Q1-1: Methods Analysis

Methods like VGG, ResNet are trained using traditional supervised learning, that means they are only able to identify the objects they've seen during training, or they need to be finetuned for downstream tasks to get better results.

On the contrary, CLIP was pretrained on a large scale image-text dataset, and utilized contrastive learning to learn the relative relationships between images and texts. By pulling similar image-text pairs closer while pushing dissimilar pairs farther, CLIP learned a good representation for images and texts in the embedding space, thus makes CLIP more generalized and able to do zero-shot learning by finding the most similar or possible one when the label is unseen before.

Q1-2: Prompts Analysis

I used ViT-L/14 as visual encoder, and try out six different prompts. (Results are listed in next page.) The prompt ‘a photo of a {object}’ gets highest accuracy, I think the reason is that in the original paper* they used this prompt to train CLIP.

Also, the difference between ‘This is a photo of {object}’ and ‘This is not a photo of {object}’ is not significant (even higher in the ‘not’ case), I think it means the text encoder didn’t capture the semantic meaning of ‘not’ well.

Finally, using only ‘{object}’ as prompt also gives decent result, although not as good as the original prompt ‘a photo of a {object}’. But using ‘{object} {object} {object} {object} {object}’ will drop model performance significantly.

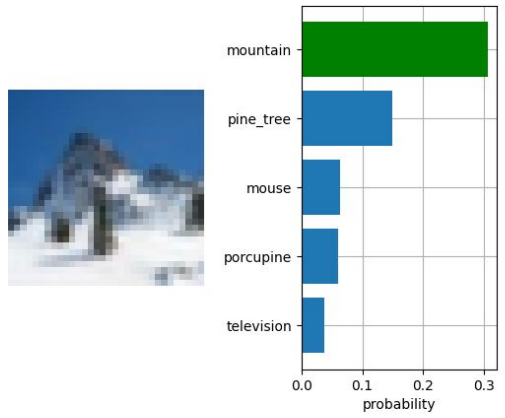
* Alec Radford et al. Learning Transferable Visual Models From Natural Language Supervision.

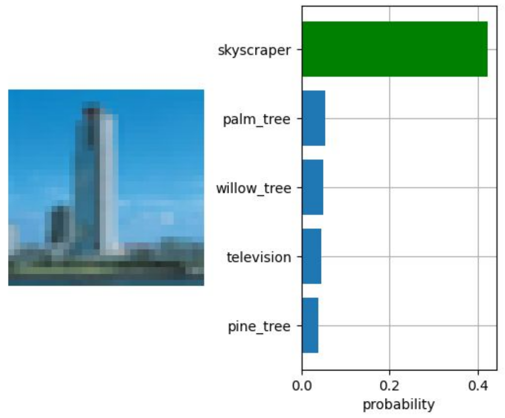
Q1-2: Prompts Analysis

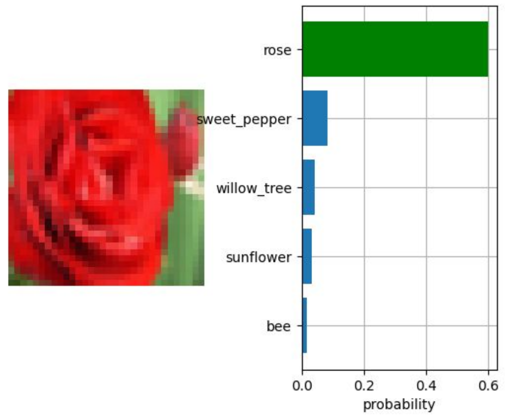
Prompts	Score (Validation Accuracy)
{object}	0.7304
{object} {object} {object} {object} {object}	0.5680
a photo of a {object}	0.8140
This is a photo of {object}	0.6784
This is not a photo of {object}	0.6952
No {object}, no score.	0.4572

Q1-3: Quantitative Analysis

Here are three successful cases, using ‘a photo of a {object}’ as prompt:

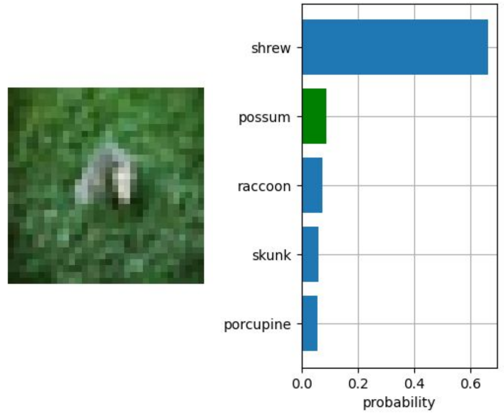


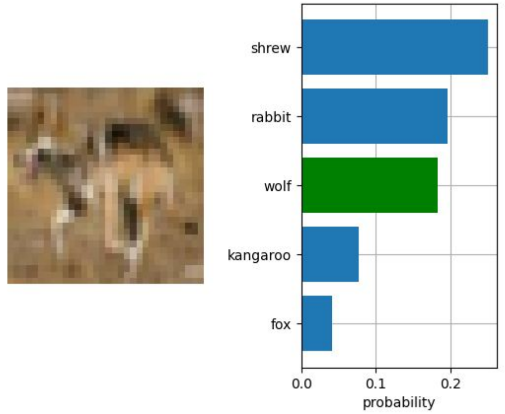


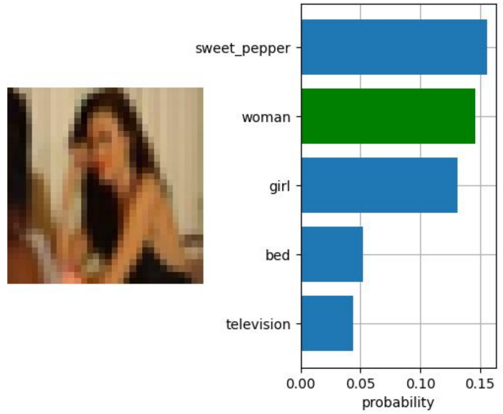


Q1-3: Quantitative Analysis (Appendix: Failure Cases)

Here are three failure cases, using ‘a photo of a {object}’ as prompt:







Q2-1: Parameters

- Encoder: vit_large_patch14_clip_224
- Encoder Projection: Linear (1024, 768)
- Epoch: 10
- Learning Rate: 1e-4
- Weight Decay: 1e-5
- Batch Size: 64
- Text Max Sequence Length: 128 (prefix not included)
- Text Padding: To max sequence length
- Image Padding: None
- Loss: Cross Entropy
- Optimizer: Adam
- Scheduler: Cosine Annealing
- Preprocessing:
    - T.Resize([224, 224]),
    - T.RandomHorizontalFlip(),
    - T.RandomRotation(10),
    - T.RandomGrayscale(),
    - T.ToTensor(),
    - T.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])

Q2-1: Metrics (Greedy, decoding step = 50)

Based on these results, all the following models are picked based on CIDEr score.

Model Saved Methods	CIDEr Score	CLIP Score
Settings: LoRA, rank=8		
By Best Validation Loss	0.8426	0.7185
By Best CIDEr Score	**0.8611**	0.7223
By Best CLIP Score	0.8269	**0.7245**

Q2-1: Metrics (Decoding Strategy)

Based on these results, I'll use beam =5 when using beam search strategy.

Beam Count	CIDEr Score	CLIP Score
Settings: LoRA, rank=8		
Greedy	0.8611	0.7223
Beam = 3	0.8959	**0.7236**
Beam = 5	**0.8984**	0.7223
Beam = 10	0.8887	0.7225
Beam = 20	0.8884	0.7194

Q2-1: Metrics (Greedy, steps = 50)

Settings	CIDEr Score	CLIP Score
Adapter, embedding size = 64	0.7980	0.7149
Adapter, embedding size = 128	0.8009	0.7143
Prefix, prefix length=32	0.8251	0.7162
Prefix, prefix length=64	0.8344	0.7207
LoRA, rank=4	0.8411	0.7165
LoRA, rank=8	**0.8611**	**0.7223**

Q2-1: Metrics (Beam Search, beam = 5, steps = 30)

Settings	CIDEr Score	CLIP Score
Adapter, embedding size = 64	0.8554	0.7113
Adapter, embedding size = 128	0.8336	0.7141
Prefix, prefix length=32	0.8969	0.7152
Prefix, prefix length=64	**0.9089**	0.7200
LoRA, rank=4	0.8990	0.7153
LoRA, rank=8	0.8984	**0.7223**

Q2-1: Trainable Parameters

Settings	Trainable Parameters
Cross Attention + Encoder Projection Only	29.13M
Adapter, embedding size = 64	31.51M
Adapter, embedding size = 128	33.88M
Prefix, prefix length=32	29.16M
Prefix, prefix length=64	29.18M
LoRA, rank=4	29.73M
LoRA, rank=8	30.32M

Q2-1: Best Setting

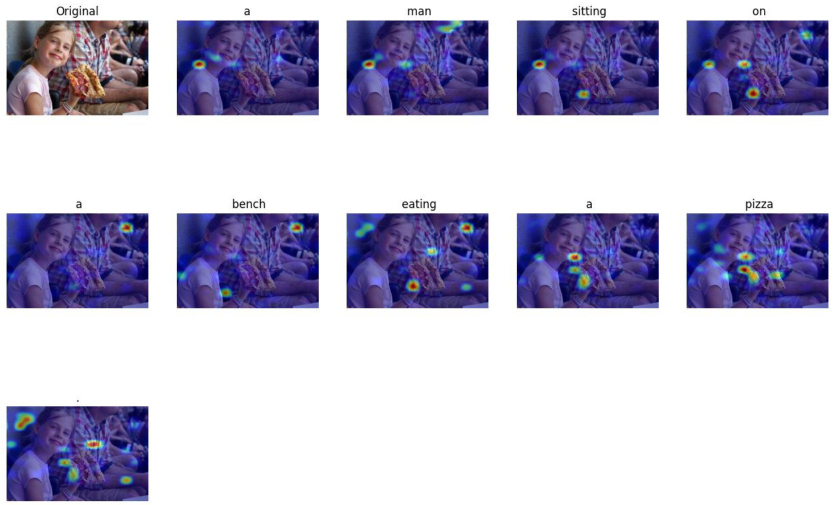
Settings	Trainable Patamerters	CIDEr Score	CLIP Score
LoRA, rank=32 Beam search, beam=5 Decoding steps = 30	33.85M	0.9225	0.7194

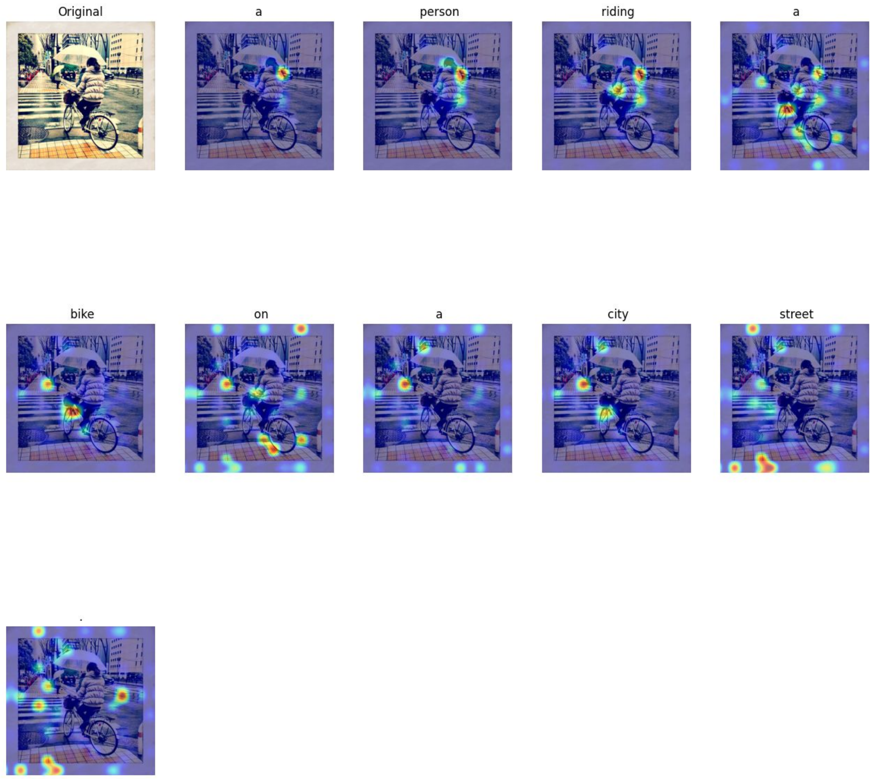
Q2-2: Attention Maps (Settings)

For simplicity, I use greedy search as decoding method.

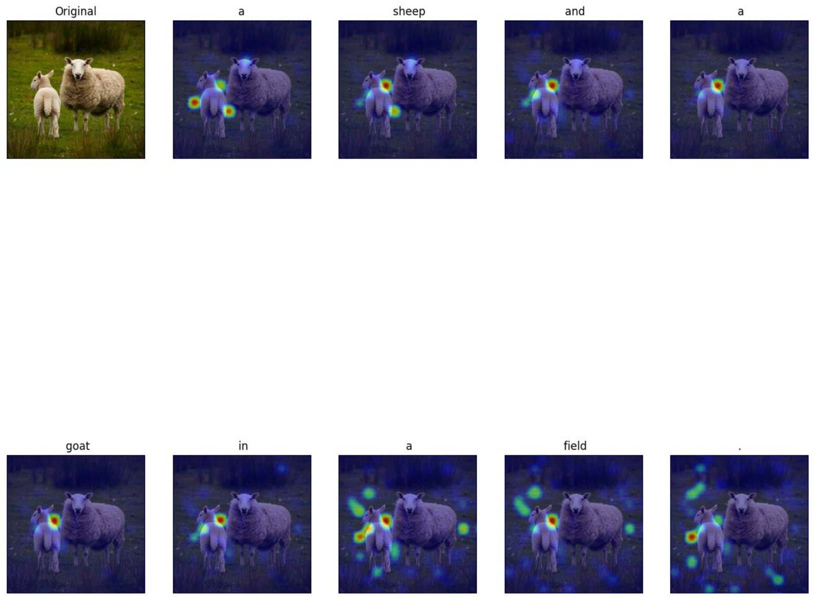
Also, I extract attention scores in cross attention from all the middle 6 transformer blocks and sum it up to visualize attention maps. (I've tried different settings and I think this is the best result I get)

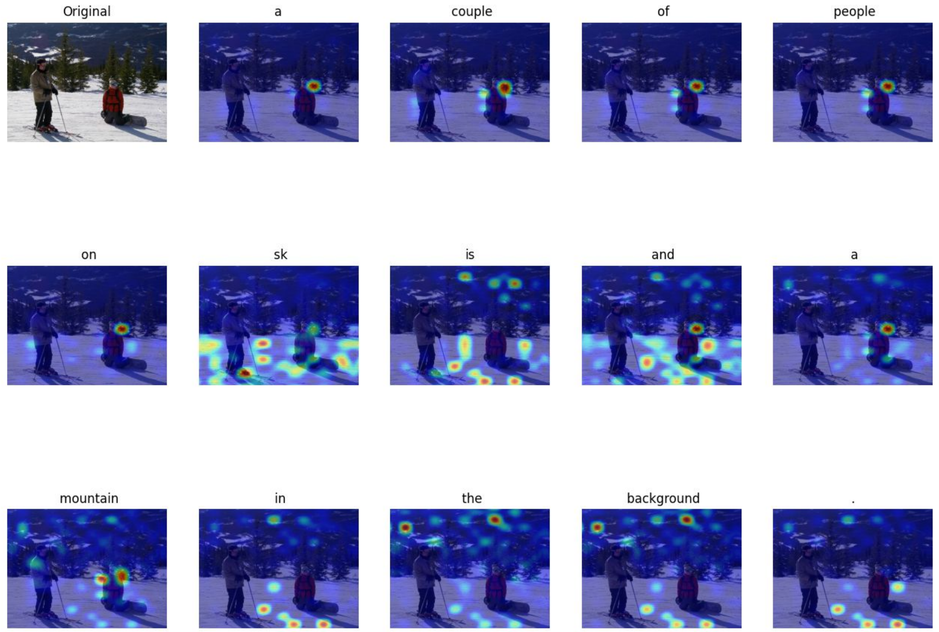
Q2-2: Attention Maps



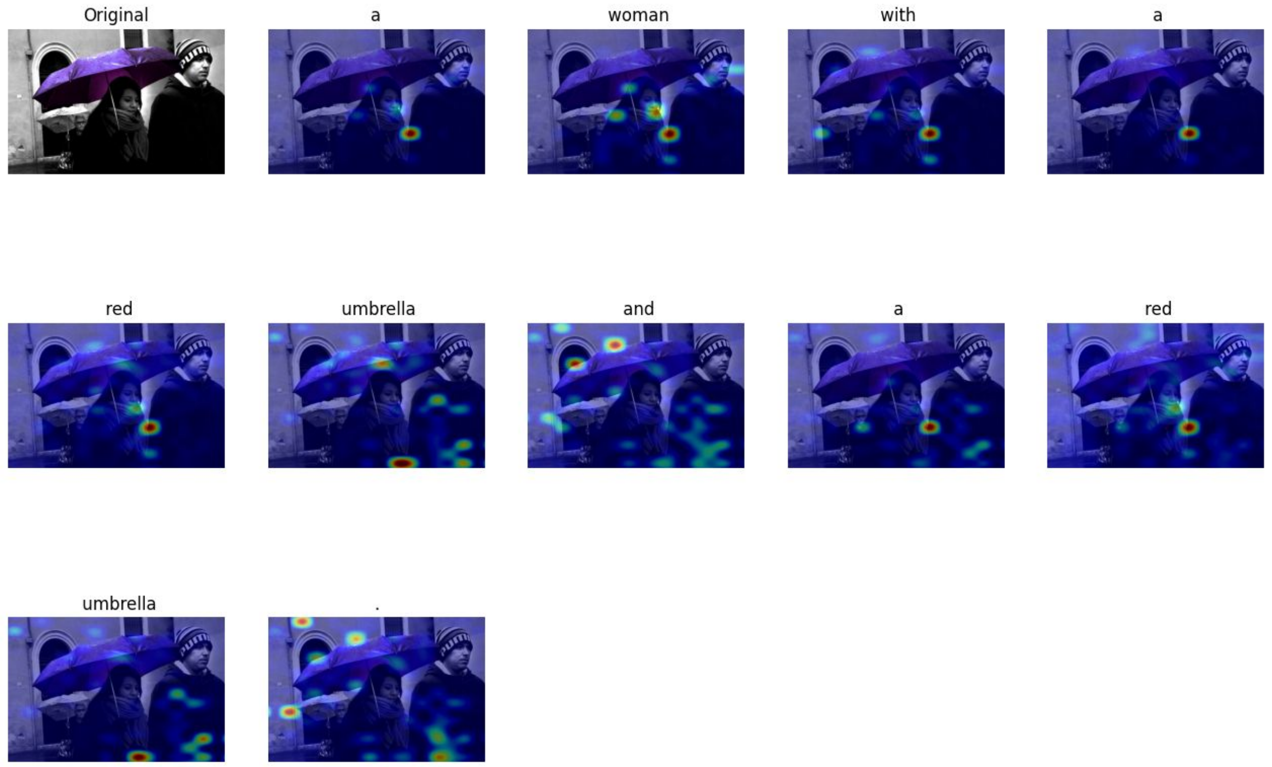


Q2-2: Attention Maps





Q2-2: Attention Maps



Q2-2: Attention Maps (Discussion)

From the previous examples, we can see that in some of the words, the model attends to the correct part in the image, such as ‘pizza’, ‘person’, ‘riding’, ‘bike’, ‘sheep’, ‘skis’, ‘background’, ‘woman’ in five different images. That means the model captured the correct semantic of the images.

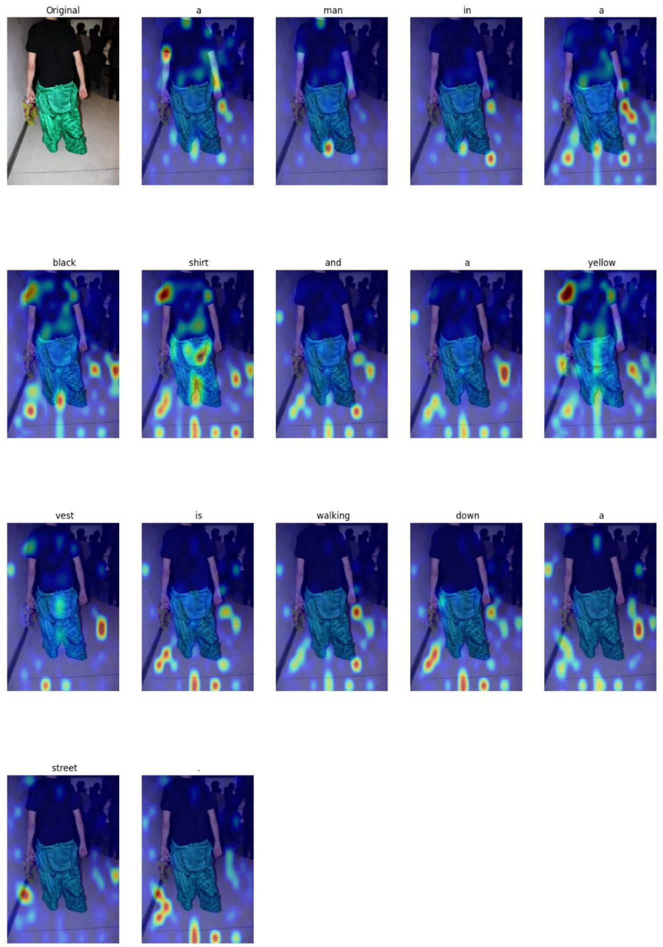
However, not all the words attend to the correct part. For example, in the word ‘mountain’ and ‘umbrella’, the model attends to the incorrect parts (tree and coat in these two example respectively) of the images. But I think it does not affect the model output dramatically.

Q2-2: Attention Maps

Lowest CLIP Score: 0.3436

For most of the word the model attends to the floor, while the output words are not related to ‘floor’.

Also, for the word ‘black’, ‘shirt’ and ‘yellow’ the model attends to somewhere that’s not related to the word. So I think this might be the reason why this example failed.

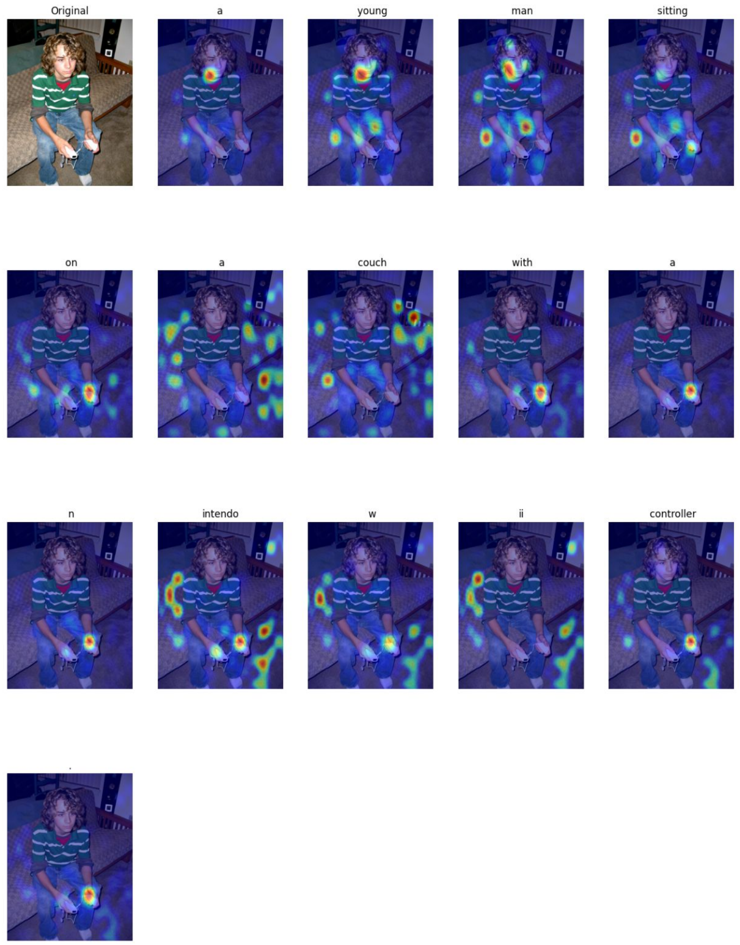


Q2-2: Attention Maps

Highest CLIP Score: 0.9997

We can see that in the word ‘young’ and ‘man’, the model attends to the person’s face. For the word ‘couch’ the model attends to the couch in the background.

As for the word ‘nitendo’ and ‘controller’, the model attends to the controller in the person’s hand. So I think this might be the reason why this example worked well.



Reference

Q1: https://github.com/openai/CLIP

Q2: https://github.com/huggingface/pytorch-image-models

Notes

- READ MORE DOCUMENTS!
- The shape of cross attention can be different (q, k, v)
- PEFT can be added only in last few layers
- Use downstream metrics to choose model (instead of validation loss)
- Inference Process in Autoregressive Transformer Architecture
- Cross Attention Explanation