HW3 VL Models

GINM11 R11944004 李勝維

Problem 1 CLIP

1. Methods analysis

Q: Please explain why CLIP could achieve competitive zero-shot performance on a great variety of image classification datasets.

A: CLIP is trained on a large number of images and their caption, and CLIP's pretraining objective is maximizing the cosine similarity between the paired image text samples while minimizing the unpaired ones.

This approach has a fundamental difference compared to previous methods: by **not** directly optimizing the benchmark, CLIP is trained under natural language supervision. CLIP can "narrow" the gap between tasks and learn a more generalized representation since natural language is a high-level representation of human understanding of the world.

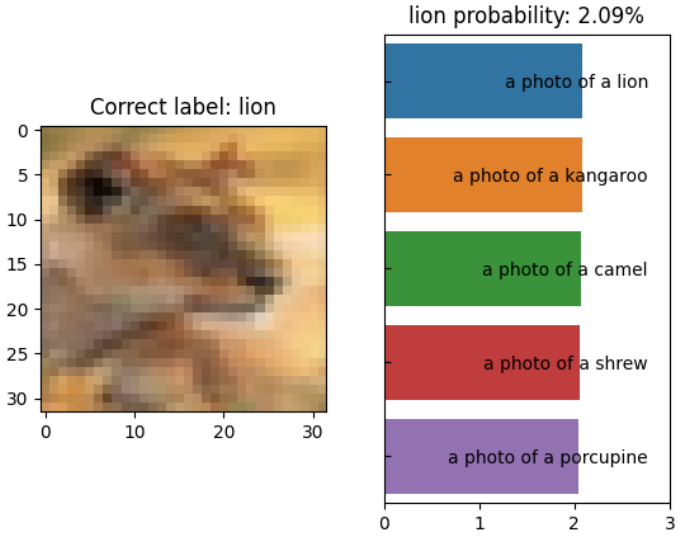
More formally, we can easily construct a linear classifier by "telling" CLIP's text encoder "This is a photo of {object}" without any labeled data, then its powerful visual representation can conduct the task with competitive performance.

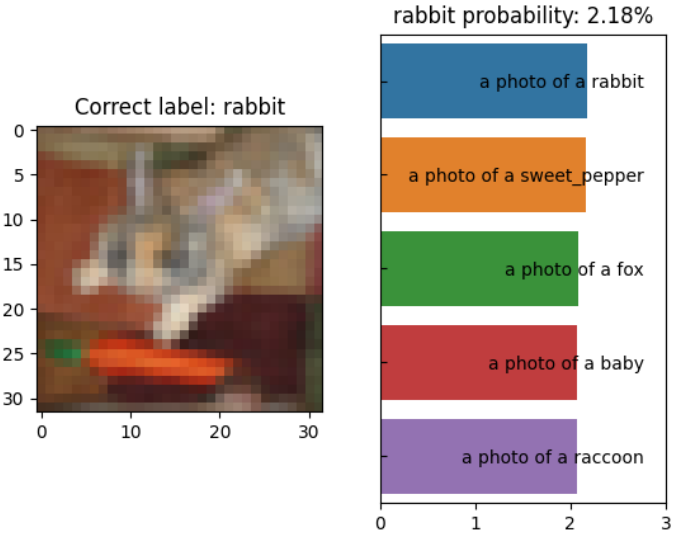
2. Prompt-text analysis

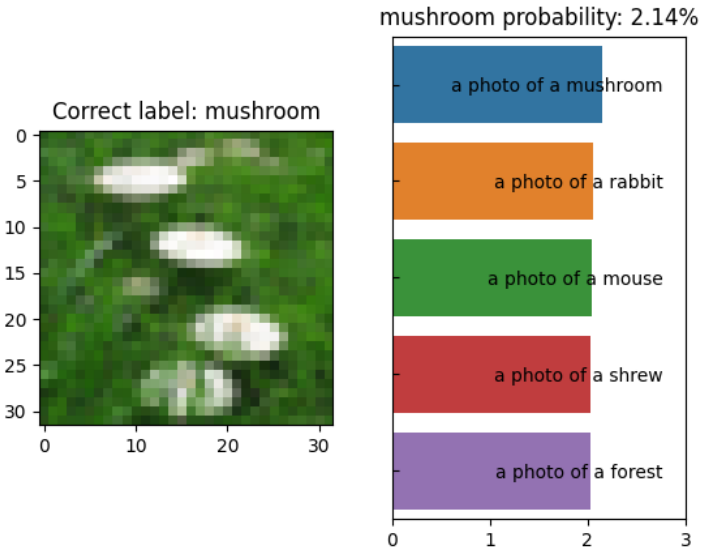
The CLIP configuration that I use is "ViT-L-14"

Prompt	Validation Accuracy
This is a photo of {object}	67.36%
This is a {object} image.	**72.84%**
No {object}, no score.	45.52%

3. Quantitative analysis







Problem 2 Image Captioning

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

Model settings:

- "Pretrained encoder": "deit3_large_patch16_224_in21ft1k",
- "num_layers": 4,
- "nhead": 8,
- "hidden_size": 1024,
- "activation": "gelu",
- "dropout": 0.1

Training settings:

- Optimizer: AdamW
- Epochs: 5
- Lr: 5e-5
- Batch size: 32
- Linear warmup for 1000 steps
- Cosine annealing after warm-up
- Auto mixed precision training with bfloat16

Decoding settings:

- Beam search with beam size = 3 and max length = 30

Scores:

- CIDEr: 0.9412714225585563
- CLIP: 0.7310674335784656

I use the deit v3 (Touvron et al. 2022) large model released by Meta as my image transformer.

The deit v3 is pretrained on ImageNet22K and intermediate finetuned on ImageNet1K.

2. Report the other 3 different attempts and their corresponding CIDEr & CLIPScore.

Hyperparameters:

(Greedy decoding + default value as previous page)

CLIP score	CIDEr score	pretrained encoder	num_layers	nhead	batch_size
7.19E-01	7.87E-01	vit_large_patch16_224	6	8	32
7.14E-01	8.15E-01	vit_large_patch16_224	8	16	32
7.18E-01	8.27E-01	vit_large_patch16_224	6	16	32
7.25E-01	8.66E-01	deit3_large_patch16_224_in21ft1k	8	16	32
7.32E-01	**8.93E-01**	deit3_large_patch16_224_in21ft1k	4	8	8
7.30E-01	9.01E-01	deit3_large_patch16_224_in21ft1k	8	8	32

Decoding Strategies(Beam search)

#Beams	CIDEr Score	CLIP Score
Greedy	0.8932304083679930	0.7320361833055300
3	0.9412714225585563	0.7310674335784656
4	0.9447991262307799	0.7304935871122136
5	0.9408538464367473	0.7290712530350405
6	0.9468843814599852	0.7277360876690016
7	0.9477172865308948	0.7275496387428382
8	0.9498816789336173	0.7275578268018097
9	0.9492018213553265	0.7275632855077907
10	0.9453278783705885	0.7267187213417936
15	0.9525480681625157	0.7270469260389010

100	0.9538702565136582	0.725669796870633

Problem 3 Attention Visualization

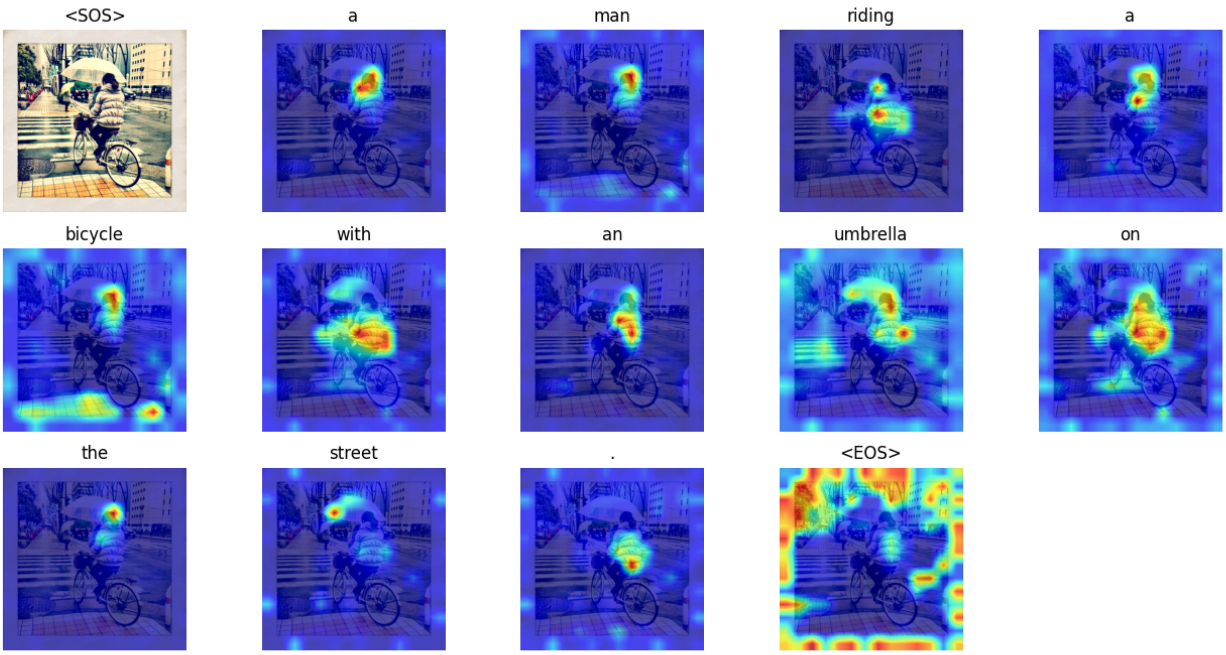
Note: For my visualization code to work, you need to modify PyTorch's source code, line 597 of “torch/nn/modules/transformer.py.” Modify “need_weights=False” to “need_weights=True” to return attention scores

Also, I use greedy decoding here to simplify the problem.

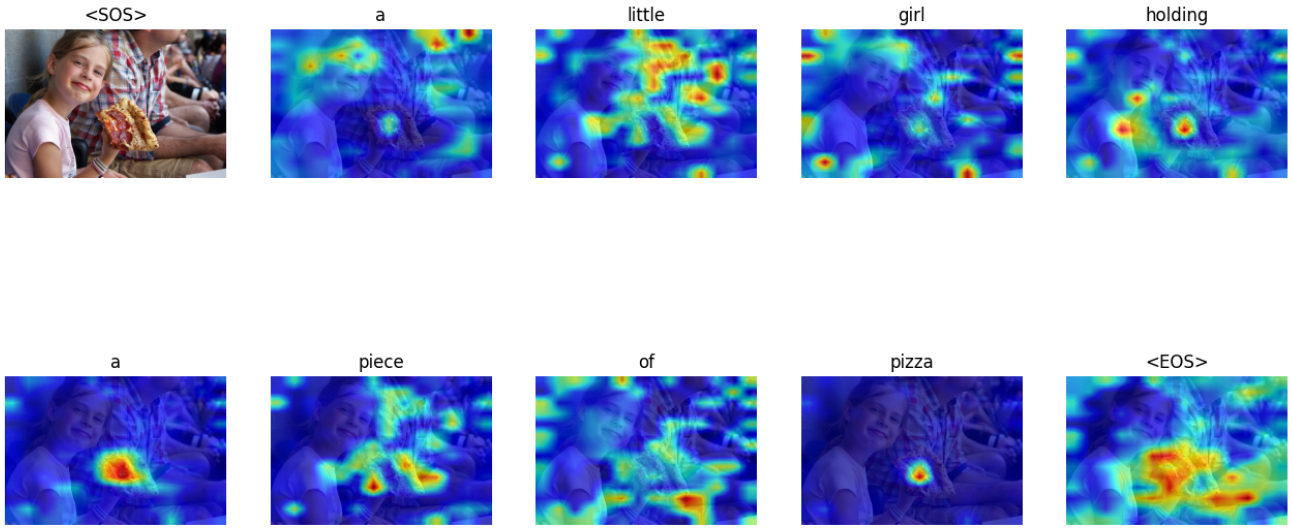
```
# multihead attention block
def _mha_block(self, x: Tensor, mem: Tensor,
               attn_mask: Optional[Tensor], key_padding_mask: Optional[Tensor]) -> Tensor:
    x = self.multihead_attn(x, mem, mem,
                            attn_mask=attn_mask,
                            key_padding_mask=key_padding_mask,
                            need_weights=True)[0]
    return self.dropout2(x)
```

1. Visualize the predicted caption and the corresponding series of attention maps

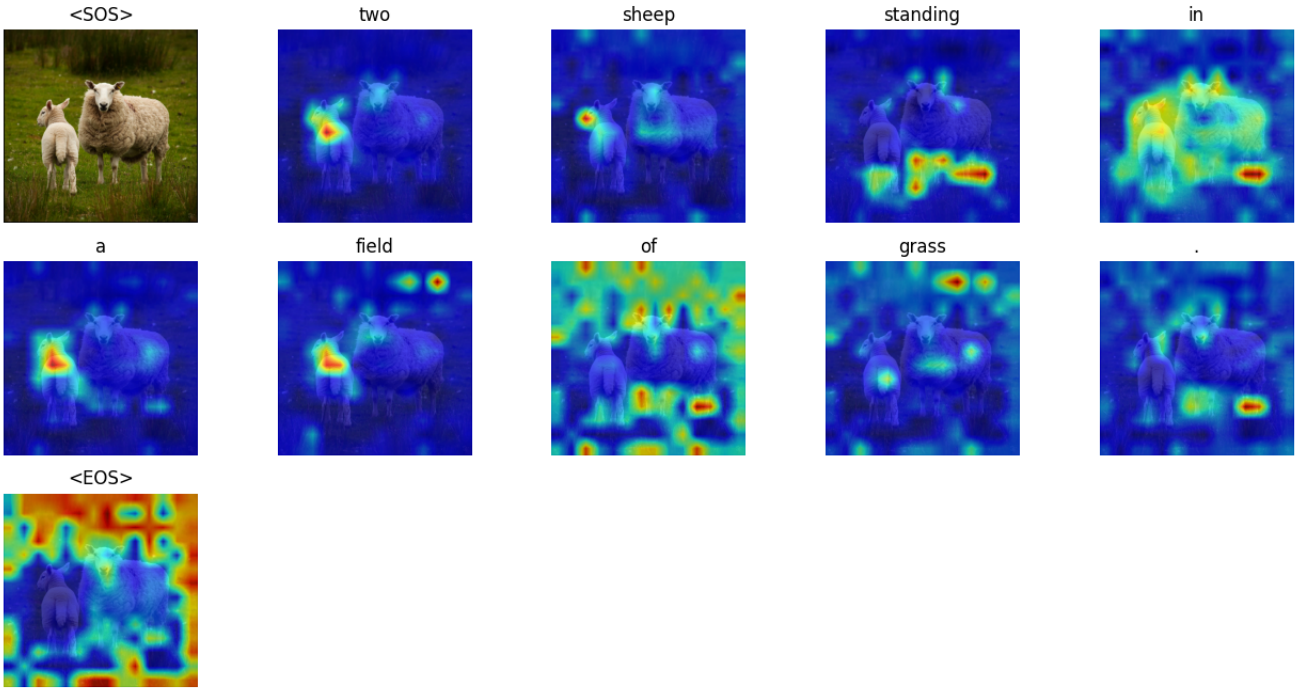
Bike:



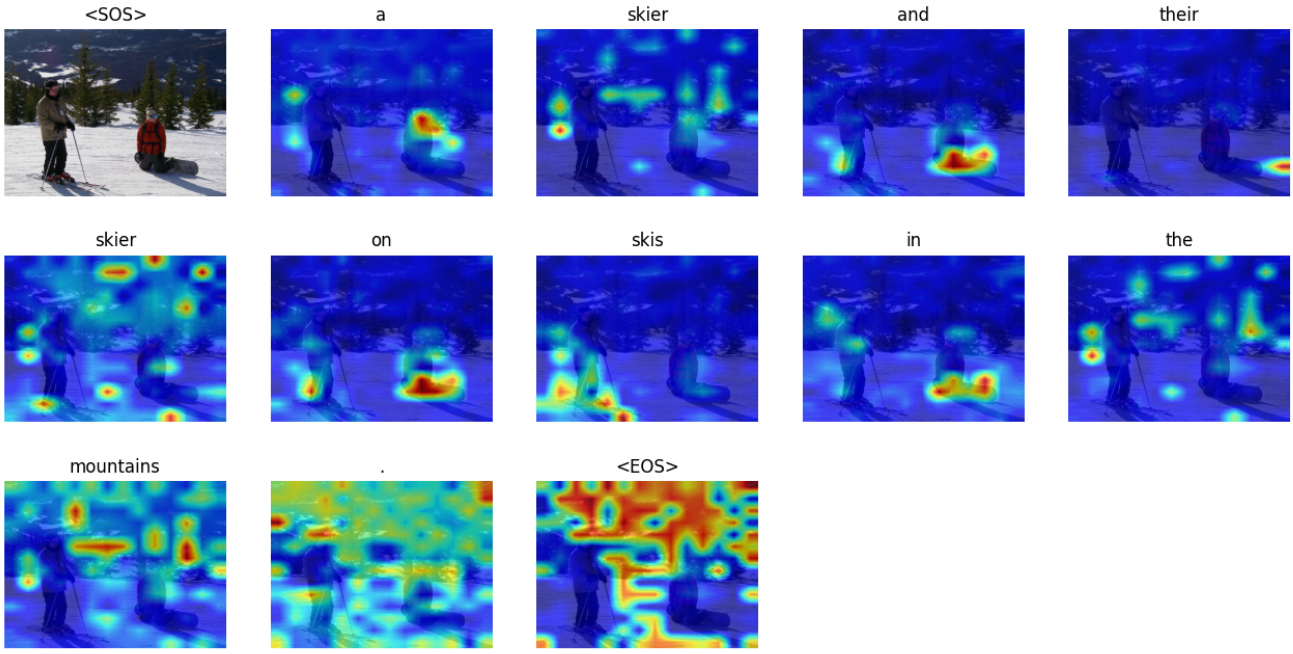
Girl:


<SOS>
a
little
girl
holding
a
piece
of
pizza
<EOS>

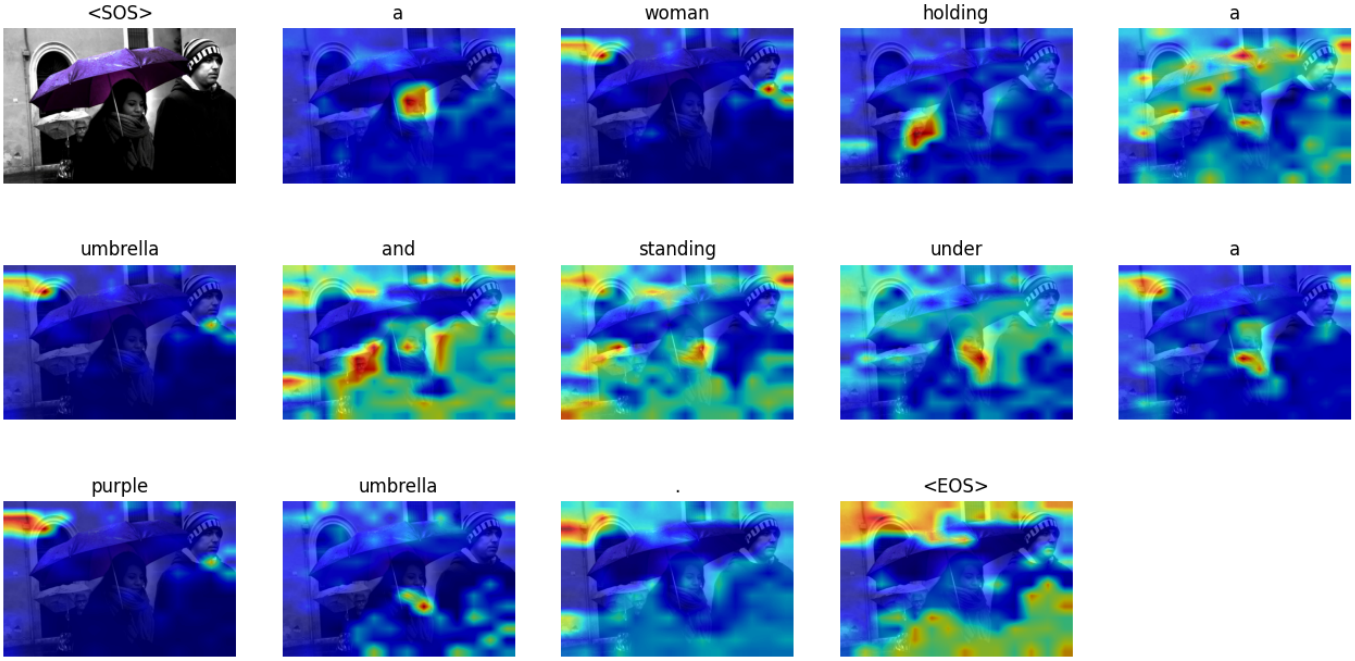
Sheep:


<SOS>
two
sheep
standing
in
a
field
of
grass
.
<EOS>

Ski:

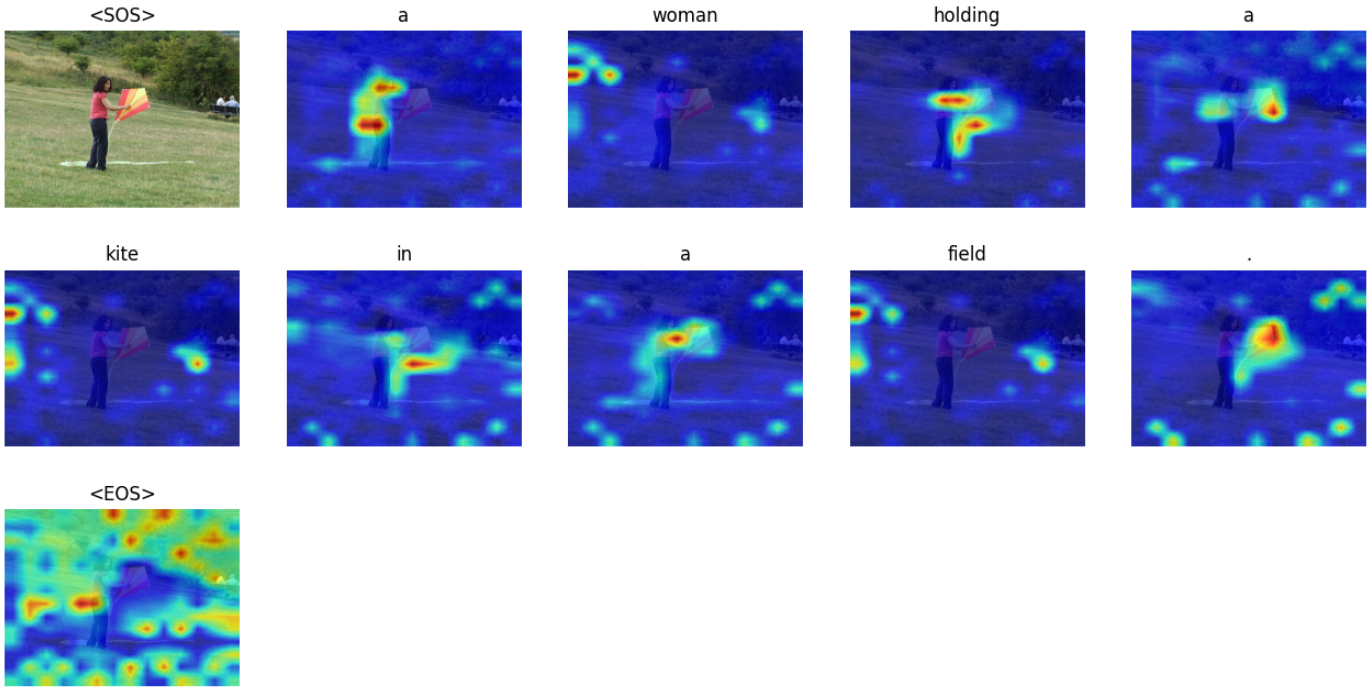


Umbrella:

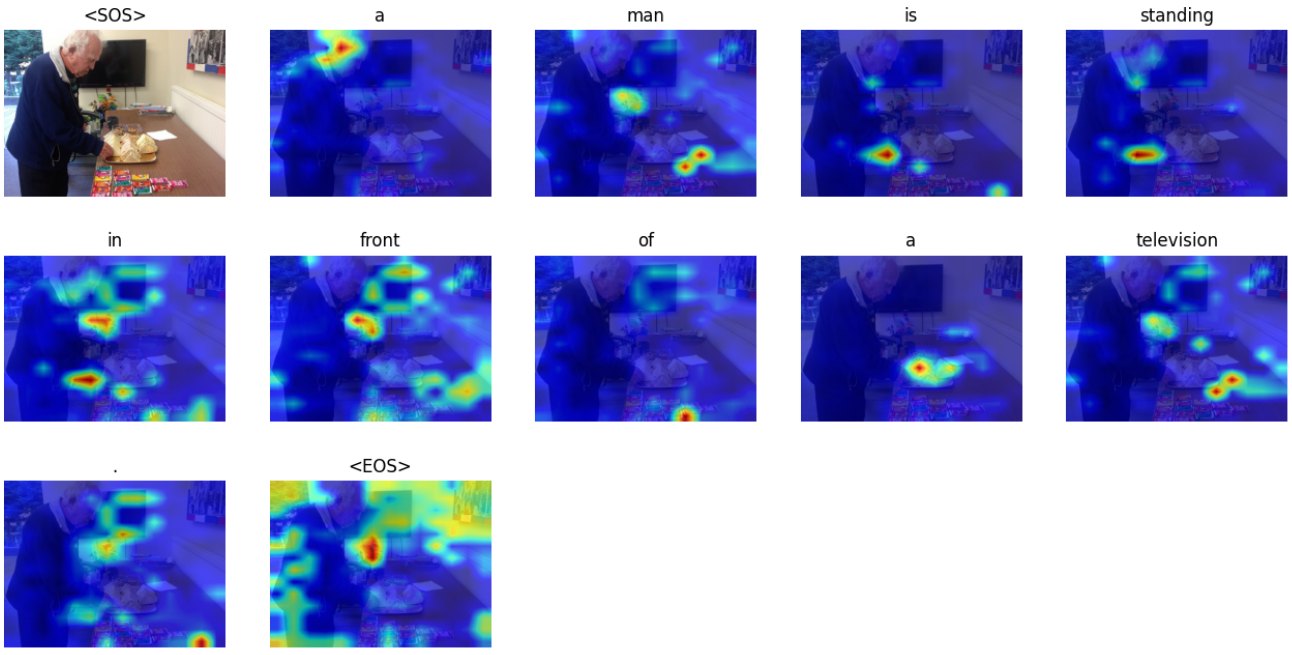


2. According to **CLIPScore**, you need to visualize:
    i. top-1 and last-1 image-caption pairs
    ii. its corresponding CLIPScore

Top-1: 000000179758.jpg, CLIP score = 0.9765625



Last-1: 000000204777.jpg, CLIP score = 0.3839111328125



3. Analyze the predicted captions and the attention maps for each word according to the previous question.
    a. Is the caption reasonable?

In most cases, the caption describes the image fairly well, except for “bike.jpg,” “ski.jpg” and “000000204777.jpg.” The model misidentifies the woman as a man on “bike.jpg.”

For “ski.jpg,” the model fails to produce a sentence with the correct structure, but it produces somewhat related words such as “skier,” “skis,” and “mountains.” I think beam search helps solve this issue.

As for “000000204777.jpg,” the model misunderstood the old man’s direction: instead of describing the old man’s position with the table, the model describes the position with the television.

In conclusion, the model performs very well in my opinion, it only makes some forgivable mistakes

    b. Does the attended region reflect the corresponding word in the caption?

Yes.

- Bike: the model attends on person/umbrella/street while generating the corresponding words.
- Girl: the model attends to the girl(hairstyle)/pizza/holding(arm) while generating the corresponding words.
- Sheep: the model attends to the sheeps/standing(leg)/grass while generating the corresponding words.
- Umbrella: this is an interesting one, the model attends to the part of the next word described; it attends to the umbrella when generating the second “a”; it attends to the women while generating the first “a” and so on. But I think the attention results are reasonable.

The model also attends the whole picture while generating <EOS>, I think it checks if every object in this image has been generated or not before generating <EOS> token.