Efficient Image Clustering Conditioned on Text Criteria

Sheikh Shafayat

Recap on my previous paper presentation…

IMAGE CLUSTERING CONDITIONED ON TEXT CRITERIA

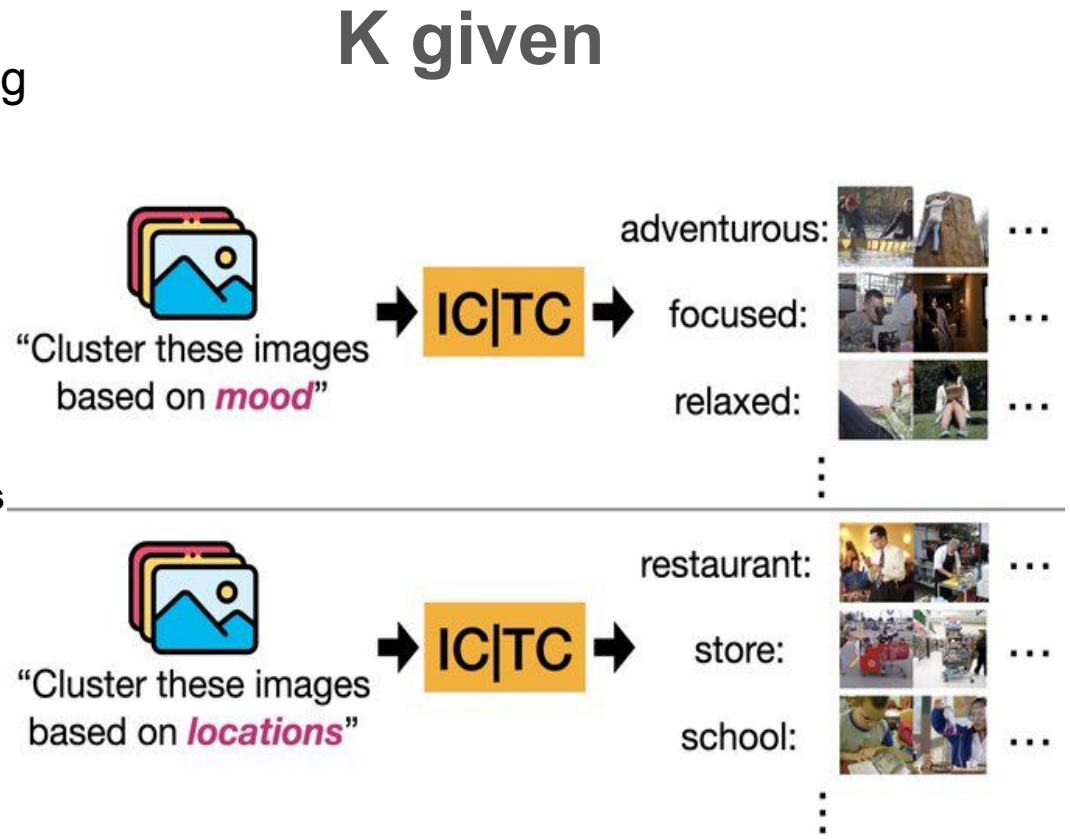
Sehyun Kwon[†][1], **Jaeseung Park**[†][1], **Minkyu Kim**[◇], **Jaewoong Cho**[◇], **Ernest K. Ryu**[†*], **Kangwook Lee**[◇♣*]
[†]Seoul National University, [◇]KRAFTON, [♣]University of Wisconsin–Madison, [*] Co-senior authors

ABSTRACT

Classical clustering methods do not provide users with direct control of the clustering results, and the clustering results may not be consistent with the relevant criterion that a user has in mind. In this work, we present a new methodology for performing image clustering based on user-specified text criteria by leveraging modern vision-language models and large language models. We call our method **I**mage **C**lustering Conditioned on **T**ext **C**riteria (IC|TC), and it represents a different paradigm of image clustering. IC|TC requires a minimal and practical degree of human intervention and grants the user significant control over the clustering results in return. Our experiments show that IC|TC can effectively cluster images with various criteria, such as human action, physical location, or the person's mood, while significantly outperforming baselines.[2]

What is the problem?

- They are doing image clustering
- Not just any kind of clustering
  - Clustering based on user query
  - Query is **word** based
- Use cases:
  - You can cluster **the same images in many different ways**
  - By mood, location, event



More example…

How does it work?

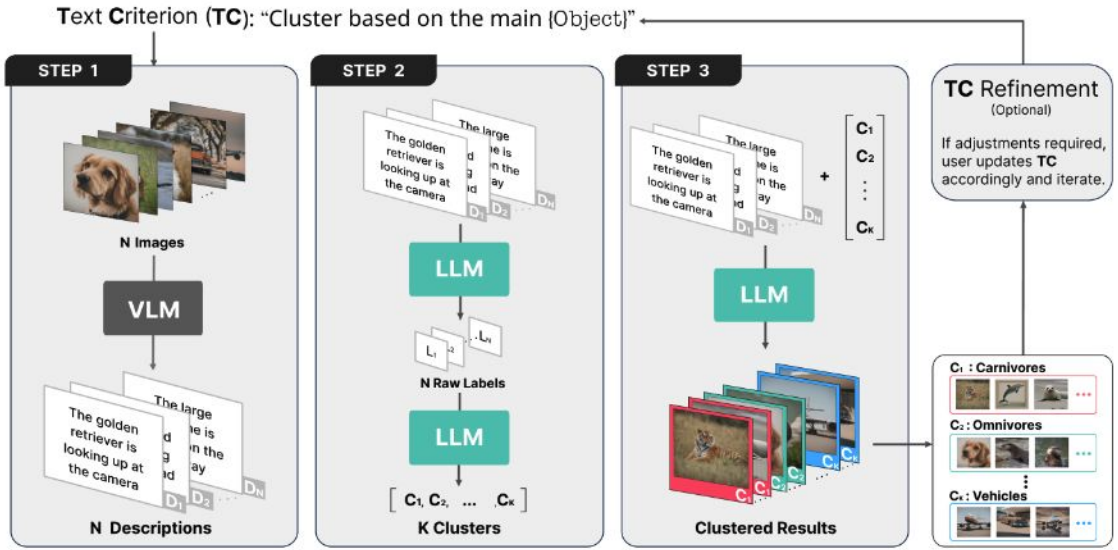


Figure 2: The IC|TC method. (Step 1) Vision-language model (VLM) extracts detailed relevant textual descriptions of images. (Step 2) Large language model (LLM) identifies the names of the clusters. (Step 3) LLM conducts clustering by assigning each description to the appropriate cluster. The entire procedure is guided by a user-specified text criterion (**TC**). (Optional **TC** Refinement). The user can update the text criterion if the clustering results are unsatisfactory. See Appendix B.4 for an unabridged sample output.

But there were some problems…

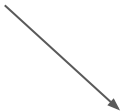
Cons about the paper… 🧐

- Computationally **VERY expensive**
- Need to run every step for every query
- For **every query**:
  - Caption all images in the database using VLM
  - Cluster those captions using LLM
  - Put each image to corresponding cluster using LLMs

The Question of this Project was?

Notice the three steps:

- For **every query**:
  - Caption all images in the database using VLM
  - Cluster those captions using LLM
  - Put each image to corresponding cluster using LLMs



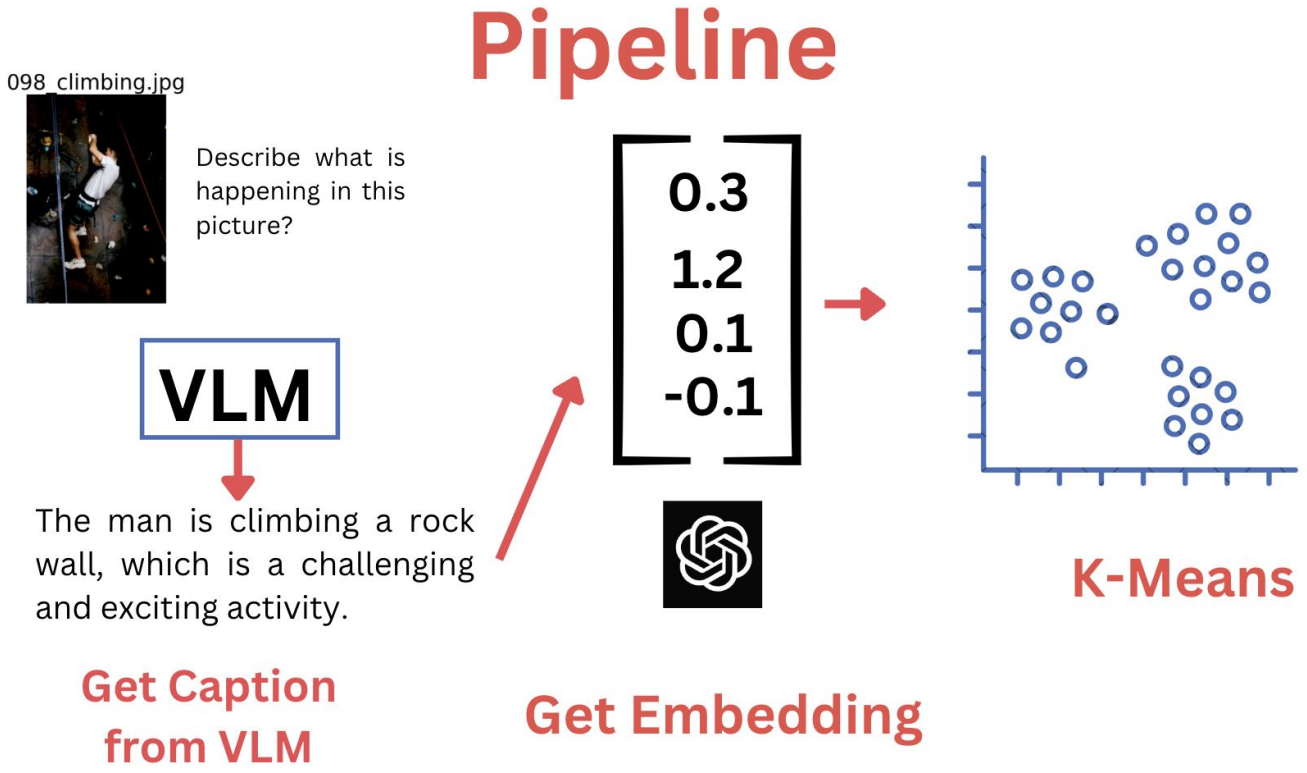
Can we replace these LLM calls?

How Did I Approach?

What was my approach?

- Simple:
  - Generating caption like before
  - Then get text embedding of the caption
  - Perform embedding clustering
    - Using K-Means clustering algorithm

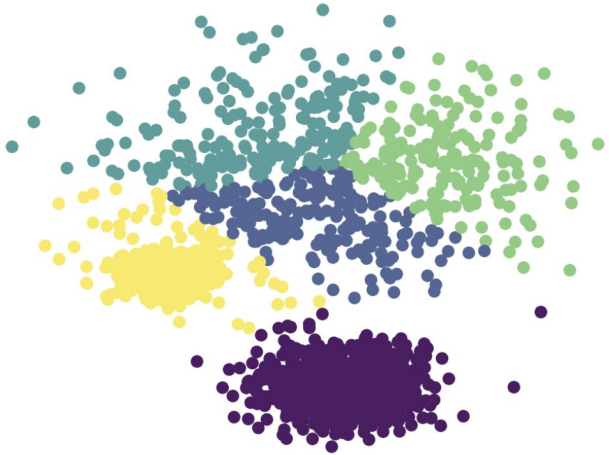
What was my approach?



But, does it actually work?

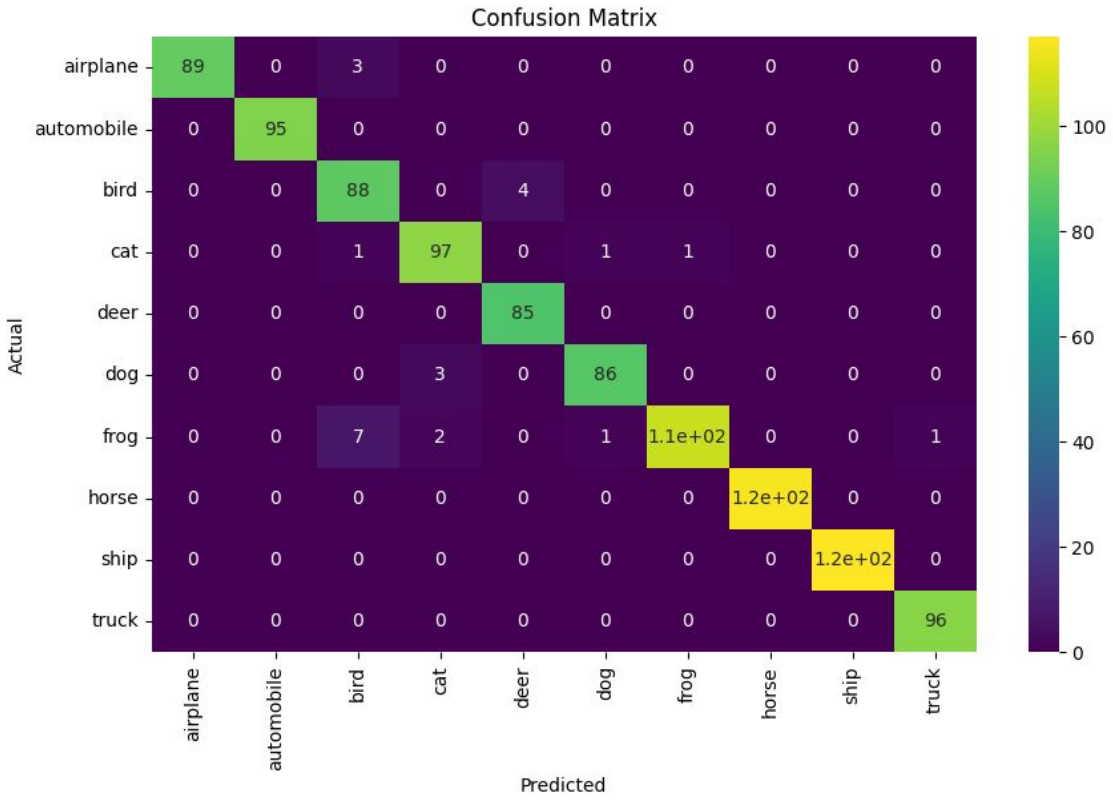
It is not obvious why it should work

- So, first experiment with CIFAR-10
  - Cluster CIFAR-10 test set
    - Majority label of each cluster is the label for all the images in that cluster



Results?

- Pretty Good!:
  - Accuracy: 0.97
  - F1: 0.976
  - Precision: 0.977
  - Recall: 0.976
- Comparable with 0.987 (acc) reported in the paper



Note, number of samples are not same across classes. I was working with 1000 samples

That was proof of concept

Let’s solve the real problem

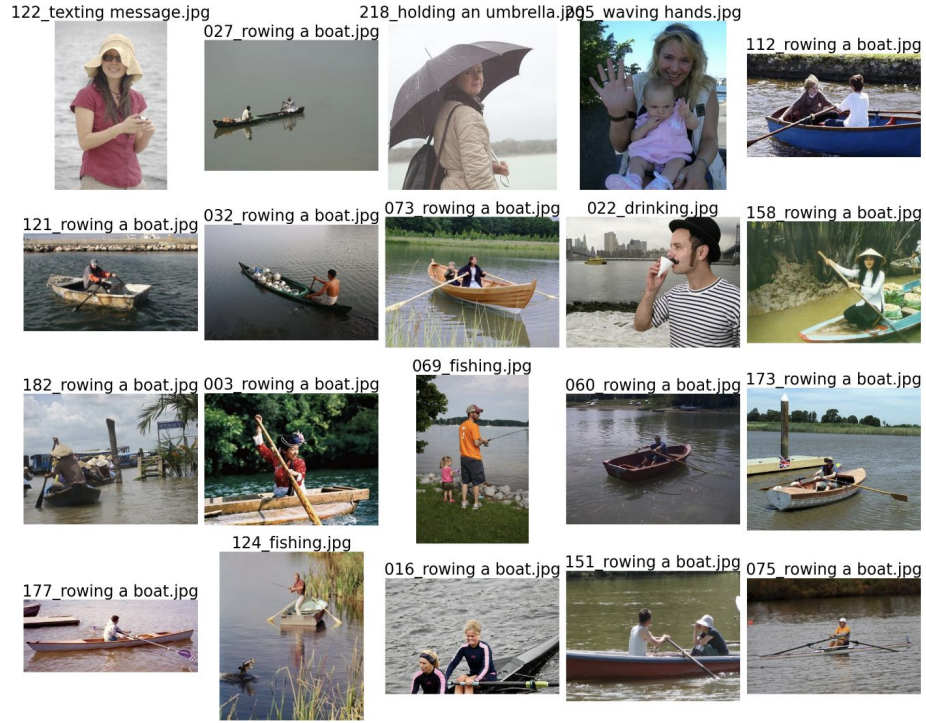
Stanford 40-Actions Dataset

- There are 40 different actions people are doing
- We need to classify them
- Authors also relabel 1000 data
  - Mood
  - Location

How does my clustering method do?

The results are interesting…

```
Cluster 0:
['The person is located near a body of water, possibly a lake or a river.', 'The person is located in a boat, float
['texting message', 'rowing a boat', 'holding an umbrella', 'waving hands', 'rowing a boat', 'rowing a boat', 'rowi
gpt4 summary: boating
```



The results are interesting…

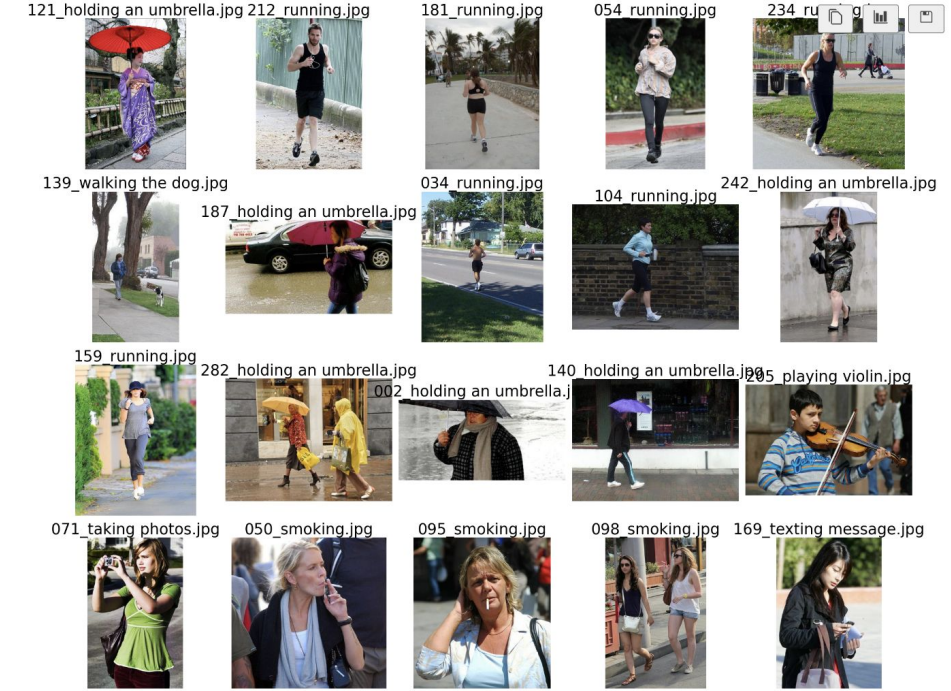
```
Cluster 2:
['The person is located in a field, standing next to a horse.', 'The person is located in a field, standing next to a
['shooting an arrow', 'feeding a horse', 'feeding a horse', 'feeding a horse', 'feeding a horse', 'feeding a horse',
gpt4 summary: with horse
```

The results are interesting…

```
Cluster 2:
['The person is located in a field, standing next to a horse.', 'The person is located in a field, standing next to a
['shooting an arrow', 'feeding a horse', 'feeding a horse', 'feeding a horse', 'feeding a horse', 'feeding a horse',
gpt4 summary: with horse
```

The results are interesting…

```
Cluster 4:
['The person is located on a sidewalk, walking down a street.', 'The person is located on a sidewalk, walking down
['holding an umbrella', 'running', 'running', 'running', 'running', 'walking the dog', 'holding an umbrella', 'runn
gpt4 summary: walking
```



The results are interesting…

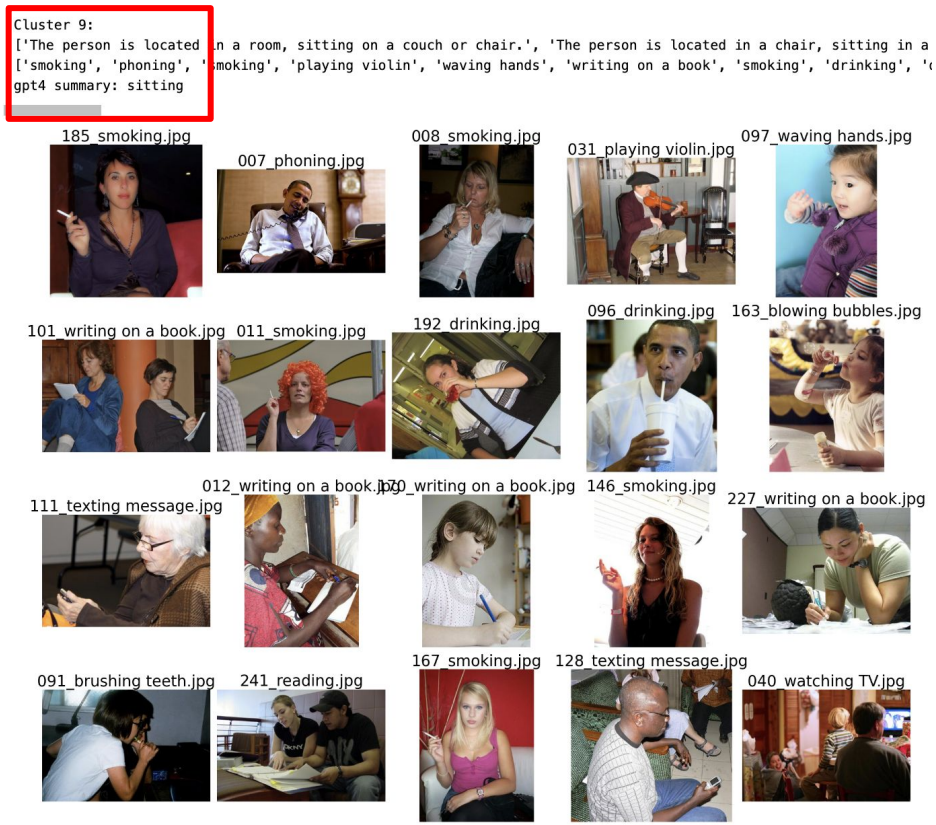
```
Cluster 2:
['The person is located in a field, standing next to a target and holding a bow and arrow.', 'The person is located
['shooting an arrow', 'shooting an arrow', 'shooting an arrow', 'shooting an arrow', 'shooting an arrow', 'shooting
gpt4 summary: Holding bow
```

The results are interesting…

- Sometimes **it works**, sometimes it fails
- We got the labels by asking GPT4 [denoted *gpt4 summary*]
  - Taking the 20 images closest to each centroid
  - Summarizing their captions

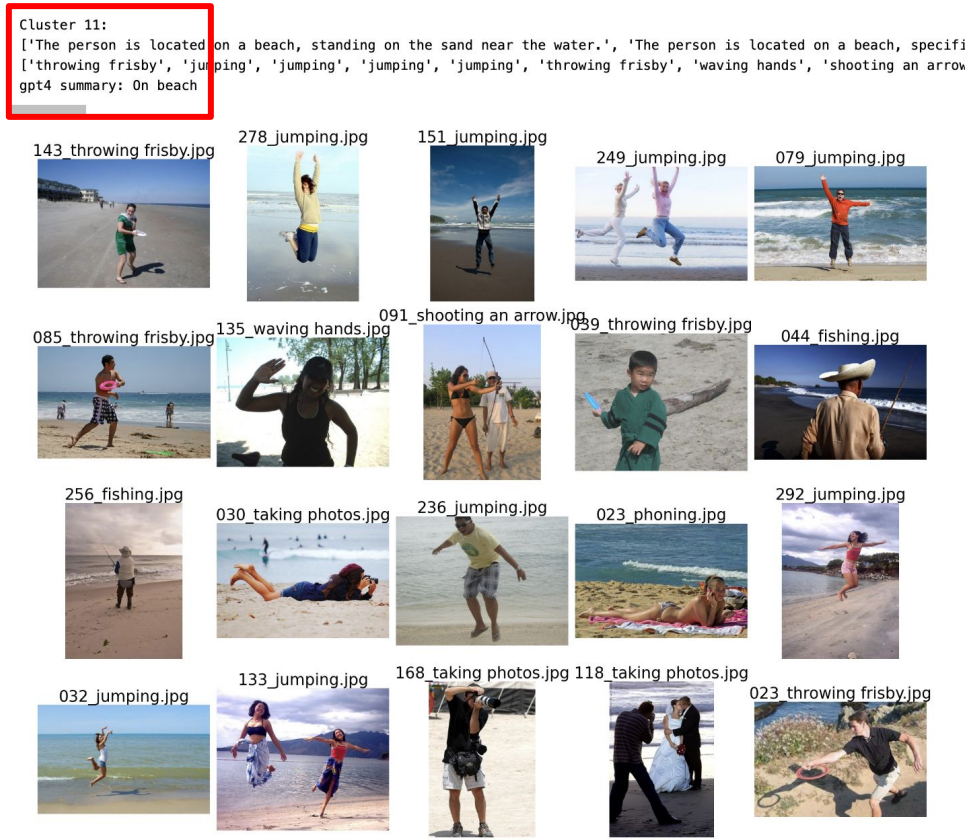
Note that…

- Our label does not always correspond to the label given the dataset
- There is no “sitting” cluster in the dataset
- But “sitting” cluster is just as valid



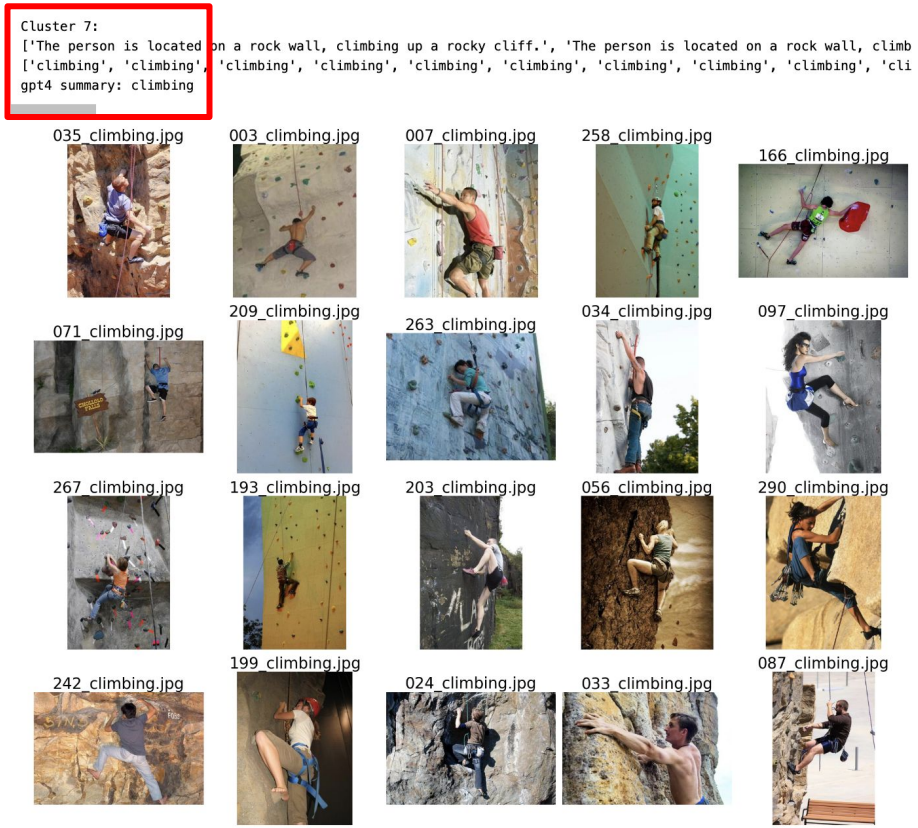
Note that…

- The label is “on beach”
- Which is kind of right



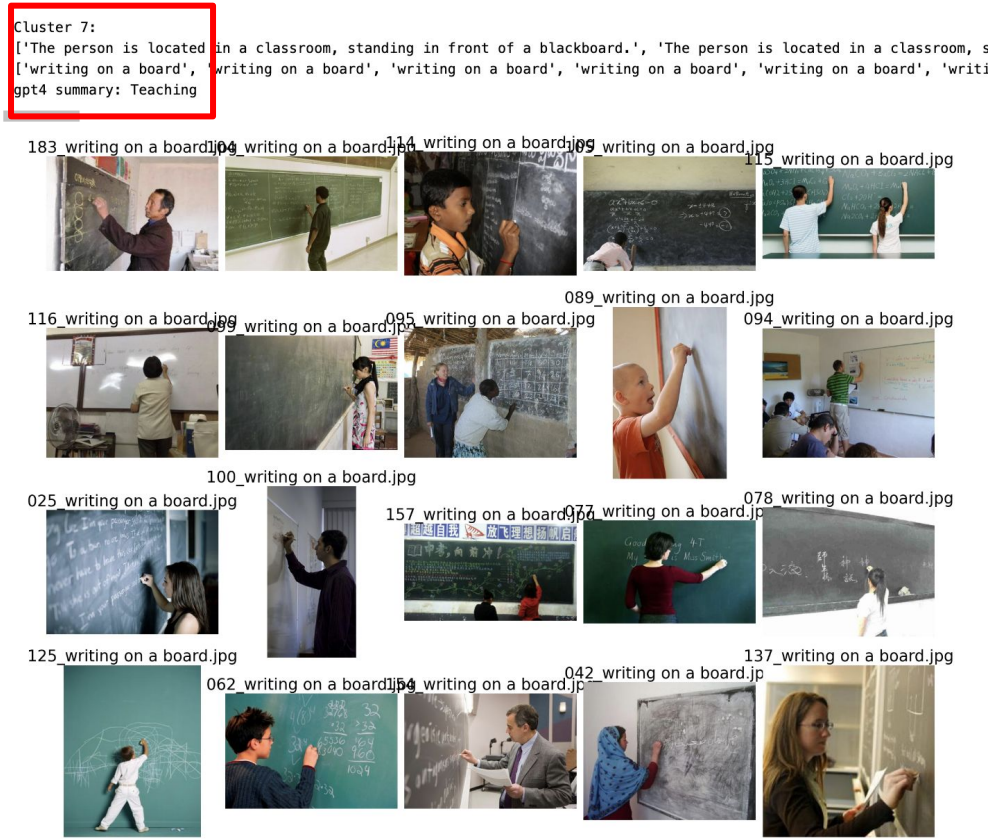
Note that…

- Sometimes the cluster we get is same as the given dataset
- “Climbing” is indeed a category in the dataset



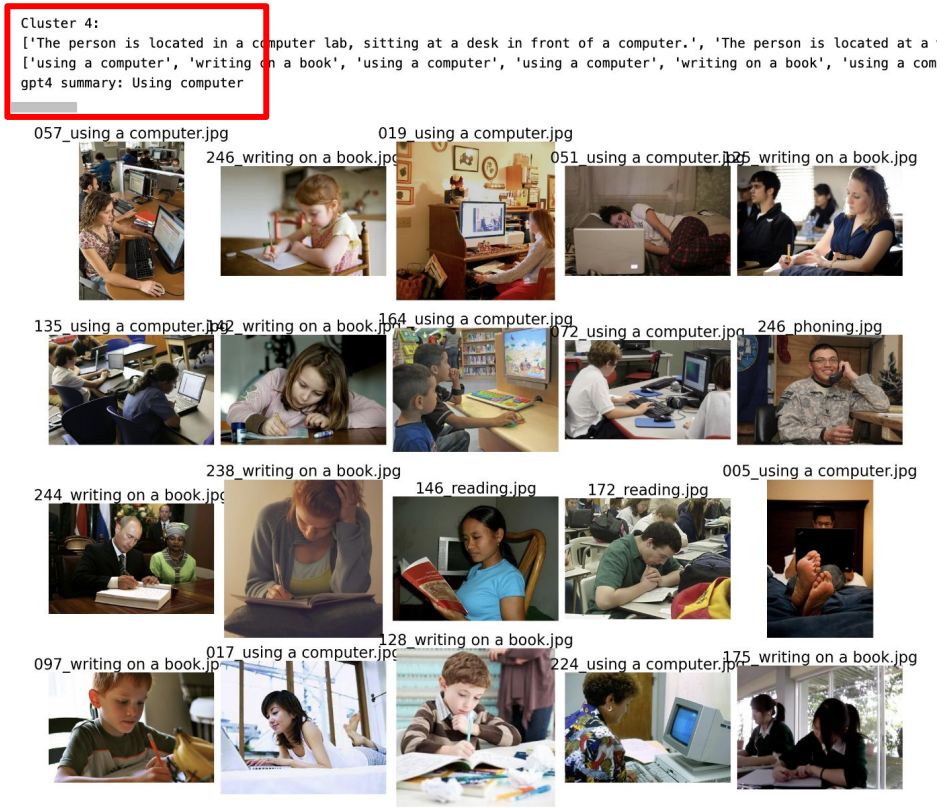
Note that…

- Sometimes our cluster name is slightly different



Note that

- Sometimes the clustering is not very good



Let’s see the numbers

My implementation vs paper's expensive implementation

Dataset	Criterion	SCAN	Ours
Stanford 40 Action	Action	0.397	**0.774**
	Location	0.359*	**0.822***
	Mood	0.250*	**0.793***

Mine
0.580465
0.741
0.5461

My implementation:

- Much faster (~ 1 min max)
  - No complex prompting
- But does not reach as high score

Note that, the numbers are not really comparable:

- Their cluster membership inference requires several more LLM/nltk calls
- While mine does not

Why it doesn’t reach as high score?

- Sometimes the caption model fails
  - It was LLaVa model
  - Happens often
- Got classified as “planting flowers”

```
The person, a young boy, is located
in a garden, standing next to a
bush
```

Why it doesn’t reach as high score?

- Sometimes the caption model fails
  - It was LLaVa model
  - Happens often
- Got classified as “planting flowers”

```
The person, a young boy, is located
in a garden, standing next to a
bush
```

Why it doesn’t reach as high score?

- Sometimes it is not really a “mistake”
- It got classified as “cooking”

We can control K, right?

What happens if you control K?

- K = 2
  - Only “Standing” and “Sitting”
- K = 5
  - “Standing”, “Sitting”, “Working”, “Climbing”, “Walking”
- K = 10
  - 'Standing', 'Playing guitar', 'Washing dishes', 'Standing in a field', 'Walking', 'Sitting', 'Climbing', 'Boating', 'Next to car', 'Positioned' (?!)

The larger K, the more fine-grained the clustering

What’s Next?

One easy way might improve it:

- Spurious correlation hurt generalization
  - Water in the background != boating
  - A simple post processing might help
  - I want to avoid expensive LLM calls
    - Cosine similarity with embeddings might work

In Summary

Summary

- I found a very easy solution to speed up the computation
  - Original implementation takes several hours per query
  - Mine takes ~1 mins
- My results are qualitatively good
  - But not as good as the ones on the paper
- Some simple post processing might
  - Improve the numbers even further

Thanks for listening 🤗