Zero-Shot Video Retrieval With Vision-Language Models

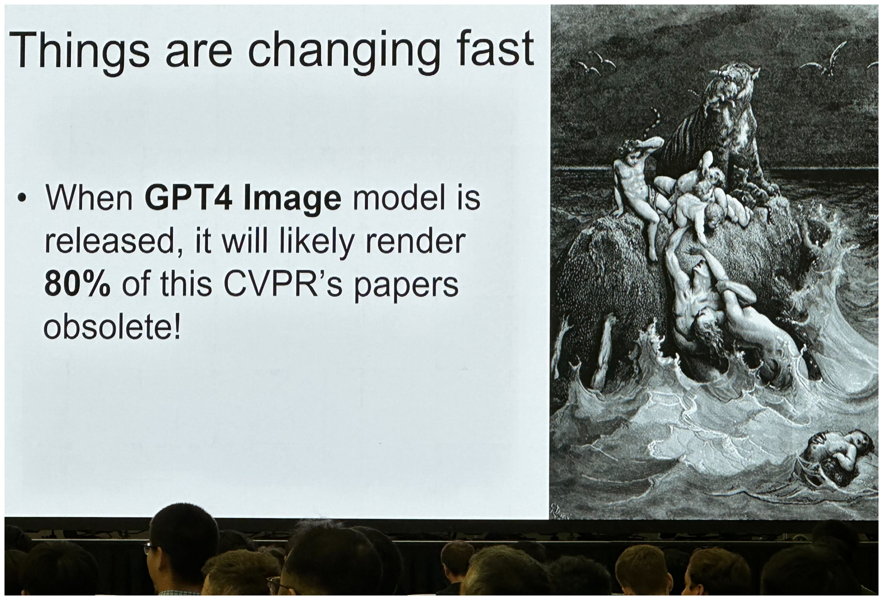
CSC2508 Final Report
Ben Agro
`agrobenj, 1005857020`

April 11, 2024

Abstract

Foundation models like Large Language Models (LLM) and Vision Language Models (VLM) have been shown to outperform task-specific models in many domains with little to no fine-tuning. This new artificial intelligence paradigm inspires us to apply pre-trained VLMs to text-to-video retrieval. While these models cannot directly process videos, we investigate a simple workaround: prompting the VLM to provide detailed descriptions of video frames to generate text documents that can be used for semantic text search. We show this simple approach provides a competitive baseline for zero-shot video retrieval using the MSR-VTT benchmark, indicating the promise of applying foundation models to the task of video retrieval. We present extensive ablations to understand which parts of the system are important for performance and highlight many avenues for future work on applying VLMs to video retrieval.

1 Introduction



Things are changing fast

- When **GPT4 Image** model is released, it will likely render **80%** of this CVPR's papers obsolete!

Figure 1: Alexei A. Efros at the 2023 Conference on Computer Vision and Pattern Recognition (CVPR). Towards testing this hypothesis, in this work, we apply the latest open-source Vision Language Models to the task of video retrieval from text queries.

In recent years, Large Language Models (LLMs) have proven to be useful for a wide variety of tasks, from chat-bots [13, 3] to text retrieval [20, 28, 11, 25]. These LLMs are pre-trained on large datasets of text from the internet and exhibit excellent zero-shot (no additional training) and fine-tuned (minimal additional training) performance.

Building on the success of LLMs, Vision-Language Models (VLMs) have emerged to suit new use cases like image description and text-to-image generation [7, 2, 22]. These so-called "foundation models" depart from the norm for most of the 2010s: training a deep neural network for one or a few related tasks. Some, like pre-eminent researcher Alexei A. Efros (see fig. 1), believe that these foundation models will soon surpass most, if not all, previous task-specific models.

Towards testing this hypothesis, in this work, we apply the latest open-source VLMs zero-shot to *video retrieval from text queries*. In this task, users ask text queries, and the system retrieves the top k relevant videos. Modern approaches to this task train deep neural networks to embed the text query and videos in a shared vector space. Distance/similarity measures can be used to estimate how well a text query (embedding) matches a video (embedding), and the top k most similar videos can be retrieved using k-nearest neighbors (KNN).

In this work, we use a (relatively small) pre-trained VLM to describe images from the given videos to generate a text document for each video. We retrieve the most relevant documents (or sections of documents) to a provided text query using semantic search methods. This simple approach achieves competitive zero-shot video retrieval results without requiring task-specific models or training. We conduct extensive ablations investigating the effect of model size, semantic search method, using audio, directly using VLM embeddings instead of text generations, and using a sliding context window to endow the VLM with some temporal understanding of the videos.

In the remainder of this paper, we will explain related work and background knowledge, describe our method and its variants, and present and discuss results. We have a Google Colab demo: https://colab.research.google.com/drive/1yzXtIWKpKGXke4TqUAGQMtrVbSj_PRsh?usp=sharing, and our code is available to reproduce our results here: https://github.com/BenAgro314/CSC2508_final_project.

2 Related Work

2.1 Video Retrieval

Given a query sentence, video retrieval aims to search for a video that is semantically relevant to the query sentence from a video database [4]. Modern approaches to this task train deep neural networks to map from text and video to embeddings in a shared feature space, trained using text-video pairs from the internet [10, 9, 12, 14, 1]. Videos are retrieved by finding the top k video embeddings with the highest cosine similarity to the text query embedding. These approaches rely on training task-specific models on paired video-text data, which is expensive and time-consuming. In contrast, our work uses pre-trained VLMs and does not require *any training for video retrieval*.

2.2 Multi-Modal Foundation Models

Recent developments in Large Language Models (LLMs) initiated mainly by the GPT family of models (GPT-2, GPT-3, ChatGPT, GPT-4) have inspired developments in multi-modal foundation models that leverage diverse data modalities. Vision Language Models (VLMs) utilize the modalities of images and text due to their abundance on the internet, computational tractability, and various downstream applications. In this work, we use the open-source VLM LLaVA [7], which jointly trains a fine-tuned version of the LLaMA LLM [21] and the CLIP visual encoder [16]. *Much* more powerful VLMs such as GPT-4 Vision [13] and Gemini Ultra [3] are becoming available through APIs, but these are currently too expensive to be used to process the frames of many videos.

2.3 Semantic Text Search

Modern text search (e.g., question-answering, document retrieval) is accomplished through the use of text embeddings, which are vectors that represent the semantic information in a piece of text (e.g., a sentence or document) [6]. Text embeddings can be compared via various distance or similarity measures (e.g., cosine similarity) and used in k-nearest-neighbours search for similar text. In this work, we utilize the embeddings from AnglE [6], a model that provides state-of-the-art text embeddings across various tasks. AnglE uses a new optimization procedure in complex space to overcome the problem of vanishing gradients inherent to the cosine similarity objective commonly used during training.

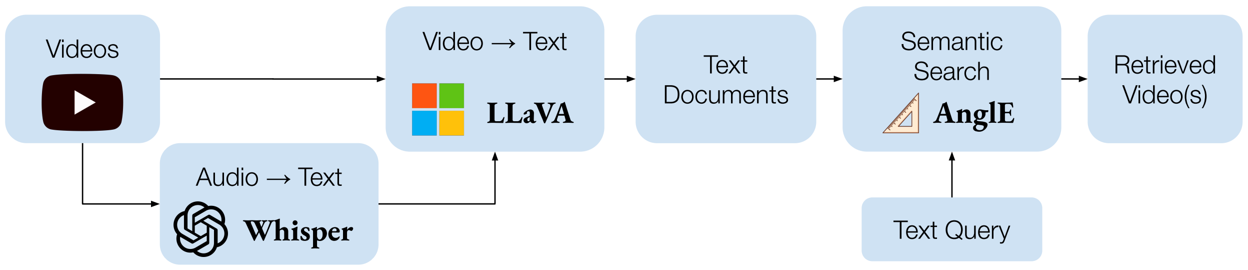


Figure 2: A diagram of our proposed system.

3 Method

Refer to fig. 2 for a high-level diagram of our proposed approach. OpenAI's Whisper speech recognition system first processes the audio from the database of videos [17] to produce timestamped closed captions (e.g., what people say in the videos). Videos are processed frame-by-frame, using only one video frame per second to reduce overhead. The video frame and associated closed caption (if available) are ingested by the VLM LLaVA [7], which is prompted to produce a detailed description of the video frame, taking into account the closed caption. The frame descriptions are aggregated for the entire video to create a text document. Given a text query, we employ semantic search to find the most relevant documents/videos. To do this, we embed all the documents using AnglE (angle optimized text embeddings) [6] and embed the text prompt. The top k document embeddings are retrieved via cosine similarity, and the corresponding video(s) are returned.

Below, we describe each component in more detail.

3.1 Speech Recognition

For some videos, the content and context are provided mainly in narration/speech. However, most multi-modal LLMs only accept images, not audio, in their additional modalities. Thus, we used OpenAI's Whisper [16], a speech recognition system, to generate closed captions for each video. To scale this to many videos, we employed `wHISper.cpp`, which allows for high-performance inference of Whisper (5-10x faster than real-time).

3.2 Video to Text

To extract video frames into text descriptions, we employed the VLM LLaVA [7]. LLaVA is available in two sizes: 7B parameters and 13B parameters. To run this efficiently on our available hardware, we use a 5-bit quantized version of LLaVA 13B, along with `LLaMA.cpp`, a library for high-performance inference of LLMs.

For each video, we split it into a sequence of images at one frame per second for computational tractability. LLaVA receives the following prompts:

```
A chat between a curious human and an artificial intelligence assistant. The assistant gives
helpful, detailed, and polite answers to the human's questions.
USER: I am watching a video. The video is at the following image: <insert image tokens here>.
Using this context, caption the image in detail.
ASSISTANT: <sample LLaVA with temperature 0.1>
```

We highlight in green the "system prompt". If there are closed captions associated with that frame, we change the prompt to include them:

```
A chat between a curious human and an artificial intelligence assistant. The assistant gives
helpful, detailed, and polite answers to the human's questions.
USER: I am watching a video. The video is at the following image: <insert image tokens here>,
```

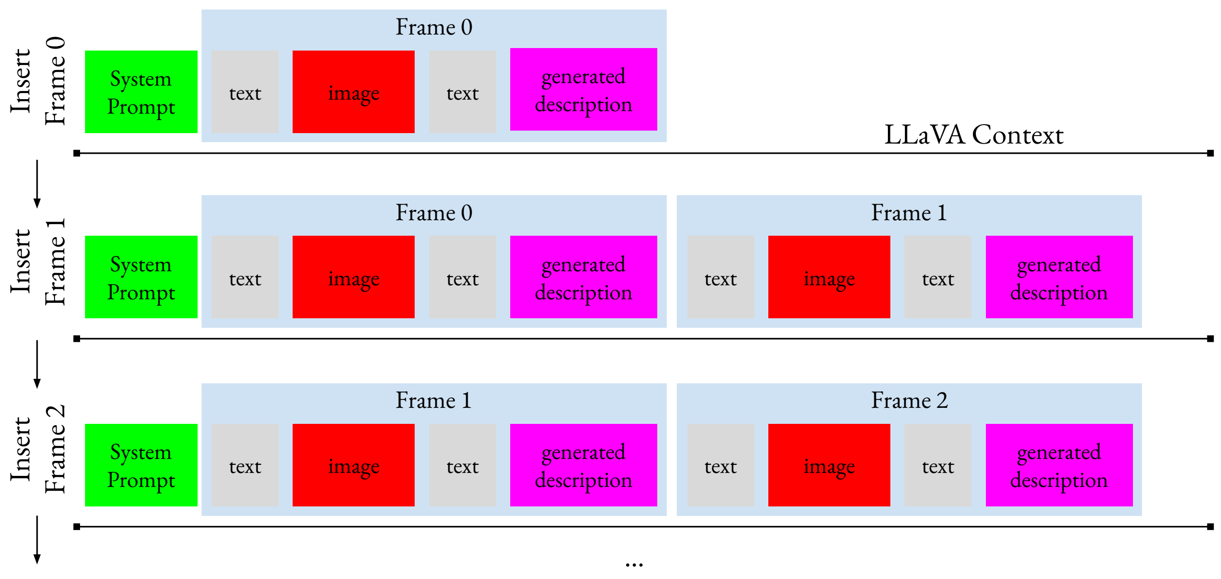


Figure 3: A diagram of the sliding window mechanism for processing videos with LLaVA. The "generated descriptions" are sampled from LLaVA and saved to create a "text document" for each video.

```
with the following closed captions: <insert closed captions here>
Using this context, caption the image in detail.
ASSISTANT: <sample LLaVA with temperature 0.1>
```

LLaVA has a context window of 2048 tokens, which can fit only 2-3 video frames in practice. We use a sliding window approach to endow LLaVA with a temporal understanding of the video while adhering to the limited context window. Namely, after the first two frames are processed using the above prompts, we remove the first frame from the context, slide the second frame to the front of the context (without moving the system prompt), and then add the subsequent frame to the now free part of the context. See fig. 3 for a detailed illustration. The generated descriptions are aggregated for all frames in the video to create a "text document".

3.3 Text Search

We want a text-search method that understands semantics. For example, in response to an image of hobbits from the Lord of the Rings movies, LLaVA might generate the caption "The image depicts three small boys standing next to a wizard in a setting that looks like Middle Earth". An example query might be "Frodo Baggins" (the name of a hobbit). Although the keywords "Frodo" and "Baggins" do not appear in the caption, it should still be ranked as relevant due to the tangential Lord of the Rings reference "Middle Earth."

To accomplish this, we employ the latest text embedding approach AnglE [6], to generate embeddings for the text documents and text queries and then use cosine similarity to retrieve the most relevant documents. These embeddings are 1024-dimensional vectors that encode semantic information about the text. Specifically, we generate a separate embedding for every LLaVA-generated caption in the document (each document has a set of associated embeddings, one for each second of the corresponding video). We also embed the text query. We score document similarity to the query sentence based on the *most similar* embedding in that document. Then, we return the top k most similar documents.

4 Experiments

In this section, we describe and discuss our experiments: the dataset and metrics we used for evaluation and results for comparison against the state of the art and various ablations.

Dataset and Metrics The dataset we used for evaluation is "Microsoft Research Video to Text" (MSR-VTT) [24] consisting of 10,000 video clips from 20 categories; each video clip is annotated with 20 English sentences. For comparison against the state-of-the-art (table 1) we use the standard "1k-A" test split of 1000 videos from prior works [27, 5, 8, 26]. For our ablations and additional experiments (all other tables), we use the smaller validation split of MSR-VTT with 497 videos.

The evaluation metric is Recall @ k: the percentage of query sentences with their associated video retrieved in the top k videos. Following prior works [23, 5, 8, 26], we report Recall @ 1, Recall @ 5, Recall @ 10, and the average recall across $k = 1, 5, 10$.

Zero-Shot Text-to-Video Retrieval Figure 4 and table 1 compare our method (described section 3) against the state-of-the-art zero-shot video retrieval methods from the MSR-VTT leaderboard [27]. Note that we omit methods from the leaderboard that trained on MSR-VTT (e.g., for pre-training). While our method does not achieve state-of-the-art performance, it provides a competitive baseline without requiring task-specific models or training. These results indicate the promise of applying Vision-Language Foundation models to various tasks, including video retrieval. We note that we outperform all methods published before mPLUG-2 [23] in February of 2023.

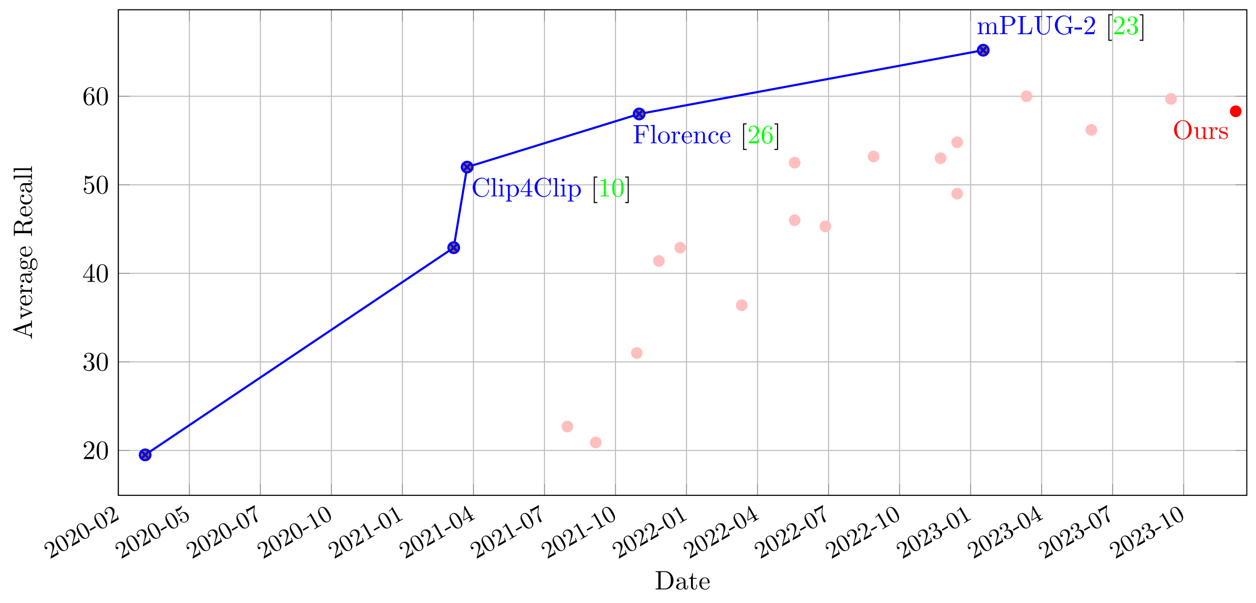


Figure 4: Average recall of recent zero-shot video retrieval methods [27]. The state-of-the-art (upper frontier) is colored in blue, our method is highlighted in red, and the other methods are colored pink. Note that we omit methods from the leaderboard that trained on MSR-VTT (e.g., for pre-training).

	Recall @ 1	Recall @ 5	Recall @ 10	Average
mPLUG-2 [23]	**47.1**	**69.7**	**79.0**	**65.3**
UTM-L [5]	42.6	64.4	73.1	60.0
BT-Adaptor [8]	40.9	64.7	73.5	59.7
Florence [26]	37.6	63.8	72.6	58.0
Ours	38.8	63.0	73.0	58.3

Table 1: Comparing our method against the SOTA zero-shot text-to-video retrieval methods on the MSR-VTT [24] benchmark [27].

Effect of VLM Model Size Table 2 investigates the effect of VLM parameter count on video retrieval performance. We see that LLaVA 13B outperforms LLaVA 7B, indicating that using even larger VLMS like GPT-4 would yield impressive results.

	Recall @ 1	Recall @ 5	Recall @ 10	Average
LLaVA 7B	42.8	71.0	79.6	64.5
LLaVA 13B	**43.2**	**71.9**	**80.2**	**65.1**

Table 2: Comparing video retrieval performance of our system across different VLM model sizes. Note that due to hardware constraints, all models used 5-bit quantization.

Effect of text retrieval method In table 3, we investigate various methods for text retrieval with a fixed set of documents (those produced by LLaVA 13B in table 1). *BM25* [19] ranks each document based on its keywords with respect to the query keywords. *SentenceTransformer* [18] maps each caption to a 768-dimensional embedding for semantic search or clustering tasks (we specifically use the `all-mpnet-base-v2` model). We try three different methods of aggregating these sentence-level embeddings into document-level retrieval scores:

- Max pooling: Produce a single embedding vector for each document by taking the maximum features across all captions in each document.
- Mean pooling: The same as max pooling, but taking the average features.
- Max sim: A document's "max sim" score is the maximum cosine similarity between the query embedding and its caption embeddings.

In addition to SentenceTransformer, we ablate the same three options using the embeddings from AnglE [6], specifically using the `UAE-Large-V1` model. We observe that AnglE using max similarity gives the best results; thus, we use that for our main method and results.

	Recall @ 1	Recall @ 5	Recall @ 10	Average
BM25	29.5	51.8	60.6	47.3
SentenceTransformer max pooling	25.4	52.2	64.6	47.4
SentenceTransformer mean pooling	37.4	65.9	74.8	59.4
SentenceTransformer max sim	37.8	65.7	75.2	59.6
AnglE max pooling	22.4	48.7	60.1	43.7
AnglE mean pooling	40.2	68.8	77.8	62.3
AnglE max sim	**43.2**	**71.9**	**80.2**	**65.1**

Table 3: Comparing the video retrieval results using different text search methods. In this experiment, the documents generated from the videos are fixed.

Effect of Using Audio Table 4 shows the video retrieval results with and without using the whisper-generated closed captions in the VLM prompt. As expected, adding audio increases performance because, for some videos, the spoken words provide as much, if not more, information about the content of the video than the images.

	Recall @ 1	Recall @ 5	Recall @ 10	Average
Without Audio	42.4	70.5	79.3	64.1
With Audio	**43.2**	**71.9**	**80.2**	**65.1**

Table 4: Comparing video retrieval with and without using closed-captions from Whisper as input to the VLM.

Effect of Sliding Context Window In this experiment, we ablate the effect of using the sliding context window mechanism depicted in fig. 3. We compare it against the simple alternative of captioning each frame individually and independently of one another. The results are presented in table 5, where we observe significant gains in recall from using the sliding window. The sliding window gives the model a temporal understanding of the video, which is necessary to understand actions and content spanning multiple frames.

	Recall @ 1	Recall @ 5	Recall @ 10	Average
Without Sliding Window	39.9	68.0	76.8	61.6
Sliding Window	**43.2**	**71.9**	**80.2**	**65.1**

Table 5: Comparing video retrieval results with and without using the sliding context window depicted in fig. 3.

Can we use LLaVA's embeddings instead of text generations? One natural question is why we are using text outputs of LLaVA instead of the last layer of features before token sampling. These features contain all the information required to sample the text. Motivated by this observation, we tried saving LLaVA's embeddings generated for each input video frame instead of the generated text. We also embedded the query using LLaVA. We used the max-similarity approach to retrieve the top k most relevant documents. To ensure the embeddings for the query and the video frames have a similar semantic meaning, we adjust the prompts for LLaVA. We use the same system prompt as in section 3.2. We use the prompt "`USER: Describe the following image in detail for embedding the video frames. ASSISTANT:`", and for embedding the query text, we use the prompt "`USER: Describe the following text in detail. ASSISTANT:`". We design these prompts such that the text that follows the prompts (and thus the embeddings used to generate that text) would be as similar in meaning as possible.

Table 6 presents the results, where we observe that using LLaVA's embeddings performs much worse than using text. We hypothesize that LLaVA's embeddings are optimized for future prediction of text and not text retrieval tasks. This result explains why works like AnglE [6] fine-tune LLaMA's embeddings for text retrieval.

	Recall @ 1	Recall @ 5	Recall @ 10	Average
Embeddings	10.9	24.6	32.9	22.8
Text	**43.2**	**71.9**	**80.2**	**65.1**

Table 6: Comparing using LLaVA's embeddings and LLaVA's image captions to represent videos.

Failure Cases In Figure 5, we introspect some failure cases by showing the query sentence, the ground truth video, and the top video retrieved by our system. We notice that the videos retrieved mostly align with the query; however, the video retrieval task is ambiguous as multiple videos reasonably match the query sentence. For the failure case with the query sentence "a throwback band lipsynchs poorly while they stand in front of a church", the model incorrectly retrieved a choir singing inside a church. While visually similar, these videos have very different meanings. Understanding poor lipsynching would require more fine-grained visual and audio understanding than our system currently has (because it processes one frame per second and only gets audio information through closed captions).

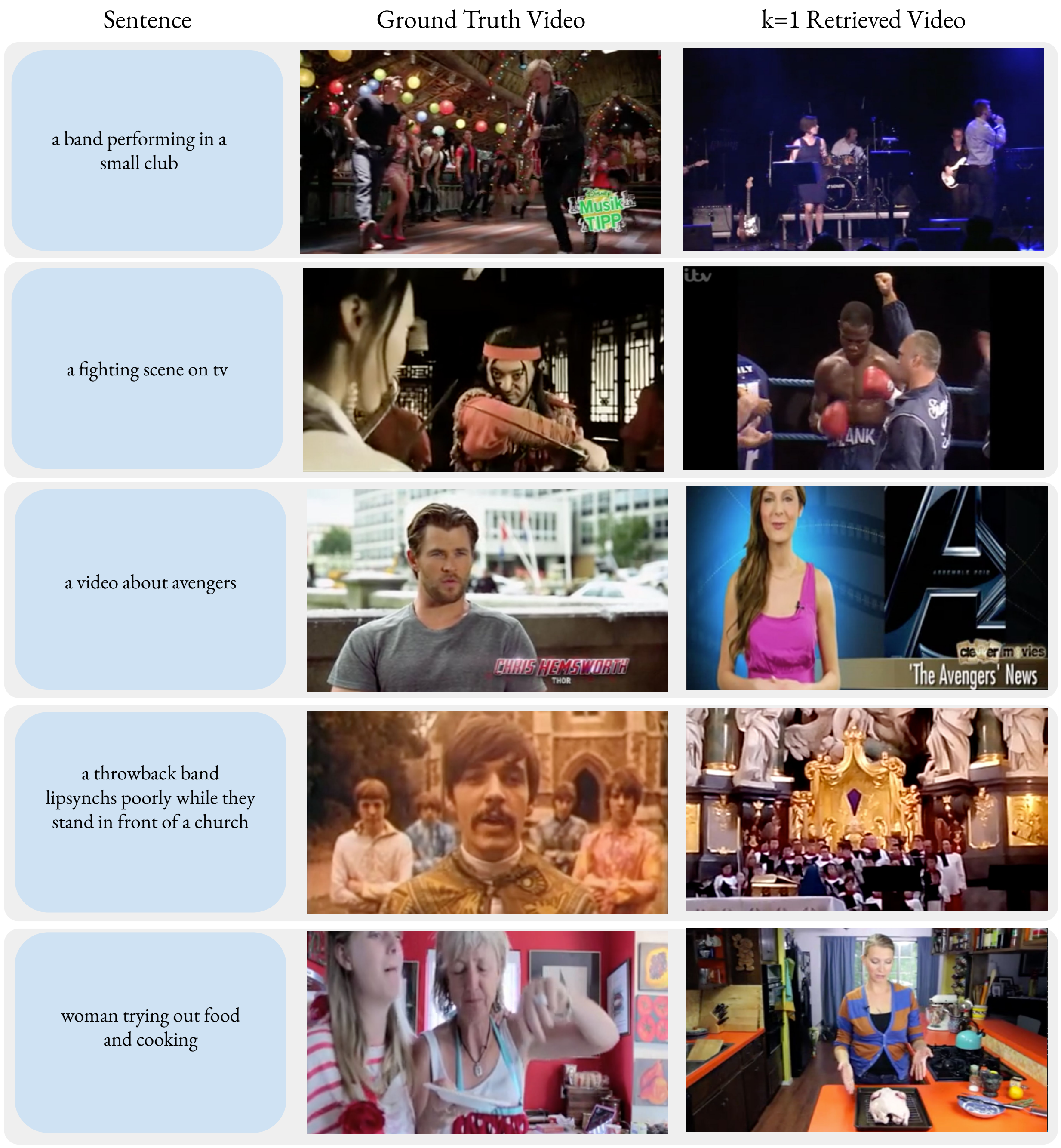


Figure 5: Visualizing some failure cases of our model.

5 Conclusion and Future Work

In this work, we have investigated the applicability of Vision Language Models to video retrieval by generating captions for the videos frame-by-frame. In the zero-shot setting, this technique achieves competitive results with the current state-of-the-art, indicating that there is promise in further investigation. In this work, we only scratched the surface of the potential applications of foundation models to video retrieval. Some directions for future work include:

- Applying larger VLMs like OpenAI's GPT-4 Vision [13] and Google's Gemini [3]. Furthermore, there are now open-source VLMs like CogVLM [22] that greatly outperform LLaVA.
- Using VLMs with a larger context window than LLaVA's 2048 tokens (e.g., GPT-4 Vision Turbo has a context length of 128k) would allow for a large sliding window and for more frames to fit in context, bringing improved temporal understanding of the videos.
- The current text retrieval system performs nearest-neighbor search among text embeddings; the document embeddings are created independently of the query. A query-dependent re-ranking step with a cross-encoder or other method (e.g., LLMs [20]) could improve text-search performance.
- The biggest bottleneck to scaling this system to larger video datasets is the inference time of the per-frame inference time of the VLM. While we used `LLaMA.cpp` to make inference as fast as possible (around 3 seconds per frame), there is room for improvement. Namely, the CLIP vision encoder was not offloaded to the GPU, greatly increasing its inference time. Offloading vision encoders to the GPU is an open issue on the `LLaMA.cpp` GitHub, and would unlock many opportunities for future research.
- While our experiments in table 6 show that using LLaVA's embeddings is ineffective for video retrieval, we believe this avenue holds promise as a more principled retrieval method. Namely, these embeddings *should* contain all the information of all possible captions that the VLM could generate, so they should be capable of improving retrieval performance. Future work in VLM prompting and fine-tuning for video retrieval could yield fruitful results.
- Due to limited computational resources, we could not extensively ablate different prompting procedures, such as system prompts. Prior work has found that VLM performance strongly depends on the prompts [15].
- In our approach, we convert audio to text using Whisper speech recognition. Investigating modern multi-modal foundation models that include audio as an input stream could improve performance (see the failure cases discussed above).

References

[1] Max Bain et al. “Frozen in time: A joint video and image encoder for end-to-end retrieval”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 1728–1738.

[2] James Betker et al. “Improving image generation with better captions”. In: *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf* (2023).

[3] *Gemini: A Family of Highly Capable Multimodal Models*. 2023. URL: https://storage.googleapis.com/deepmind-media/gemini/gemini_1_report.pdf.

[4] Jie Jiang et al. “Tencent Text-Video Retrieval: Hierarchical Cross-Modal Interactions with Multi-Level Representations”. In: *IEEE Access* (2022).

[5] Kunchang Li et al. “Unmasked teacher: Towards training-efficient video foundation models”. In: *arXiv preprint arXiv:2303.16058* (2023).

[6] Xianming Li and Jing Li. “Angle-optimized text embeddings”. In: *arXiv preprint arXiv:2309.12871* (2023).

[7] Haotian Liu et al. “Visual Instruction Tuning”. In: *NeurIPS*. 2023.

[8] Ruyang Liu et al. “One for all: Video conversation is feasible without video instruction tuning”. In: *arXiv preprint arXiv:2309.15785* (2023).

[9] Yang Liu et al. “Use what you have: Video retrieval using representations from collaborative experts”. In: *arXiv preprint arXiv:1907.13487* (2019).

[10] Huaishao Luo et al. “Clip4clip: An empirical study of clip for end to end video clip retrieval and captioning”. In: *Neurocomputing* 508 (2022), pp. 293–304.

[11] Xueguang Ma et al. “Zero-Shot Listwise Document Reranking with a Large Language Model”. In: *arXiv preprint arXiv:2305.02156* (2023).

[12] Ron Mokady, Amir Hertz, and Amit H Bermano. “Clipcap: Clip prefix for image captioning”. In: *arXiv preprint arXiv:2111.09734* (2021).

[13] OpenAI. *GPT-4 is OpenAI’s most advanced system, producing safer and more useful responses*. 2023.

[14] Jesús Andrés Portillo-Quintero, José Carlos Ortiz-Bayliss, and Hugo Terashima-Marín. “A straightforward framework for video retrieval using clip”. In: *Mexican Conference on Pattern Recognition*. Springer. 2021, pp. 3–12.

[15] *Prompt Engineering*. URL: https://platform.openai.com/docs/guides/prompt-engineering.

[16] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[17] Alec Radford et al. “Robust speech recognition via large-scale weak supervision”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 28492–28518.

[18] Nils Reimers and Iryna Gurevych. “Sentence-bert: Sentence embeddings using siamese bert-networks”. In: *arXiv preprint arXiv:1908.10084* (2019).

[19] Stephen E Robertson et al. “Okapi at TREC-3”. In: *Nist Special Publication Sp* 109 (1995), p. 109.

[20] Weiwei Sun et al. “Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent”. In: *arXiv preprint arXiv:2304.09542* (2023).

[21] Hugo Touvron et al. “Llama: Open and efficient foundation language models”. In: *arXiv preprint arXiv:2302.13971* (2023).

[22] Weihan Wang et al. “Cogvlm: Visual expert for pretrained language models”. In: *arXiv preprint arXiv:2311.03079* (2023).

[23] Haiyang Xu et al. “mplug-2: A modularized multi-modal foundation model across text, image and video”. In: *arXiv preprint arXiv:2302.00402* (2023).

[24] Jun Xu et al. “Msr-vtt: A large video description dataset for bridging video and language”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 5288–5296.

[25] Wenhao Yu et al. “Generate rather than retrieve: Large language models are strong context generators”. In: *arXiv preprint arXiv:2209.10063* (2022).

[26] Lu Yuan et al. “Florence: A new foundation model for computer vision”. In: *arXiv preprint arXiv:2111.11432* (2021).

[27] *Zero-Shot Video Retrieval on MSR-VTT*. 2023. URL: https://paperswithcode.com/sota/zero-shot-video-retrieval-on-msr-vtt.

[28] Yutao Zhu et al. “Large language models for information retrieval: A survey”. In: *arXiv preprint arXiv:2308.07107* (2023).