Leveraging Lightweight AI for Video Querying in a RAG Framework

Adam Chun
Stanford University
adamchun@stanford.edu

Emily Hsu
Stanford University
ehsu24@stanford.edu

Abstract

In recent years, the exponential growth of video content across diverse fields has necessitated the development of advanced tools for managing, analyzing, and retrieving information from extensive video archives. Current approaches to video-language understanding that leverage large language models (LLMs) and multimodal preprocessing are computationally expensive and slow, limiting their utility in time-sensitive applications. This paper builds upon existing research to streamline video content querying using retrieval augmented generation (RAG) and lightweight AI models for indexing.

Our study introduces a system designed to extract information from videos efficiently by bypassing the extensive video-to-text conversion process. Our proposed solution uses CLIP and DETR to produce frame embeddings and textual indices, respectively, which can then be used in RAG to locate relevant video segments. We fine-tuned the out-of-box CLIP model on MSR-VTT video-captioning data and augmented the fine-tuning process using the Atemporal Probe (ATP) model. Then, we evaluated our model against existing state-of-the-art model, VLog. We find that our approach significantly reduces runtime and memory allocation during the video pre-processing stage while maintaining comparable accuracy in question-answering tasks.

1. Introduction

In recent years, the proliferation of video content across various domains—ranging from education and entertainment to healthcare and security—has led to an exponential increase in the amount of long-form video data generated daily. This surge has created a pressing need for advanced tools and techniques to effectively manage, analyze, and retrieve information from these extensive video archives. Traditional video analysis methods, which often rely on manual review, are becoming increasingly impractical due to the sheer volume of data.

One emerging solution to this challenge is the development of systems capable of understanding and responding to natural language prompts about video content. Despite the grand strides that large language models (LLMs) have made in understanding text, video understanding remains a frontier that is relatively unexplored. Recent attempts [2] [15] have utilized multimodal LLMs (ex. ChatGPT-4V) in combination with multimodal pre-processing (ex. automatic speech recognition, or ASR) to analyze videos frame-by-frame and output a long textual document that describes the video's scenes which can be inputted into an LLM for video-understanding tasks.

However, this approach is both time intensive and computationally costly because it converts all video data to text upfront. Thus, it falls short for many real-world, time-sensitive tasks that require fast and inexpensive retrieval of objects in video footage, such as quickly identifying security threats in many hours of security footage or identifying inappropriate video content on social media platforms before it spreads to sensitive user groups. Our research aims to address these limitations and enable querying within long videos by employing retrieval augmented generation (RAG) [13] with lightweight AI models for indexing.

1.1. Problem Statement

The problem we aim to address is how to streamline the process of extracting information from videos for VQA (visual question-answer) tasks. That is, when prompted with an "Is there _____ in the video?" question, it can accurately identify the presence of the object (or lack thereof) along with the reference time.

We will do this by circumventing the long multimodel to text conversion times required for generating a full video summary; instead, we introduce a faster method of video processing that prepares an index for the content in the video. Then, using a RAG approach [13], we select the relevant index in the video and extract information based on the user query. Specifically, our system will take in long-form video content and generate text and frame embeddings, with which we can then use to respond to user prompts about the video.

2. Related Work

Many researchers have made significant contributions in video-language understanding. Our proposed model builds on top of existing work in the field using insights gained from past experimentation.

2.1. Video-to-script generation

With the rise of LLMs, researchers are beginning to integrate pretrained models for video understanding. One prominent example is VLog (Video as a Long Document) [2], which uses BLIP-2 [14] and GRiT [18] as dense image captioners, Whisper for automatic speech recognition (ASR), and ChatGPT as an LLM reasoner. VLog [2] generates a text script from a long video, which the LLM then uses as input to respond to user queries.

Inspired by this architecture, a recent exemplar using LLMs for video understanding is MM-VID, which employs GPT-4V(ision) to "to transcribe multimodal elements into a long textual script" [15]. Given an input video, MM-VID performs multimodal pre-processing, including automatic speech recognition (ASR) to extract transcriptions from the video. The input video is then split into multiple clips using scene detection [4]. Then, clip-level video frames are used as input for GPT-4V, which generates a detailed description for each video clip. Finally, GPT-4 generates a coherent script for the full video from the clip-level video descriptions, ASR, and available video metadata [15].

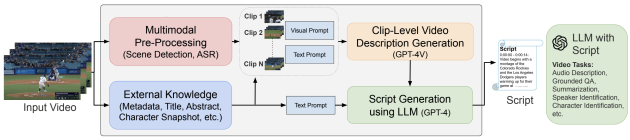

Figure 1. MM-VID consists of four modules: (1) Multimodal Pre-Processing, (2) External Knowledge Collection, (3) Clip-Level Video Description Generation, and (4) Script Generation.

The strengths of this approach is that it can produce a rich textual representation of the video for many downstream tasks like character identification, audio description generation, and grounded question-answer (QA) [15]. The drawback is that the upfront processing time for video-to-text generation can be costly and slow, especially when using complex AI models like BLIP-2 [14] and GRiT [18] to process long videos.

2.2. Lightweight AI models

To address the issue of model complexity, two popular lightweight AI models stand out: CLIP [17] and DETR [9]. While BLIP (Bootrstrapping Language-Image Pre-training) [14] is a transformer-based architecture that uses off-the-shelf frozen pre-trained image encoders and frozen large language models to enable image-to-text generation, CLIP (Contrastive Language-Image Pre-training) [17] jointly trains an image encoder and a text encoder on text-image pairs to enable zero-shot classification . Although BLIP-2 is highly expressive, its lightweight counterpart, CLIP, is highly efficient and is found to be more accurate at zero-shot ImageNet classification [1].

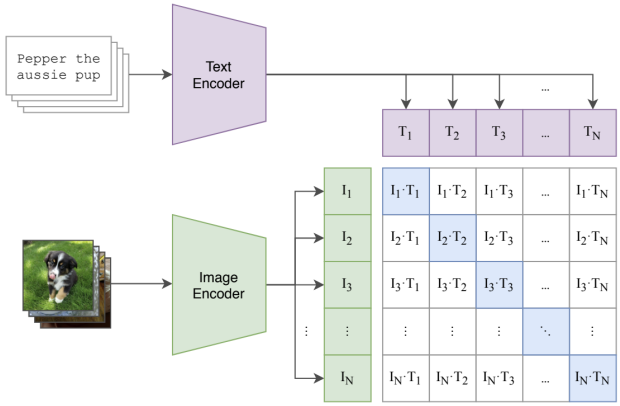

Figure 2. Architecture of CLIP.

Among state-of-the-art image-captioning models, GRiT [18] (Generative RegIon-to-Text transformer) can produce rich descriptive sentences of an image scene. Rather than classifying objects from a closed set, GriT produces open-set object descriptions in free-form using a text decoder that "autoregressively generates a list of text tokens to describe the given object" [18]. On the other hand, DETR (DEtection TRansformer) [9] views object detection as a direct-set object prediction problem using a conventional CNN backbone. Although GRiT [18] can generalize to unseen object classes, DETR [9] is much faster at inference time because it doesn't rely on autogressive models.

2.3. Atemporal Probe (ATP)

Another recent breakthrough in the field of video understanding is the Atemporal Probe model (ATP) [8]. In this paper, the researchers found that information of event temporality is often not necessary to achieve strong state-of-the-art performance, even compared with large-scale video-language models. The ATP model selects the most informative frames from a video without leveraging temporal information, making it more efficient than traditional methods that maintain sequential data.

2.4. Retrieval Augmented Generation (RAG)

Finally, RAG [13] is a significant contribution to the field of LLMs because it allows the pretrained model to access an "external memory" as context for its output. In relation to video understanding, LLMs can leverage text

documents generated from long videos using RAG to respond to user queries. Solutions like MM-VID [15] use the Naive RAG paradigm [11] to support interactive querying, but there may be opportunities to expand the capabilities of video understanding models with the Advanced or Modular RAG paradigms that incorporates re-ranking to minimize the parts of the long video that needs to be analyzed [11].

The most recent development in using RAG for video understanding is iRAG [7], or incremental RAG. Instead, converting the entire video into a text document upfront, iRAG "quickly indexes large repositories of multimodal data, and in the incremental workflow, it uses the index to opportunistically extract more details from select portions of the multimodal data to retrieve context relevant to an interactive user query" [7]. They also employ a novel re-ranking method in the Advanced RAG paradigm [11] to reduce the portions of the video that need to be considered for analysis.

3. Datasets and Features

3.1. MSR-VTT

Microsoft Research Video to Text (MSR-VTT) is a large-scale video dataset commonly used for video-captioning tasks. This dataset contains 10K video clips from 20 different categories, and each video clip is annotated with 20 English sentences by Amazon Mechanical Turks, amounting to 200K clip-sentence pairs in total. We use the standard split of 6,513 clips for training, 497 clips for validation, and 2,990 clips for testing. The duration of each clip is betwen 10 to 30 seconds.

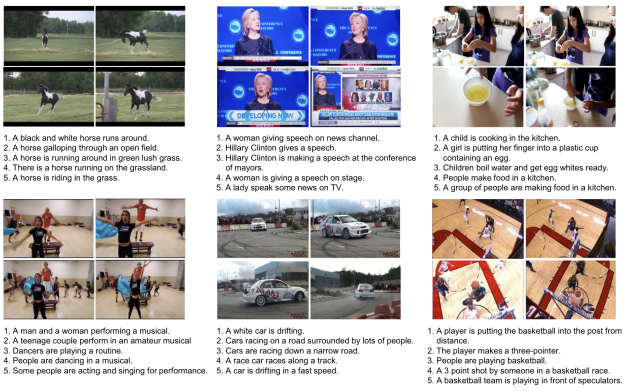

Figure 3. Examples from the MSR-VTT dataset. Each video is represented by 4 frames and assigned five human-labeled sentences.

Data preprocessing. Since CLIP is designed for processing single images as opposed to videos, we preprocessed the dataset by randomly extracting 16 frames from each video clip and organizing them into directories. Since each video is a continuous scene, we don't need to sample as frequently to get a good representation of the video. The metadata for each split (train, validation, test) was stored in CSV files, with each row containing the path to the video frames directory and the associated caption.

3.2. VQA-v2

The dataset we use for evaluating our baseline model against our proposed model is the Visual Question Answering (VQA-v2) dataset [6], which contains open-ended questions about images. These questions range across various task domains, requiring the model to identify the presence, orientation, number, location, color, type, or movement of objects.

Figure 4. Examples from the VQA-v2 dataset

The full dataset contains 265,016 images with at least 3 questions (5.4 questions on average) per image, 10 ground truth answers per question, and 3 plausible (but likely incorrect) answers per question. Each image has a maximum dimension of 500x500 pixels with 72 DPI. For the purposes of our experiment, we extract 7,799 questions from the dataset that start with "Is there..." to see how well our model performs against the baseline on identifying the presence of objects and retrieving the corresponding scene.

Data preprocessing. For the VQA-v2 dataset since the dataset contains images, not videos, we concatenate the images together to form a 29-minute long video at 1 frame per second. To do this, we define the dimensions of the video by the first image in the stack, and then we resize, center, and pad all subsequent frames to align with these dimensions. Our video consists of 600 images, which is randomly shown for a duration from 1-5 seconds.

4. Methods

For our baseline method, we use VLog [2] out-of-the-box for preprocessing the video. VLog uses BLIP-2 [14] and GRIT [18] as dense image captioners. The baseline method outputs a text summary of each second in the video, forming a document which is then used to respond to user queries.

Our proposed method is a lightweight version of VLog [2] that incorporates key frame sampling, the details of

which are expanded upon in the section below. All of the training and fine-tuning was performed on a VM equipped with 1 NVIDIA Tesla P4 GPU.

4.1. Finetuning CLIP using ATP for frame selection

Given that CLIP is a trained on a large corpus of online images, we fine-tuned the CLIP using the pre-processed MSR-VTT dataset that's specialized for image-captioning tasks. This takes advantage of transfer learning given that image-captioning is very similar to identifying objects present in a video. We utilized the pre-trained CLIP model [3] from the Hugging Face library, which combines a Vision Transformer (ViT) for image processing and a Transformer for text processing. We use the ViT-B/32 vision encoder-based CLIP mode for evaluation. Since the input to CLIP is an image-text pair, we use the Atemporal Probe (ATP) model [5] to extract a single frame from the video. ATP is built [8] on top of frozen image and text encodings from CLIP, and it learns to "select a single (frozen, image-derived) embedding that can provide as strong a signal as possible for the final task" [8].

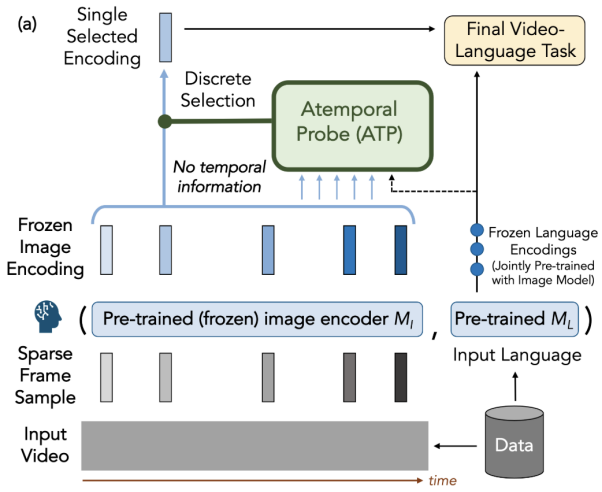

Figure 5. Architecture of ATP.

At a high-level, we leverage CLIP to extract a set of image and language representations; our ATP model then selects a single frame from the representations, and forwards that to our final CLIP model for the fine-tuning step. Formally, given a video V with frames $F = \{v_1, v_2, ..., v_n\}$, ATP processes these frames using CLIP to produce representations:

$$M_i(F) = \{x_1, x_2, \ldots, x_n\} \quad (1)$$

Finally, ATP select a single representation $x_i \in \{x_1, x_2, \ldots, x_n\}$ to pass to our final CLIP model for fine-tuning. After selecting the best frame associated with the caption, we fine-tuned CLIP on the selected frames using the AdamW [12] optimizer with a learning rate of 1e-6 and weight decay of 0.2. We trained the model for 50 epochs. For each batch, we loaded in the frame-caption pair using the CLIP processor. The processed tensors were then passed through the CLIP model, which computes a symmetric cross-entropy loss 6. CLIP aims to maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch while minimizing the cosine similarity of the embeddings of the $N^2 - N$ incorrect pairings [17]. We optimize a symmetric cross entropy loss over these similarity scores, which is used to update the model weights through stochastic gradient descent.

```
# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits,
    labels, axis=0)
loss_t = cross_entropy_loss(logits,
    labels, axis=1)
loss = (loss_i + loss_t)/2
```

Figure 6. Pseudocode of symmetric loss function in CLIP model for image loss (loss_i) and text loss (loss_t) [17].

4.2. Integrating CLIP + DETR with RAG

We then processed the video inputs from the VQA-v2 dataset using the fine-tuned CLIP model and DETR [9] to produce image-language encodings. As alluded to in the Related Works section, these models are lighter versions of the vision models utilized in VLog [2], namely BLIP2 [14] and GRIT [18]. To leverage RAG for retrieving the index of relevant clips, we build the following database structure introduced by the recent iRAG paper [7] from scratch.

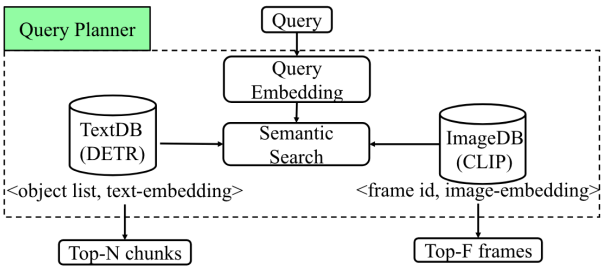

Figure 7. Architecture of iRAG's Query Planner [7].

DETR forms our textual indices with descriptions of objects detected in the frame. Alongside these textual indices, we include a database of frame vectors from CLIP to improve the quality of the retrieved context. As a result, when a user query is received, we retrieve the top-N text chunks and the top-F frame chunks that are most similar to the query. We merge then these two chunks to produce our top-k context clips that is then fed as input into the LLM,

namely Chat-GPT 3.5. By utilizing lightweight models, we seek to match the performance of our baseline while reducing pre-processing times and compute resources.

5. Experiments

We ran various experiments with the same setup as iRAG [7] to compare the performance of the lightweight AI models against VLog: (1) DETR only, (2) CLIP only, and (3) DETR + CLIP. We use the pre-processed VQA-v2 dataset for evaluation.

5.1. Experimental Setup

For the DETR-only setup, we use DETR out-of-the-box to produce captions of the video frames, which is then utilized as a database which we search through utilizing FAISS [10] and the OpenAI embeddings of the query. The CLIP-only setup encodes each frame and stores it into a ImageDB; when a query is given, the query is also encoded by CLIP and is used to search ImageDB also utilizing FAISS.

5.2. Results

5.2.1 Results of Fine-tuning CLIP

We fine-tuned CLIP utilizing the same approach as Portillo-Quintero et al. [16] on 1000 frames extracted from ATP from the MSR-VTT dataset. Specifically, we extracted the key frame from each video using ATP and encoded this frame with CLIP. We then aim to maximize the cosine similarity between this key frame embedding and the ground-truth video caption embedding. We obtained a recall@1 of 23.2%.

5.2.2 Quantitave Results on VQA Task

To compare the runtime of our models to the baseline, we run the model on a 29-minute video @ 1fps. This is an important metric because we expect our lightweight models to be significantly faster than the complex AI models used in VLog [2]. In addition, we extract one frame from each unique image by utilizing PySceneDetect to detect scene changes [4].

Method	Runtime
VLog (Baseline)	48 minutes
DETR only	83.85 seconds
CLIP only	10.50 seconds
DETR + CLIP	103.11 seconds

Table 1. Runtime comparison between different experiments on a 29-minute video @ 1fps.

Method	Memory
VLog (Baseline)	10.92612 GB
DETR only	809.69 MB
CLIP only	923.49 MB
DETR + CLIP	1.714 GB

Table 2. Allocated memory comparison for preprocessing stage

In terms of runtime, the "DETR only" and "CLIP only" models are 34x and 288x faster than the VLog baseline, respectively. The combined DETR+CLIP model is 28x faster than the VLog baseline. This speed-up can be attributed to the fact that DETR [9] uses a conventional CNN which leverages parallel processing to process the video frames as opposed to GRiT [18], which relies on autogressive methods for sequential text generation. Additionally, CLIP [17] synthesizes into a linear classifier at inference time, so there is no need for addition pre-processing during runtime.

For the same reasons, DETR and CLIP require significantly less memory allocation during the preprocessing stage. We see a ten-fold difference from the combined DETR+CLIP model compared to the baseline VLog when it comes to memory allocated. We observe that the runtime and computational costs are compounded the longer the video is, and the inference cost for VLog will become prohibitive for long videos.

Next, we evaluate the models on the pre-processed VQA-v2 dataset. Since we are only using "Is there...?" questions, we can evaluate whether the model accurately identifies the presence of the object anywhere in the video.

Method	Accuracy	Recall	Precision	F1
VLog (Baseline)	**0.52**	**0.60**	0.54	**0.57**
DETR only	0.45	0.30	0.47	0.37
CLIP only	0.46	0.40	0.46	0.43
DETR + CLIP	0.51	0.50	**0.61**	0.55

Table 3. Accuracy of models on "Is there...? " VQA task.

From the results, we observe that the baseline model has the highest accuracy but DETR+CLIP produces near identical accuracy as VLog. The DETR+CLIP model has noticeably higher precision among the variants. This could be attributed to the fact that it is more sensitive to certain high-confidence object detections in an image, but this could lead to missing less prominent details, which corresponds to lower recall.

Since temporal information is important for video understanding tasks, we also evaluated whether the model can retrieve the frame / timestamp in which the object was present. Furthermore, we calculated the percentage of timestamps which overlap with the timestamps retrieved by the base-

line model. This is an important preliminary step because an object can appear multiple times in the video. Therefore, the models may retrieve multiple timestamps, so we want to see how much they overlap with each other.

Method	Timestamp Acc.	Timestamp Overlap
VLog (Baseline)	0.21	–
DETR only	0.18	0.71
CLIP only	0.16	0.75
DETR + CLIP	**0.25**	**0.79**

Table 4. Accuracy of retrieving the correct timestamp where the object in question was present in the video.

For the "DETR only" and "CLIP only" models, we notice a slight decrease in timestamp retrieval accuracy compared to the baseline. This can likely be attributed to the fact that VLog produces a more extensive and rich text representation of the video compared to individual lightweight models. However, the lightweight counterparts still retrieve a majority of the timestamps, although not all the same as VLog. Noticeably, the best performing model on the text-to-clip retrieval task is the combined "DETR + CLIP" model. This may be because we use both the text embeddings and image embeddings to retrieve the relevant frame for the query.

5.2.3 Qualitative Results on VQA Task

Figure 8. Frames retrieved by baseline, DETR, CLIP, DETR+CLIP in respective order to query "Is there a fire hydrant?"

Qualitatively, we found that the frames retrieved by the different variations of models corresponded closely to the query's main subject. As seen in 8, all of our lightweight variations were able to successfully identify fire hydrants despite large variations in color, lighting, and perspective; this consistency indicates that these lightweight models are sufficient for our query type without utilizing details with more fine-grained textual details.

5.3. Discussion

Analyzing results from the experiments, we observe that the lightweight AI models take significantly less time and memory to process long videos, while producing comparable results to the VLog baseline. We view this as a relatively small trade-off between runtime/memory and accuracy. In the context of processing and understanding long videos, our research shows that we can dramatically reduce the time it takes to process videos by using lightweight AI models for video-language understanding coupled with RAG to index and retrieve relevant frames.

We also find that by including lightweight vision-language models such as CLIP, we improve the quality of our indexing as seen in the improved performance from our DETR only model to DETR + CLIP model in Table 3.

A limitation of our experiment is that we are only running the models on "Is there...?" V-QA tasks, which do not require rich descriptions of the video. While the lightweight AI models can accurately identify the presence of objects in videos, they may not be suited for tasks that require a deeper level of video understanding like generating audio transcription, recognizing different characters or activities, and identifying the causality of events.

Nevertheless, our research makes a significant contribution to the growing effort to understand long-form videos and explores a novel way to encode them for downstream LLM tasks.

6. Conclusion

In conclusion, our research demonstrates that a system using lightweight AI models, CLIP and DETR, can effectively streamline video content querying through RAG. By avoiding the extensive and computationally expensive video-to-text conversion process of traditional models, our approach significantly reduces processing time while maintaining comparable accuracy for question-answering tasks. The experimental results show that the CLIP+DETR model provides a substantial speed-up, up to 28x faster than the VLog baseline, and performs comparably to the baseline in terms of identifying and retrieving relevant video segments.

Our findings underscores the potential of using more efficient models for real-time and resource-constrained applications. This efficiency is particularly beneficial for scenarios where quick identification of objects and events in video footage is crucial, such as security surveillance and content moderation on social media platforms.

For future work, there are several avenues worth exploring. Further fine-tuning and optimization of the CLIP and DETR models on video-language datasets could enhance their performance. Investigating the integration of more advanced RAG paradigms, such as the Advanced or Modular RAG [11], could potentially improve the retrieval of relevant video segments. Specifically, if given a complex query that requires granular details beyond the capabilities of DETR + CLIP, we would prompt our model to process retrieved frames with more heavyweight models. Fi-

nally, given more time, we would also want to evaluate the lightweight AI models on more benchmarks like MSR-VTT [20] for video-captioning and NeXT-QA [19] for reasoning about causal and temporal actions.

7. Contributions & Acknowledgements

Team Contributions. Adam spearheaded the code implementation in this project, including the data preprocessing of the VQA-v2 dataset, and integrating it with VLog, DETR, and CLIP. Emily took the lead on writing the project milestone and final report, conducting a literature review of the existing work in the space of video-based AI models and RAG, outlining the methods for the project, and analyzing the results by understanding the underpinnings of existing models. Together, we explored, discussed, decided on the frameworks and approaches we would use for this project.

References

[1] Clip: Connecting text and images. https://openai.com/index/clip/, 2021. 2

[2] Vlog: Video as a long document. https://github.com/showlab/VLog, 2023. 1, 2, 3, 4, 5

[3] Clip. https://github.com/openai/CLIP, 2024. 4

[4] Pyscenedetect. https://www.scenedetect.com/, 2024. 2, 5

[5] Revisiting the "video" in video-language understanding. https://github.com/StanfordVL/atp-video-language, 2024. 4

[6] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*, 2015. 3

[7] M. A. Arefeen, B. Debnath, M. Y. S. Uddin, and S. Chakradhar. irag: An incremental retrieval augmented generation system for videos, 2024. 3, 4, 5

[8] S. Buch, C. Eyzaguirre, A. Gaidon, J. Wu, L. Fei-Fei, and J. C. Niebles. Revisiting the "video" in video-language understanding, 2022. 2, 4

[9] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers, 2020. 2, 4, 5

[10] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou. The faiss library. 2024. 5

[11] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang. Retrieval-augmented generation for large language models: A survey, 2023. 3, 6

[12] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014. 4

[13] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. 1, 2

[14] J. Li, D. Li, S. Savarese, and S. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023. 2, 3, 4

[15] K. Lin, F. Ahmed, L. Li, C.-C. Lin, E. Azarnasab, Z. Yang, J. Wang, L. Liang, Z. Liu, Y. Lu, C. Liu, and L. Wang. Mm-vid: Advancing video understanding with gpt-4v(ision), 2023. 1, 2, 3

[16] J. A. Portillo-Quintero, J. C. Ortiz-Bayliss, and H. Terashima-Marín. A straightforward framework for video retrieval using clip, 2021. 5

[17] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021. 2, 4, 5

[18] J. Wu, J. Wang, Z. Yang, Z. Gan, Z. Liu, J. Yuan, and L. Wang. Grit: A generative region-to-text transformer for object understanding, 2022. 2, 3, 4, 5

[19] J. Xiao, X. Shang, A. Yao, and T.-S. Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9777–9786, June 2021. 7

[20] J. Xu, T. Mei, T. Yao, and Y. Rui. Msr-vtt: A large video description dataset for bridging video and language. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), June 2016. 7