Inter IIT Tech Meet 13.0

Machine Learning

GIF - Question Answering

Team-5

Index

Abstract

Video Question Answering is a multimodal task in machine learning that aims to answer a given question by making use of a provided video as the context. To this date, it is one of the toughest challenges as it requires the model to have not only excellent semantic knowledge and reasoning abilities , but also the ability to capture the relationships across spatio-temporal domains.

GIF Question Answering can be thought of as a specific subset of VideoQA, where the input video(gif) is relatively small. This can allow for a variety of optimisations to the methods applied towards solving VideoQA, some of which will be picked up in this report and implemented in the approaches.

We first conduct a brief survey of the given dataset and the methods devised to generate question answer pairs. We then present an efficient pipeline to retrieve *keyframes* from the input GIF, named as CLIP-Cut. This general purpose pipeline finds use in the subsequent approaches that were implemented. We first tried to convert the problem to a language modeling task and fine-tuned a language model on this ‘specialized’ question answering task. Then we proceeded to test the spatio-temporal abilities of Large Language Models, inspired by recent research.

Dataset

Dataset Overview

The dataset provided to us, TGIF, is a dataset that was compiled as a part of a CVPR submission in 2016. The authors were kind enough to provide a detailed description of the dataset :

BASIC STATISTICS

Splits	Train	Validation	Test	Overall
# Animated GIFs	80,000	10,708	11,360	102,068
# Frames (Total/Average)	3,258,373/40.76	431,359/40.30	453,718/39.96	4,143,450/40.62
# Shots (Total/Average)	204,553/2.56	26,559/2.48	27,933/2.46	259,338/2.54
Duration (Total/Average)	81h/3.66s	11h/3.60s	12h/3.65s	103h/3.65s
# Sentences (Total/Average)	80,000/1	10,708/1	34,101/3	125,782/1.23
# Tokens (Total/Median)	911,593/11	10,831/11	34,101/11	1,418,555/11
# Unique Tokens	10,685	4,755	7,083	12,228
Average Term Frequency	85.32	25.72	54.31	116.00

The salient points and observations are listed below:

1. The average number of frames per GIF is roughly 41, which is a motion media volume that is extremely tiny(for reference, a minute-long video on average has 1440 frames)
2. The dataset possesses a train test split, where 80k samples are for training and the rest are for testing.
3. The dataset in itself contains the url of the gif followed by a human description of the same.
4. The textual descriptions available are manually annotated, and essentially provide a rough summary of the entire GIF.
5. While there does exist a human annotated question-answer dataset , TGIF-QA, we chose not to incorporate that in any of our approaches. It classifies questions into a variety of categories such as action recognition, frame count, etc.
6. A search for the five most common words returned the following : [man, woman, with, his, on], from all the descriptions.

Synthetic Question-Answer pair generation

For the task at hand, it was important to first develop a question-answer pair dataset using synthetic methods, as manual annotation would be too time consuming. After a thorough search, we came up with an efficient pipeline to generate a decent set of synthetic question-answer pairs from the given dataset, more specifically, the descriptions.

We employ the T5-base model fine-tuned on SQuAD from HuggingFace for ***Question Generation.*** Our exact approach follows the steps listed below :

1. Consider each url, and the corresponding description.
2. Extract *potential answers* from the description, making use of the spacy library for parts-of-speech tagging. The broad categories include:
    a. living entities
    b. nouns
    c. verb
    d. prepositional phrases (eg. in a box)
    e. colors
3. Supply these potential answers to the pipeline, and retrieve question-answer-url triplets.

Using this approach, we were able to generate 72k question-answer pairs (owing to time constraints, we were unable to generate QA pairs from all the GIFs).

Further Scope and Improvements

One could also consider a prompting-based approach for this synthetic data generation process. We did write a script using a 4-bit GGUF quantized version of LLaMa -7B-Instruct, using llama_cpp, and tried to run inference, however, the inference pipeline was slower than the one we have described above, and in a few cases the question-answer pairs were not right.

We also tried other approaches, such as, generating knowledge bases from the text descriptions and distilling (subject-predicate-object) pairs from them, which could potentially improve the reasoning abilities of LLMs. Thereere exist various standard algorithms for achieving this, such as the Multi Liaison Algorithm for extracting such knowledge triplets, that could be implemented for further enriching the dataset.

Unfortunately, we were unable to perfect this line of thought owing to time constraints. However, the diversity among the generated questions is quite sufficient to incorporate reasonable QA abilities in the subsequent models that are developed.

CLIP-Cut : a GIF Processing Pipeline

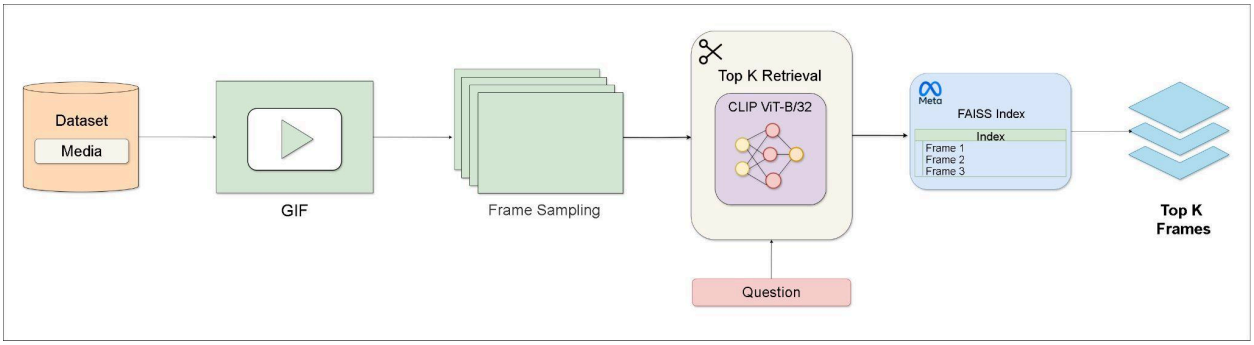
Why do we need a GIF Processing pipeline?

While it is true that the motion media that we are working with is not necessarily voluminous, it is important to realize that *computational constraints* essentially forbid working with the entire video, or set of frames to be more precise. It is important to mainly work with those frames that are relevant to the present context. One also needs to consider *efficiency* of the overall pipeline, which in itself is a crucial factor.

Further, cutting down on the number of frames *cleverly* can allow us to introduce other multimodal LLMs into the picture, at least for valuable inference on these frames, while not losing out on valuable information or including irrelevant frames.

These reasons motivated us to construct a fast *keyframe* retrieval pipeline.

Pipeline



We propose CLIP-Cut, an efficient pipeline for keyframe extraction from GIFs. CLIP-Cut is essentially based on two core pillars, mainly, CLIP and FAISS.

We first sample all the frames from the GIF url, and then, make use of the FAISS library to generate an faiss-index by grouping every 3 frames together. Then, the CLIP transformer is utilized to project the GIF frames and the text input, comprising the question and the description, to project the language and image features into the same space, and retrieving the most similar frames using an index search. The snippet below demonstrates the efficiency:

```
[43] %%time
keyframes = clip_cut.retrieve_images_from_gif(url_example, clipm, clipp, 'what is happening in the gif?', 3)

framerate 14
framecount 116
CPU times: user 1.31 s, sys: 268 ms, total: 1.58 s
Wall time: 17.7 s

[44] %%time
total_frames = clip_cut.get_gif_frames(url_example)

framerate 14
framecount 116
CPU times: user 409 ms, sys: 135 ms, total: 544 ms
Wall time: 16.7 s
```

Implementation

CLIP-Cut has been implemented with three auxiliary methods :

1. *get_gif_frames* : This method allows the retriever to make use of OpenCV and access the GIF frames from Tumblr. We sample all the frames for further processing.

2. *create_faiss_index* : As the name suggests, we first encode each image using the CLIP processor, and then make use of the faiss-gpu library to develop an Euclidean distance based vector index for supporting fast search and retrieval. We select L2 based indexing as it provides a reasonable tradeoff between speed and accuracy.

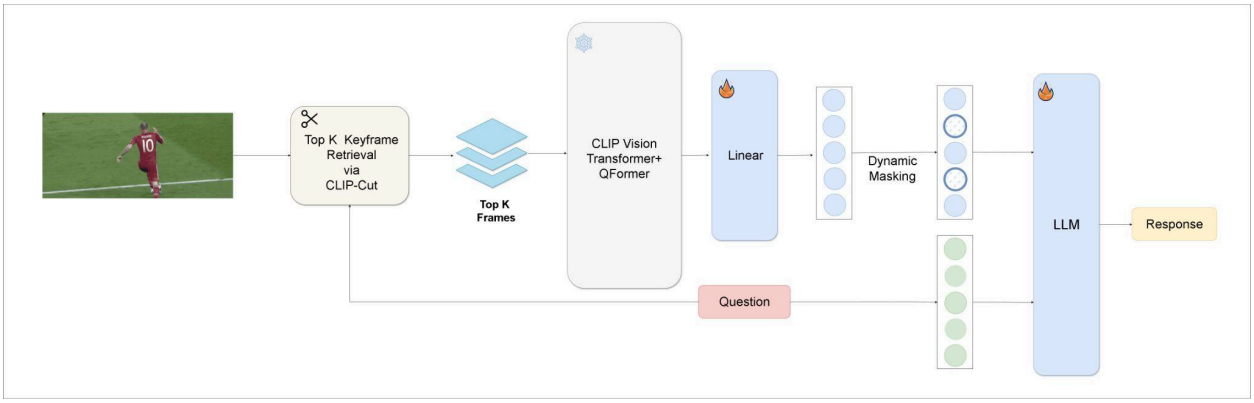
3. search_similar_frames : We encode the text using the CLIP model and effectively align the image and text modalities; the FAISS index allows for fast search and effective retrieval of the top k relevant frames.

4. Retrieve_image_from_gif : It combines the above mentioned methods and retrieves the most relevant k-frames using the index created earlier.

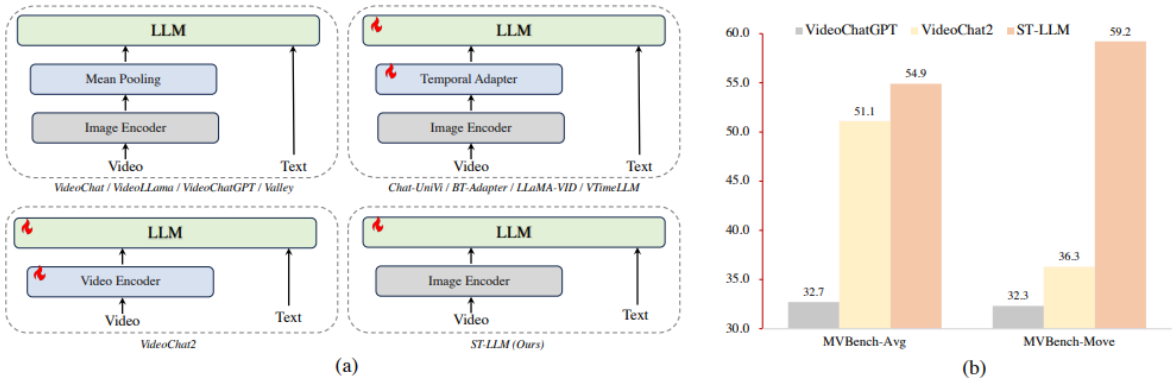
In such a way, we achieve high-speed keyframe extraction and retrieval. We now proceed to the various approaches that were implemented.

Approach 1 : Developing a Video Understanding LLM

Model description and architecture



A recent ECCV submission made by researchers at Tencent, [ST-LLM: Large Language Models Are Effective Temporal Learners] deliberated over the spatio-temporal reasoning abilities of LLMs without explicitly defining Temporal Adapters or carrying out extensive pre-training tasks. The results of the paper were quite commendable, with the model beating several existing video-large language models on several VideoQA benchmarks such as MSVD-QA, MVBench, MSVRTT-QA.



Also, this is one of the few approaches that does not incorporate fine-tuning of the base image encoder or insert any adaptive layers. In fact, the model closely resembles a video chat-bot model.

We took inspiration from this approach and tried to implement the key ideas from the paper in our own model, with a few key changes however :-

1. Firstly, owing to computational constraints, we had to limit ourselves to the *lighter versions* of models, i.e, the language model head that we used was GPT2, while the paper made use of Vicuna 7B. We also replaced EVA-CLIP with the base CLIP processor.

2. We also got rid of the *global residual branch* that the authors had introduced for handling the memory loss issue in lengthy videos, since that is not a concern for our use case.

3. It follows naturally, that the number of epochs and the size of the training dataset was also smaller in our implementation.

4. The dynamic masking implemented is a bit different than what is proposed in the paper; it is not the same type of masking.

Barring these points, special care has been taken to ensure the implementation is reasonable.

Our final architecture consists of the CLIP-Cut retriever that returns the top-k frames relevant to the question. These frames are then passed to a CLIP vision encoder, followed by the pre-trained Q-Former of BLIP2. We then use a linear layer to project the image features to the dimensions of the LLM input. The question is also tokenized, and we simply prepend the frame features before the question, and let our model predict the response.

Training and Performance

Barring the few changes mentioned above, we implemented the identical dual objective training procedure that coupled the new ***masked video modeling loss*** and the standard language modeling cross entropy loss.

First, we mask the image modality by selecting a masking rate and randomly sampling a masking matrix where the values are either 0 or 1 depending on the value generated using torch.rand.

The masked language modeling loss aims to improve the spatio-temporal understanding of LLMs while simultaneously using its semantic powers. During training, we randomly mask a certain sample of the frame tokens that have been generated. Using these masked tokens, we generate an output after a forward pass through the language head, and label it $f_{llm}(I)$ Then, we conduct an extra forward pass through the language model, and label this outcome as $f_{llm}(I')$

The masked video modeling loss is then defined as the squared sum error of values at the unmasked positions :

$$\mathcal{L}_{mvm} = \frac{1}{(1-\rho)KT} \sum_{i=1}^{T} \sum_{j=1}^{K} (v_{i,j}^{-1} - \hat{v}_{i,j}^{-1})^2, \quad (i,j) \notin M,$$

Results

Owing to computational constraints, we were able to train our model on a dataset of 1000 QA pairs for 3 epochs(~7 hours) on a single P100 GPU. Our results are shown below:

epoch	avg_loss	avg_perplexity
1	4.77	1.042
2	3.49	1.03086
3	3.49	1.03084

Owing to the extremely small corpus of training text, our model was not really able to learn anything, therefore, returns answers that correspond to the most frequent terms in the dataset, such as, a man, a woman, a girl etc. The value of avg_perplexity here reinforces this.

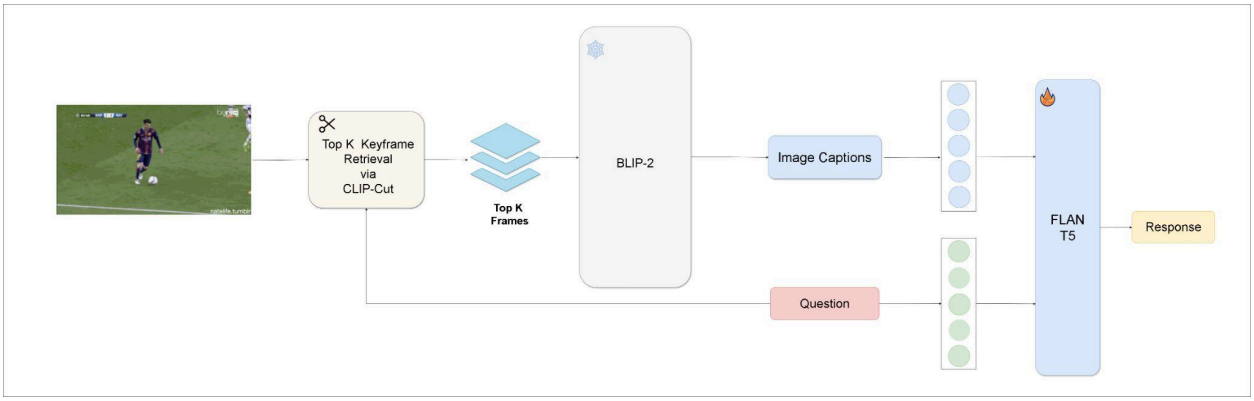
Inference on 500 QA pairs yielded the following scores:
BLEU(0.457), Accuracy(0.028).

Possible Improvements

Quite a few works that have been recently published in the realm of Video Understanding and have performed well on VideoQA have generally been pre-trained on a task that enhances their reasoning abilities and helps the model in aligning the language and video modalities. We could potentially implement such techniques to greatly improve model performance.

Approach 2 : Efficient GIF Captioning and QA

Model description and architecture



This is an approach that was inspired by the results in a paper that tackled the VQA task using visual descriptions, and the following insights that were derived from analyzing the dataset:

1. The number of frames per GIF is relatively small on average (40)
2. A GIF itself is akin to a moving snapshot; well-selected keyframes should be able to convey the same story to the users as compared to the entire set of frames.

This is where we hypothesized that the GIF can effectively be converted into a natural language context, and the task can be effectively converted to a natural language question-answering task. Considering the fact that we need robust and efficient pipelines, we decided to go ahead and implement this approach.

We made use of the Salesforce BLIP2 Model for image caption generation, and concatenated the descriptions to form our context. This was then coupled with the question-answer dataset generated earlier and used to fine-tune FLAN-T5 (large) along the lines of a question-answering task.

However, owing to computational constraints, this approach had to be discarded, as captioning for even 3 keyframes required a significant amount of VRAM by even the quantized BLIP model for image captioning. Training with only one keyframe caption serving as context led to extremely poor results.

Conclusion

We have tried to tackle the complex task of GIF QA by effectively resolving it layer by layer. We first addressed the task of QA pair generation using an LLM-based synthetic data generation method. We then presented CLIP-Cut, an efficient pipeline to retrieve keyframes relevant to the context, based on the CLIP Model and FAISS. We tackled the main task by considering two approaches; one based on incorporating Video/GIF understanding abilities inside LLMs, while the other was aimed at utilizing a fixed number of key-frames that can be used to generate text captions and effectively convert the problem into a text-to-text QA task.

Literature Survey

During our research we came across quite a few interesting works that could have been considered:

1. **Other Adapter-Based approaches** : As was briefly highlighted above, there do exist other adapter-based instruction tuning approaches, where the Image Encoder is also fine-tuned, and even spatio-temporal tokens are introduced in order to improve the model's reasoning capabilities.
    a. VideoLlaVa : Compressing image and video modalities into a shared feature space and then combining with

    b. VAMOS : Along with video-language alignment, it also incorporates captioning or action recognition models which are supplied as auxiliary text input to the model.

    c. LlaMa-VQA : Based on **MultiModal Masked Learning**. It carries out extensive pre-training tasks instead of the conventional answer prediction; it also carries out question prediction using the answer and video, and video prediction using the question and answer.

2. **MultiModal Reasoning** : Multimodal reasoning is a domain that is relatively new, where most existing approaches try to incorporate reasoning abilities into VLMs by construction of question-relevant rationales. These rationales are generated in a variety of ways, such as:
    a. Mixture of Rationale : Maintains a fixed set of triggering prompts, and generating VLM responses, or in other words, VLM thoughts, and considering only the ones relevant to the context, by using a similarity metric.

    b. II - MMR : Generates subject-relation-object or knowledge triplets, and uses a zero-shot answer generated by the VLM for guiding the reasoning process.

    c. Socratic Models : It considers an intersection between the cross modal abilities of VLMs and reasoning abilities of LLMs to break the problem down into sub-parts, and consider an 'expert' for each task.

References

[1] ST-LLM: Large Language Models Are Effective Temporal Learners
[2] Generative AI for Synthetic Data Generation: Methods, Challenges and the Future
[3] Synthetic data: save money, time and carbon with open source
[4] KeyVideoLLM: Towards Large-scale Video Keyframe Selection
[5] A Video Is Worth 4096 Tokens: Verbalize Videos To Understand Them In Zero Shot
[6] Question-Instructed Visual Descriptions for Zero-Shot Video Question Answering
[7] Socratic Models: Composing Zero-Shot Multimodal Reasoning with Language
[8] Awesome-LLMs-for-Video-Understanding
[9] Video Understanding with Large Language Models: A Survey
[10] II-MMR: Identifying and Improving Multi-modal Multi-hop Reasoning in Visual Question Answering
[11] Mixture of Rationale: Multi-Modal Reasoning Mixture for Visual Question Answering
[12] Open Ended Medical VQA through Prefix Tuning of Large Language Models