Youtube Teller: a Video StoryTelling model to extract people's daily activity

Chen Wei (cwei24), Yuan Zang (yzang6), Yunhao Luo (yluo73)

1. Introduction

Video Description/Storytelling is a complicated task which involves automatically captioning a video by understanding the action and event in the video which can help in the retrieval of the video efficiently through text. For this final project, we are trying to solve a video-caption problem which specifically focuses on video captioning tasks on Youtube video.

1.1 DataSet

YouTubeClips dataset (Figure 1) is a collection of 2,089 various languages video clips with 85K English description, as well as descriptions in over a dozen languages. These short video clips are usually less than 10 seconds long.

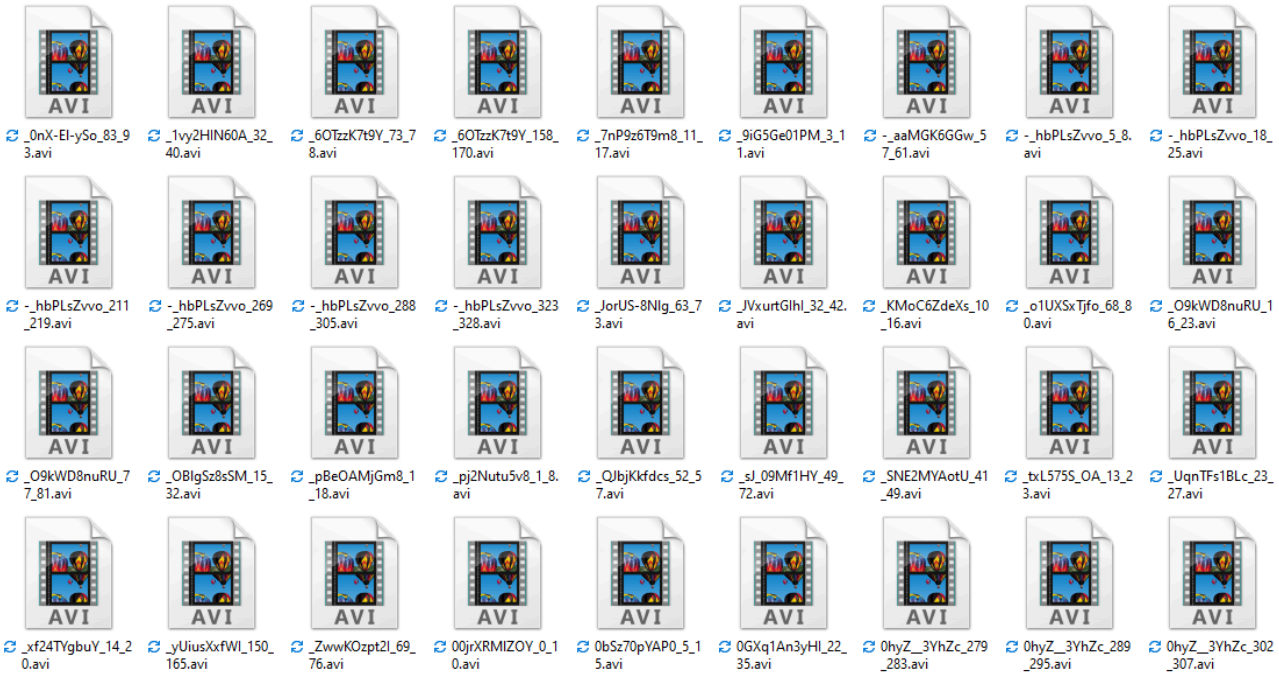

Figure 1 YouTuberClips Dataset

These clips' names are in the format: {id}_{start time}_{end time}. For example, `-4wsuPCjDBc_5_15` means Video Id: `-4wsuPCjDBc` starts from 5th seconds to 10th

seconds. The Description of YoutuberClips (Figure 2) is a batch of short sentences describing the video corresponding to video Id.

```
-4wsuPCjDBc_5_15 a squirrel is eating a peanut in it's shell
-4wsuPCjDBc_5_15 a chipmunk is eating
-4wsuPCjDBc_5_15 a chipmunk is eating a peanut
-4wsuPCjDBc_5_15 a chipmunk is eating a nut
-4wsuPCjDBc_5_15 a squirrel is eating a nut
-4wsuPCjDBc_5_15 a squirrel is eating a whole peanut
-4wsuPCjDBc_5_15 a squirrel is eating a peanut
...
```

Figure 2 Description text of YoutuberClips Dataset

For example, in the first line in Figure 2, `-4wsuPCjDBc_5_15` is the video id and "a squirrel eating a peanut in its shell" is the description. Since the captions all have similar meanings, this task is more like a video caption task instead of a storytelling one.

2. Methodology

This task can be visualized as a sequence-to-sequence generation task, which translates video clips to a sentence. The main idea inspired by the work ClipCap: CLIP Prefix for Image Captioning, which is to use a light model to merge the GPT-2 model to this specific translation task without requiring the time consuming training process of a large language model such as GPT-2.

2.1 Preprocessing

To start, we used the cv2 package to transform the video into a sequence of images. We set the frame rate to be 1s per image frame, as the videos in our dataset are generally 5 or 10 seconds in length. After several tests, the frame rate that 1s per image frame is adequate for this translation task, and also economical since it has the lower number of images generated to be fed into our model. Considering the large size of YoutuberClips dataset, we limited the number of images that can be extracted from each video. In this process, we keep a good balance of the trade off between the number of images we extracted and the quality of the model performance.

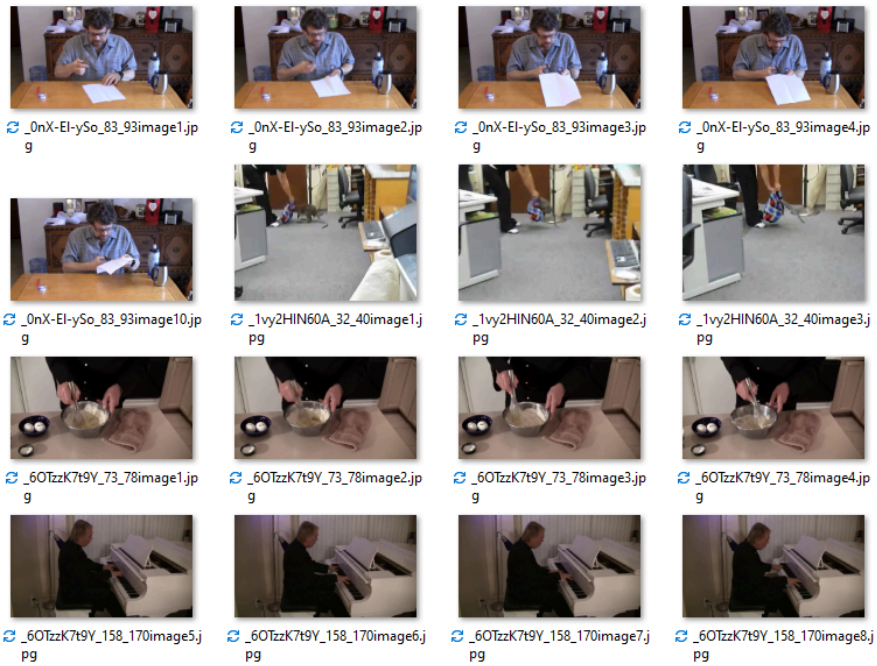

Figure 3

In the next step, we used 3 different strategies to process our image frames, so that it can be put into the model. Below are the 3 ways of input:

1. Turn our video into random single image frames.
2. Turn our video into mean image frames. That means the mean of every image frame corresponding to that video.
3. Turn our video into sequential image frames. And we will use positional encoding afterwards in our transformer part to deal with it.

We treat our video in these 3 ways in order to compare the model translation effectiveness under different video information levels. The first way can be considered as the a-glance level of video, which just randomly captures a tiny part/frame of the video. The second way is the ambiguous level overview method, which considers the whole video without thinking about the details. The third level method captures more details that existed in the whole video, hoping to improve the quality of the translation.

The reasons we believe that we could choose a single image frame is that this is a video caption project rather than a storytelling one, the generated captions are all similar. We believe that a single image frame could contain adequate information for this task, so we use this as a baseline to see how our model performs.

2.2 Models structure

The model is inspired by the idea of ClipCap: CLIP Prefix for Image Captioning. Their model structure is Figure 4. We removed the pretrained COCO dataset's weight and applied our own creativity (positional encoding) on the model.We make this choice, as we believe using prefix training to capture the video/image feature is a very lighted way of training and can achieve good results without the need of managing to "train too much". And instead of passing the image, we pass the preprocess video (image frames) into the CLIP model.

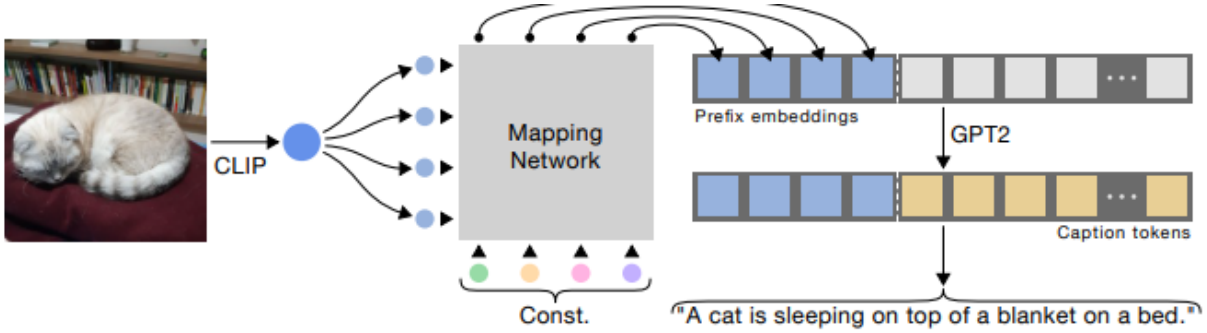

Figure 2. Overview of our transformer-based architecture, enabling the generation of meaningful captions while both CLIP and the language model, GPT-2, are frozen. To extract a fixed length prefix, we train a lightweight transformer-based mapping network from the CLIP embedding space and a learned constant to GPT-2. At inference, we employ GPT-2 to generate the caption given the prefix embeddings. We also suggest a MLP-based architecture, refer to Sec. 3 for more details.

Figure 4: ClipCap model structure

In our model (Figure 5), we applied the Transformer-encoder Mapper model structure to encode image information into embedding space, so that the GPT-2 is suitable for this special task without training. The CLIP model is designed to impose a shared representation for both images and texts. After training over a vast number of images and textual descriptions using the cross entropy loss, the visual feature of CLIP and textual representations of GPT-2 are well correlated. Therefore, the fine-tuned Mapper structure allows prefix embeddings to capture the visual information, and effectively generate the suitable input for the GPT-2 model to correctly predict.

Our modified model structure looks like:

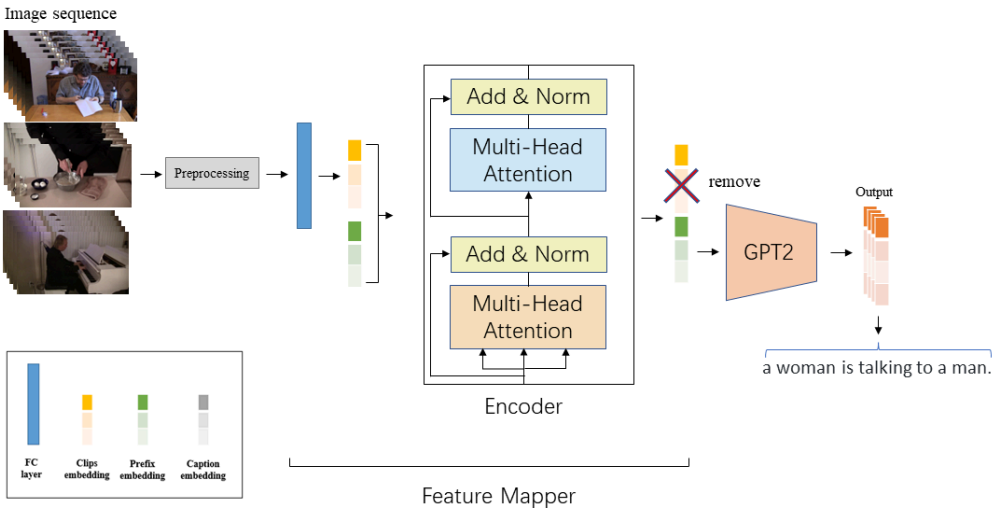

Figure 5: Our model's structure

The whole process of this model is shown above. We preprocess videos into image embeddings. And then we concatenate image embedding with prefix embeddings. Then we feed such combinations into the Feature Mapper model to allow prefix embeddings gaining enough visual information. Then, we only maintain the prefix embedding parts and feed them into the GPT-2 model to generate the final predictions.

2.3 Evaluation metrics

We plan to test our video caption model on the test dataset of uncaptioned videos to generate their captions. We evaluate the performance of our model on the similarity of the generated sentences and standard answers. Specifically, we adopt 4 metrics to evaluate the accuracy of our captions, namely, BLEU, CIDEr, METEOR, and Rouge.

The baseline model (Vision Transformer) can achieve 68.4 1-gram BLEU score and 50.7 5-gram BLEU score. We hope to improve the performance in some specific subjects, to achieve higher BLEU scores than the baseline model.

3. Result

Video	1	2	3	4
Video ID	0bSz70pYAP0_5_15	-vg3vR86fu0_1_6	60x_yxy7Sfw_1_7	9HDUADeA2xg_3_31
Sample Image Frame				
All Image Frame	Plane video image frame	Man riding image frame	Man watch video of woman image frame	Doggie playing image frame
Sample Ground Truth Sentence	an airplane is flying in a wide circular pattern.	a guy on a bicycle who tries to jump his bike on a wooden ramp.	a man seated is watching and admiring the image of a woman on the screen of his laptop.	a puppy is playing with a ball.
All Ground Truth Sentences	Plane video captions	Man riding captions	Man watch video of woman captions	Doggie playing captions
Single	a flying airplane flying over a lake.	a man is riding a motorcycle.	a woman is dancing.	a baby is walking on a rug.
Mean	a plane is flying.	a man is driving a car.	a man is talking on a computer.	a cat is playing with a toy.
Sequential	a plane is flying.	a man is riding a motorcycle.	a man is talking to a woman.	a cat is playing with a toy.

Figure 6.1: Our model's generation result

The single, mean, sequential means different preprocessing methods and how image frames are selected and passed to the model.

Method	BLEU	CIDEr	METEOR	ROUGE
Sequence	0.507	0.549	0.297	0.641
Mean	0.520	0.584	0.302	0.647
Single	0.487	0.465	0.281	0.630

Figure 6.2: Our model's evaluation results

3.1 Analysis

For all of the above methods, the method Mean, sequence, and single have the best, the second, and the third performance. This result shows that capturing a sequence image of video and giving a glance of the video, the average input of a video allows the model to more effectively extract the visual information. This result is surprising because an image sequence should provide more details of a video. Therefore, this result implies the way that processes video to a sequence of images is not effective on this dataset or maybe it requires a finetuning of parameters.

Meanwhile, compared with the ground truth, all the results are not ideal in detecting types of objects. These predictions are distinguishable for humans from the ground truth sentence. Therefore, a more effective model or a better fine tuning method of capturing video's visual information is needed.

4. Challenges

4.1 Preprocess the dataset

Our original plan is to make a cooking guide model - a video storytelling model built on the MMAC dataset. However, it is too hard to preprocess. To be specific, in the doc of MMAC, it says we need to download the CMU-MMAC dataset in advance. And since we choose pizza categories, there are 35*3 pizza files to be downloaded, which make it hard to download without scripts and is too big to process. And the vid attribute in caption_train/val/test dataset is hard to find the corresponding elements. So preprocessing this dataset just requires a lot of effort.

So to make our lives easier, we choose to switch our dataset. At first, we wanted to switch our dataset to Visual Storytelling Dataset VIST. But it seems to require access and is very big. And then we find many other dataset which are not accessible. In the end, we found this dataset which seems ok for us to preprocess. It contains Microsoft Research Video Description Corpus and Youtube Clips.

4.2 Model structure

There are so many choices for us to implement our model structure (although none of them is easy). It is really hard to make decisions and implement them all from scratch in such a short time. So we choose to modify existing model structure and try to twist it if we have time.

Some of the choices are:

1. Like GLACNet

Preprocessing video to image sequence and using ResNet-152 image feature extractor, then pass it to Bi-LSTM Encoder. Finally, use an LSTM decoder to generate sentences. Or maybe we could change Bi-LSTM to transformer or Bi-Transformer.

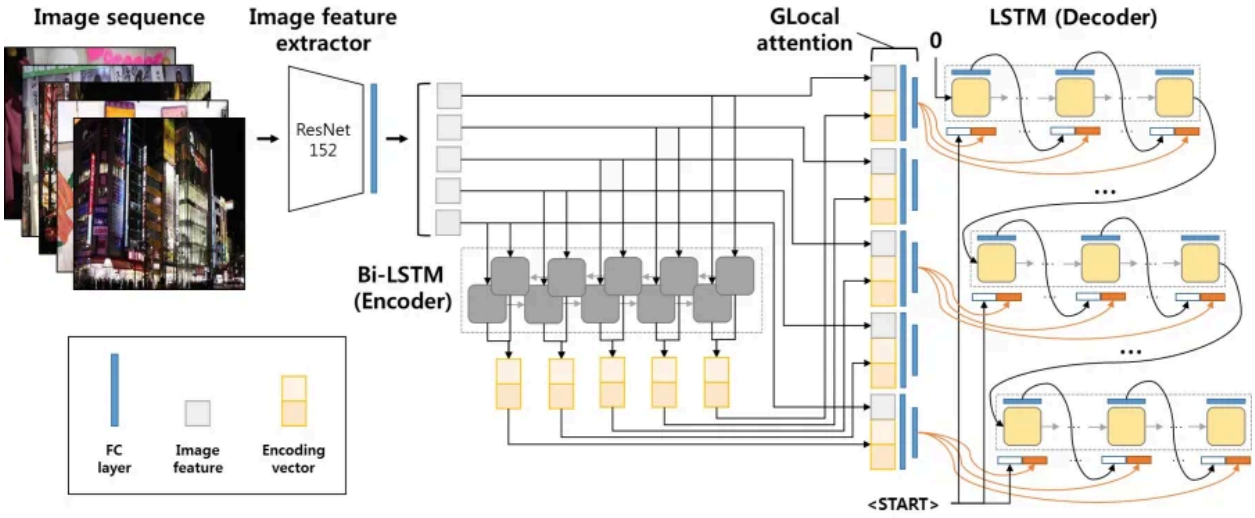

Figure 7: GLACNet model's structure

2. Like our original plan
   Pass the video directly to a 3d-CNN based video encoder to get C3D features and implement a Transformer-based encoder as well. Then we combine the information from 2 encoders and pass it to LSTM or Transformer-decoders
3. Turn the video into image frames, and then use a visual feature extractor to encode images. Then we use a Transformer Encoder to encode the sequential information of images and map the image embedding into the text representation space. In the end, we use a language model to decode the encoding into captions.

5. Reflection

Our project ultimately turned out to be ok and our model works as expected. It can generate captions that are acceptable and coherent although not being perfect.

Our biggest takeaway is that we understand the transformer model structure and how it can be used to mapping modalities in different domains better. We also learn to read the doc of some packages better.

If we have more time, we could implement grid search on the number of image frames we will use to represent a video and the frame_rate parameters when extracting image frames from videos to make the result better. Also, we could try different model structures and do more ablation studies. For example, using C3D to extract video structure might perform better than our current model, as it preserves positional information better.