Video Summarization

Madhav Chavda, Prerak Raja, Ratnesh Shah, Varad Bhogayata
School Of Engineering & Applied Science, Ahmedabad University.

***Abstract*—For most people, watching a brief video and describing what happened (in words) is an easy task. For machines, extracting the meaning from video pixels and generating a sentence description is a very complex problem. The goal of this project is to develop model that can automatically generate natural language (NL) descriptions for events in videos. Our LSTM model is trained on video-sentence pairs and learns to associate a sequence of video frames to a sequence of words in order to generate a description of the event in the video clip.**

***Keywords*—Convolution Neural Network, Video Captioning, LSTM, Sequence to Sequence, VGG16, METEOR Evaluation**

I. Introduction

The ability to describe videos in natural language enables many important applications such as human-robot interaction, describing movies to the blind, multimedia search and also for video surveillance for incident reporting. There has been an increase in work on natural language image description and also a growing interest in video description in the past year.In this report we have implemented the model described in the paper[1]. While image description handles a variable length output sequence of words, video description also has to handle a variable length input sequence of frames. The problem of generating descriptions in open domain videos is difficult not just due to the diverse set of objects, scenes, actions, and their attributes, but also because it is hard to determine the salient content and describe the event appropriately in context.

Our model consist of two main parts Convolution Neural Network (CNN) and Sequence to Sequence (Seq2Seq). As per CNN part is considered, we are feeding in frame by frame to our VGG-16 model which is pre-trained on imagenet dataset.In output we are getting features which in turn goes to our Seq2Seq model. Our Seq2Seq model comprises of two stacked Long Short Term Memory (LSTM)- first part (Encoding) is for encoding the frames one by one, taking as input the output of a Convolutional Neural Network (CNN) applied to each input frames intensity values. Once all the frames are read, the second part (Decoding) of stacked LSTM generates sentence word by word.The encoding and decoding of the frame and word representations are learned jointly from a parallel corpus.

II. Dataset

Microsoft Video Description corpus (MSVD) is based on web clips with short human annotated sentences. MSVD is a collection of Youtube clips collected on Mechanical Turk by requesting workers to pick short clips depicting a single activity. The video were then annotated by humans in single sentence. The original corpus consist of multi-lingual description but in our model during training we have only used English description.

	MSVD
#-sentences	80,827
#-tokens	567,874
vocab	12,594
#-videos	1,970
avg. length	10.2s
#-sents per video	≈41

We have done just basic preprocessing on this corpus. We have converted all characters to lower case, removed punctuation and tokenized the sentences.

III. Approach

There are basically two steps involved in our project. First one is feature extraction(To extract features of frames of given video) and Seq2Seq LSTM model(Generate caption word by word).

A. *Feature Extraction*

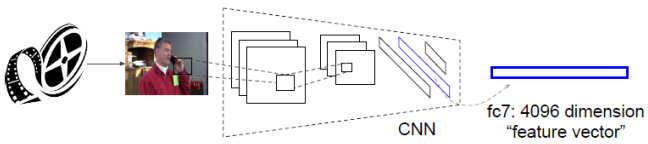

Figure 1: Feature Extraction at fc7 layer

For the purpose of feature extraction we have used the VGG16 - pre trained model. The video is re-sized to same dimension of 224x224x3. The video is also sub sampled to 50 sample for each video. This is because, as we have a short length of video consisting of only one action, its very likely that the many adjacent frames would have amount of information with very little or no change. We use the features obtained from fc7 layer and store it in a numpy file for each video. And as we have sub-sampled the video to 50 samples, the features will be 50x4096.

B. *LSTM Sequence to Sequence Model*

Learn a representation of a sequence of frames in order to decode it into a sentence that describes the event in the video. The top LSTM layer models visual feature inputs.The second LSTM layer models language given the text input and the hidden representation of the video sequence.
There are two part namely, Training and Testing. For training

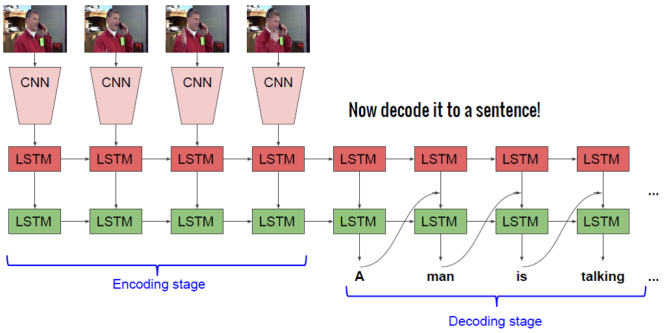

Figure 2: Sequence to Sequence Model

purpose, we first choose English language caption. We separate out the Video-ID nad Video caption. After this we build the vocabulary. We generate two numpy files namely WordToIndex and IndexToWord(if count of word is greater than predefined threshold, then store it in vocabulary). And this is how we generate vocabulary. We separate out 90% of the data for training purpose and 10% of the data for testing purpose.The next step is to initialize a model. We initialize different parameters such as image dimension, number of hidden layers(256 in our case), number of words, batch size, number of LSTM steps(10 in our case). After we have initialized the model, we have to build the model. For Model building we first read frames(phase 1). We than generate captions(phase 2), and we than calculate loss function. Now we have everything ready for training the model and thus we train the model.

Now we have to test the model. We take the test video set data. Load the IndexToWord file. Initialization of different parameters such as image dimension, number of hidden layers(256 in our case), number of words, batch size, number of LSTM steps(10 in our case). Generation of captions(phase 2 - select word with maximum probability) after reading the frames(phase 1).

IV. Results

A. *Results Obtained*

- Ground Truth: A man is pouring oil into a frying pan.
  - At 100th epoch : man is cooking
  - At 500th epoch : man is pouring some
  - At 900th epoch : man is pouring oil into a
- Ground Truth: A man is chopping an onion.
  - At 100th epoch : person is slicing
  - At 500th epoch : man is slicing
  - At 900th epoch : man is slicing

B. *Difficulties faced*

- In generating features we needed to extract 80 frames but since the dataset was large all the 80 frames were not able to load in memory. So we decided to extract 50 frames as the system was able to load it.
- In the sentences which are generated we notice that the sentences are not complete. This may be because of the words not in vocabulary due to threshold.
- Training time required for 1000 epochs was around five hours.
- Several inaccuracies in the generated sentence.

V. Evaluation

There many methods for the evaluation criterion namely BLEU, METEOR, ROUGE-L, CIDEr etc. However, the metric we used here is METEOR metric.

Video/Epochs	100^{th}	500^{th}	900^{th}
Video-1	30.57	43.16	64.34
Video-2	19.30	41.92	41.92
Video-3	51.98	30.34	37.30
Video-4	53.04	53.04	53.04

- METEOR Metric
  - It is basically used to evaluate machine translation results
  - Computed based on given hypothesis sentence and set of candidate reference sentence
  - It compares exact tokens, stemmed tokens, paraphrase matches, and semantically similar matches using wordnetsynonyms
- Other Methods(BELU, ROUGE-L, CIDEr)
  - Meteor outperforms these methods when no. of references are small
  - We have taken only single reference sentence. Therefore, we use METEOR metric.

References

[1] arXiv:1505.00487 [cs.CV]
[2] https://www.youtube.com/watch?v=iX5V1WpxxkY
[3] https://vsubhashini.github.io/s2vt.html
[4] https://medium.com/towards-data-science/sequence-to-sequence-model-introduction-and-concepts-44d9b41cd42d
[5] http://cs231n.stanford.edu/slides/2017/cs231n2017lecture10.pdf
[6] https://www.tensorflow.org/tutorials/seq2seq