CODS-SUMMER PROJECT

Scene-Describer
Finding timestamps of a scene in a video given a textual description

Contributor 1	Vishal R
Contributor 2	Royston E Tauro
Contributor 3	Mahim Dashora
Contributor 4	Nityam Churamani
Mentor 1	Rishith Bhowmick
Mentor 2	Sai Eshwar

Contents

1. Problem Statement

Given a textual description of a scene in a video,the timestamp of that scene is returned to the user

1.1 Requirements

There are a few requirements to run our project

- Python3.7
- LuaJit
- NodeJs

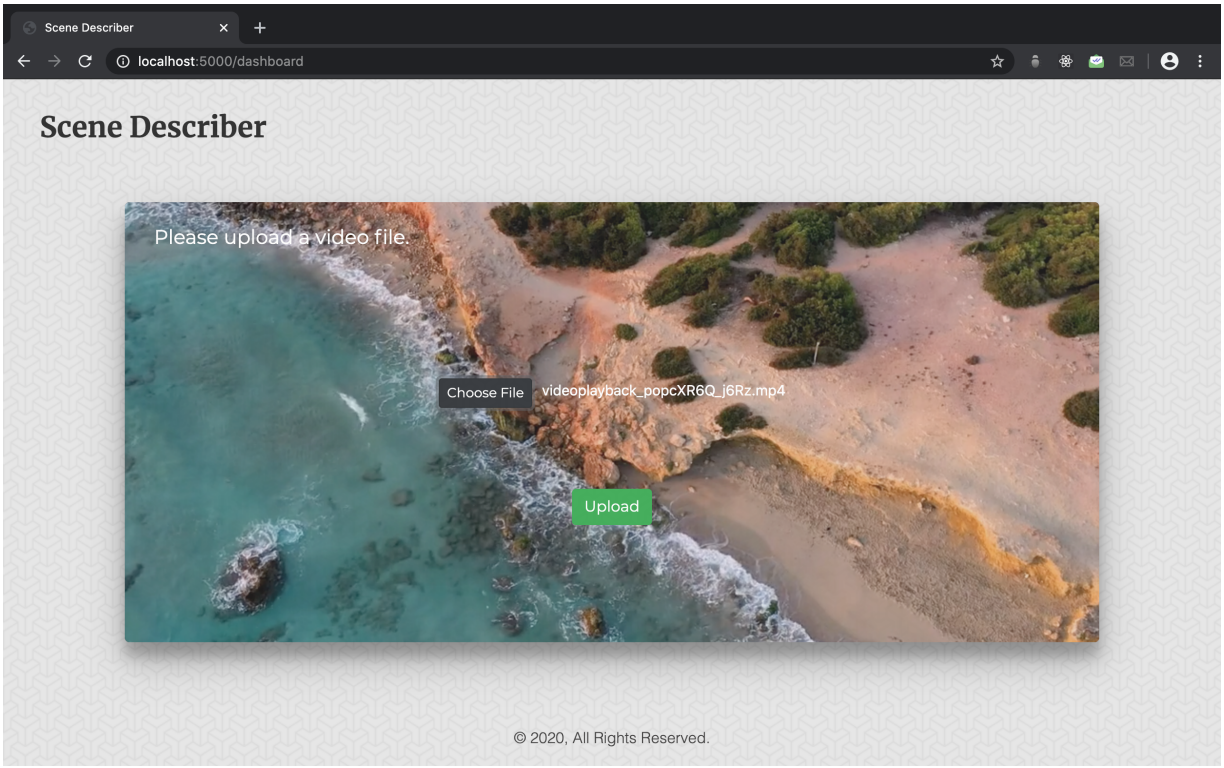


Figure 1.1: Our Website

2. Introduction

Finding required scenes in a video might have various applications,from finding your favourite scene in a sit-com to removing inappropriate scenes in a movie for kids.
We propose a simple prototype model to achieve this goal.We use applications from Image Captioning, Frame Extraction and Basic NLP methods for the completion of our project.
We also provide a Graphical User Interface in the form of the website for the user to have easy access.Our Interface is shown in Fig 1.1

3. Our Model

Our model is divided into three stages,as seen in the block diagram 3.1.

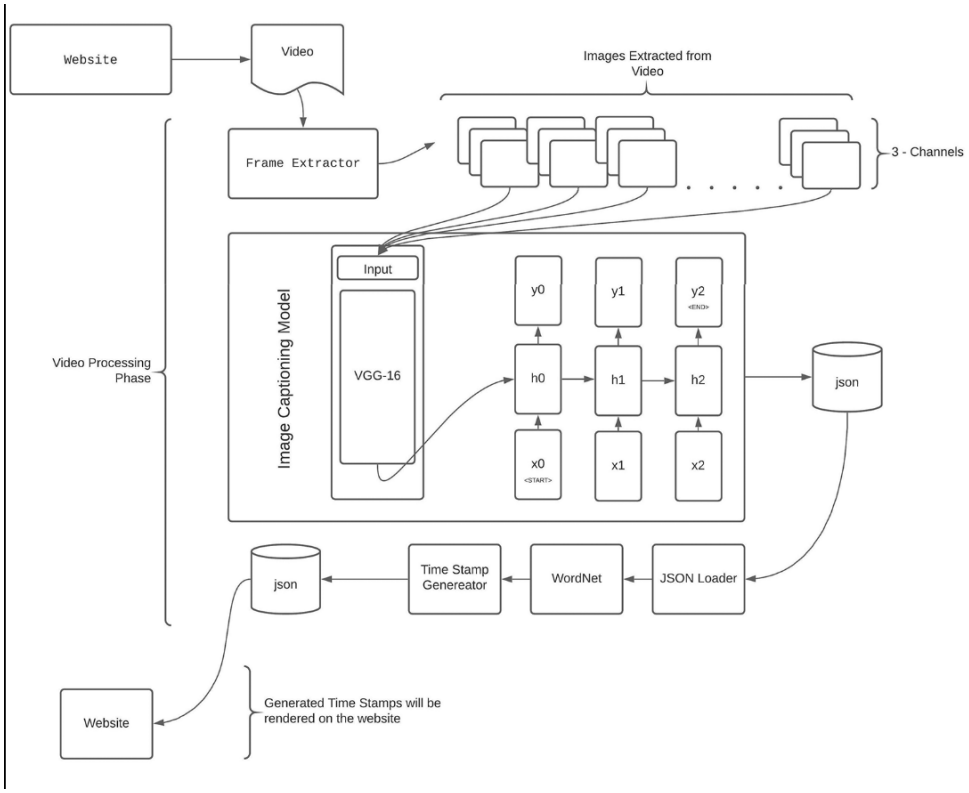


Figure 3.1: Our Model

3.1 Frame Extraction

We experimented with two techniques of frame extraction. One was a trivial extraction techniques utilizing a simple while loop and open-cv to extract the images.A Drawback of this techniques was that though it worked for smaller videos,there were ram issues when a video of 15 minutes or more was used.
To solve this problem we use the frame extraction code provided by [1] which uses the multiprocessing library provided by python to extract the images faster. The frames are extracted at a rate of 0.1 fps for faster captioning

3.2 Image Captioning

We then caption each extracted frame and store the generated captions in a JSON file.
Captioning was experimented with different techniques like Encoder-Decoder [2] , Attention [3] and Adversarial [4], but due to limited resources we were unable to generate generalized results. To counter this problem, we decided to experiment with pre-trained models, we then shortlisted our model to NeuralTalk2 [5] by the famous Andrej Karpathy and his team in DeepAI. NeuralTalk2 [5] is a simple Encoder-Decoder Model [2] written in LuaJit. Its basic structure is shown in 3.1.

3.3 Timestamp Generation

Once the captions are stored in the JSON file, the inputted textual description and the captions generated undergo basic NLP preprocessing like lemmatization.
The preprocessed description ($\vec{\tau}$) and the captions generated ($\vec{c}$) are matched using cosine-similarity (θ)

$$\cos\theta = \frac{\vec{\tau}\cdot\vec{c}_i}{||\vec{\tau}||||\vec{c}_i||}$$

where $\vec{\tau}$ is the word vector of the lemmatized textual description, $\vec{c}$ is the set of caption vectors of all the captions where each caption vector is a word vector of that caption represented by $\vec{c}_i$ and i is the iterator over each caption. The caption with the maximum similarity is selected and the rest are provided as options provided they beat a threshold of 10.0.
A problem we faced was that our model took in useless text for the base of predicting the timestamp. For Example:- If $\vec{\tau}$ is $\{'a','picture','of','a','pizza'\}$, then the model would give the timestamp of $\{'a','picture','of','a','mountain'\}$ as the output because it gave emphasis on the text 'a picture of'

$$\vec{\tau} = \{\tau_1, \tau_2, \hat{\tau_3}, \hat{\tau_4}, \tau_5, ..., \tau_m\}$$

$$\vec{\tau} = \{\tau_1, \tau_2, \underbrace{\tau_3, \tau_3, \tau_3....}_{n\ times}, \underbrace{\tau_4, \tau_4, \tau_4....}_{n\ times}, \tau_5, ..., \tau_m\}$$

$$\cos\theta = \frac{\vec{\tau}\cdot\vec{c}_i}{||\vec{\tau}||||\vec{c}_i||}$$

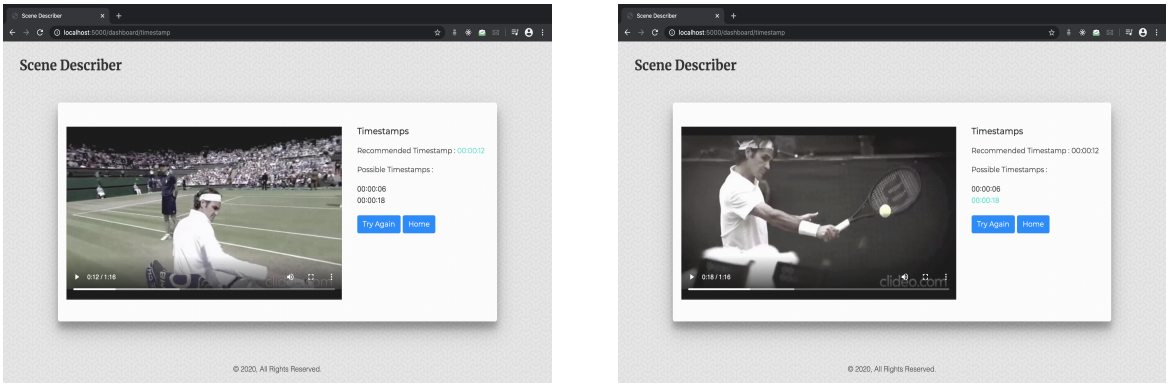
To solve this problem another feature was added called weighted similarity.It works something like this,if the user enters a textual description ($\vec{\tau}$) with some of the words in ALL CAPS ($\hat{\tau}$), then emphasis is given to the words with caps by duplicating those words n times, this prevents obvious errors of other useless text.

4. Results

We tested it out on a video with transitions of tennis, baseball and that of a train, the results where quite satisfying.
Since this model predicts timestamps, we cannot set a defined evaluation metric to it, therefore we just judge them based on their relevance to the given textual description Three different descriptions were given:

- A picture of a PERSON playing TENNIS
- A PITCHER throwing a ball in a BASEBALL game
- A TRAIN stopped at a railway STATION

The main timestamp predicted in Fig 4.1a is not of a a tennis player playing the game but the model did successfully associate the key words TENNIS and PERSON to it but Fig 4.1b does get predicted successfully.
The timestamps predicted in the rest of the captions were correct and therefore we can say our model did a successful job.



(a) Timestamp 1 (b) Timestamp 2

Figure 4.1: A picture of a PERSON playing TENNIS

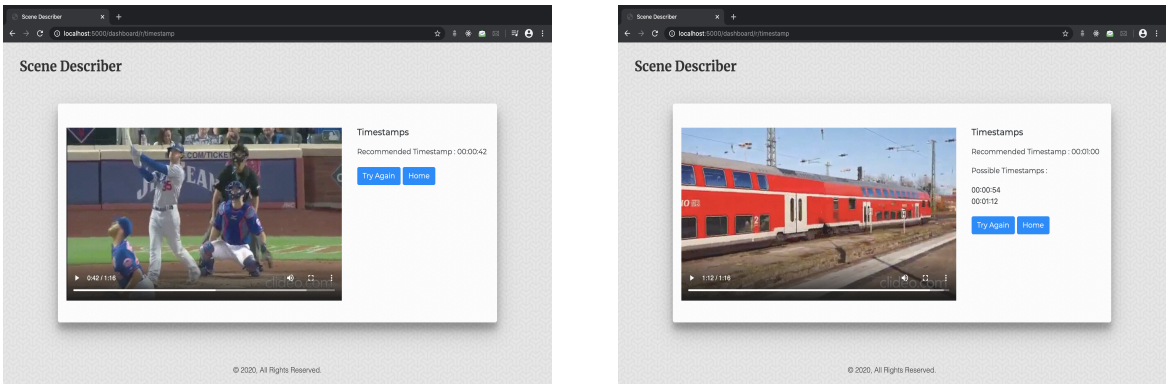


Figure 4.2: Other timestamp results

5. Future Scope

- **Condensed Movies:Story Based Retrieval with Contextual Embeddings**[6]:-
  This was a paper published by the University of Oxford, which understands scenes from sitcoms and/or movies and finds that scene in a movie or sit-com by understanding the relationships between characters.
- **Intelligent Frame Extraction**:-
  During Frame Extraction we might miss out essential scenes because of multiple frames per second, to resolve this frame we plan to utilize [7] and [8] where shots are segmented using HSV-color and RLBP-texture features.
- **Website Deployment**:-
  We plan to host the website on a server once all the enhancements are done
- **Speed and RAM Issue**:- We plan to reduce the overload on smaller RAM's which causes incorrect or no results to be produced and thereby reduce the speed at which this process takes place

6. Conclusions

Our project Scene-Describer can be used in a lot of places, from removing R-rated scenes in a movie or sitcom , to finding your favourite scene in a sit-com or movie.We utilize a lot of innovative topics from the Computer Vision and NLP field like Image Captioning. We produce

satisfactory results for a prototype model and we hope to continue and deploy this project in the long run.
Thank you to the CODS team for giving this fantastic idea for the project and for the mentors for helping us out in this project.

References

[1] Hayden Faulkner's medium article on video frame extraction https://medium.com/@haydenfaulkner/extracting-frames-fast-from-a-video-using-opencv-and-python-73b9b7dc9661

[2] O. Vinyals, A. Toshev, S. Bengio and D. Erhan, "Show and tell: A neural image caption generator," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 3156-3164, doi: 10.1109/CVPR.2015.7298935

[3] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. 2015. Show, attend and tell: neural image caption generation with visual attention. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15). JMLR.org, 2048–2057.

[4] R. Shetty, M. Rohrbach, L. A. Hendricks, M. Fritz and B. Schiele, "Speaking the Same Language: Matching Machine to Human Captions by Adversarial Training," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 4155-4164, doi: 10.1109/ICCV.2017.445.

[5] DeepAI's Image captioning model https://github.com/karpathy/neuraltalk2

[6] M. Bain, A. Nagrani, A. Brown, A. Zisserman
Condensed Movies: Story Based Retrieval with Contextual Embeddings

[7] Haq, Ijaz & Muhammad, Khan & Hussain, Tanveer & Kwon, Soonil & Maliyaem, Maleerat & Baik, Sung & Lee, Mi. (2019). Movie scene segmentation using object detection and set theory. International Journal of Distributed Sensor Networks. 15. 155014771984527. 10.1177/1550147719845277.

[8] Sajjad, Muhammad & Ullah, Amin & Ahmad, Jamil & Abbas, Naveed & Rho, Seungmin & Baik, Sung. (2017). Integrating salient colors with rotational invariant texture features for image representation in retrieval systems. Multimedia Tools and Applications. 10.1007/s11042-017-5010-5.