A Project Report

On

**Movie Semantic Search**

BY

**T.Abhinavasai Nikhil Sai Reddy.TN**

**19XJ1A0581 19XJ1A0544**

Under the supervision of

**Dr.Rajesh Kumar Tavva**

**SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF**

**PR 402: Fourth Year Project**

cid:image003.png@01D6594A.2604D070

**ECOLE CENTRALE SCHOOL OF ENGINEERING**

**HYDERABAD**

**(June 2022)**

ACKNOWLEDGMENTS

We are very much thankful to the **MAHINDRA UNIVERSITY** for giving us the opportunity to undertake our Fourth Year Project. We would also like to take the opportunity to thank and express our deep sense of gratitude to **Shri Arya Kumar Bhattacharya**, HoD of Computer Science Engineering.

We would also like to convey our heartiest thanks to **Dr Rajesh Kumar Tavva**, Assistant Professor, Department of Computer Science Engineering for providing valuable guidance, advice, constructive suggestions, positive and supportive attitude at all stages of the project with continuous encouragement, without which it would have not been possible to complete the project.

We hope that we can build upon the experience and knowledge that we have gained during the project and make it valuable for our coming future.

cid:image003.png@01D6594A.2604D070

**Ecole Centrale School of Engineering**

**Hyderabad**

Certificate

This is to certify that the project report entitled “**Movie Semantic Search”** submitted by Mr **T.Abhinavasai(HT No:19XJ1A0581), Nikhil Sai Reddy TN(HT No:19XJ1A0544)** in partial fulfillment of the requirements of the course PR 301, Project Course, embodies the work done by him/her under my supervision and guidance.

**Dr.Rajesh Kumar Tavva**

**Signature:**

Ecole Centrale School of Engineering, Hyderabad.

Date:

ABSTRACT

Movie data has a prominent role in the exponential growth of multimedia data over the Internet.Our project Movie Semantic Search is an approach to build a software or a tool to move to the semantic part of the movie rather than moving with respective to time. We use Computer Vision to develop this tool and in particular we used OpenCV and Natural Language Processing packages to develop the tool, we have total of 4 stages in this project; first, we break the movie into frame and group into shots using different ML libraries and OpenCV, second we take these shots and group them into scenes and obtain metadata as much as possible, third we integrate all the work and develop a UI tool for the user(just like a media player but with more features).In this report we have given the full description of the process followed and the problems we have faced and at last the results obtained.

CONTENTS

Title page………………………………………………………… …….1

Acknowledgements……………………………………………………..2

Certificate…………………………………………………………….....3

Abstract………………………………………………………………....4

1.Introduction…………………………………………….……… ….....6

2.Problem Definition.…………………………………………………..7

3.Background and Related work ……………………………………….7

4.Implementation……………………………………………………....12

5.Results……………………………………………………………….15

6.Conclusion…………………………………………………………...16

7.References…………………………………………………………….17

1.Introduction

Our project Movie Semantic Search is a new generation idea which enables the ease of movie analysis by experts and also gives a new enhanced experience to the users, our main idea is to move to the semantic parts of the movie rather than to different time, semantic part means to the different scenes in the movie (which are marked with respective timestamp) but are tagged with different data such as genre, characters played by actors, location, the type of shots etc.

We first break the movie into frames and group them into shots and then into scenes to extract the data for tagging the scenes and also obtain the timestamps from the script of the movie which we get on the internet(scripts have different conventions for a scene, dialogues etc..) and we match the script and the movie subtitles using python Regex which can be used to match the scene boundaries using subtitles file and script file and then can obtain the timestamp of the scene and similarly the tags which are required to be tagged for each shot and scene can be done using NLP packages. After all the tagging and grouping into scenes, we then develop a UI tool for users to access, we also are using a database for analysis of the movie (using NEO4J).

Our project will bring out a new way of movie traversal that will be much more useful for movie analysts, these are powered by the graph database and thus querying becomes much simpler for users to find different relationships between different entities as movies play a prominent role in our society and a lot of people of the country are dependent on movies for making their livelihood and thus the analysis of movies is an emerging area.

2.Problem Definition

Have you ever wanted to only watch scenes in a movie with your favourite actor, or watch only the action scenes in the movie?

Each viewer has their own preferences so can they search for scenes which they like in a movie? This isn’t possible with ordinary movie streaming platforms.

The aim of our project is to develop a tool that acts as a semantic search engine for movies, making it easier to navigate movie scenes based on certain semantic elements such as actors, emotions, genre and various other aspects. This gives the viewers more control over the content they view.

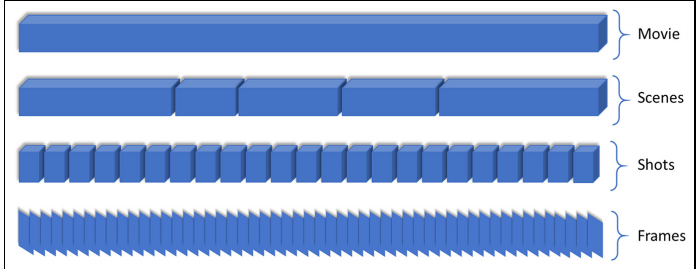
The user would have to enter the movie he wants to watch and he can navigate the movie based on certain pre-defined semantic elements, and find the scenes which fit his needs or liking.

3.Background and Related Work

In our project we had initially discussed the idea of the project, high level view of the project we are going to implement and also the tools or the requirements needed to understand the idea behind the project, which we are trying to accomplish and also to implement the idea itself we were required to understand some basic concepts like OpenCV-python, basics of NLP and how to use the package Stanza which is a NLP library. We did follow a paper on movie scene segmentation, name of the publication is “Movie scene segmentation using object detection and set theory” which has given us the idea how to segment the movie into frames and group them into shots, and how to group the shots based on a given threshold and then how to group the shots into scene(subtitle and script matching used to group into scenes).In the article the following terms were explained:

**Shot**: A Shot is an unbroken sequence of frames recorded from same camera

**Scene**: Scene consists of one or more shots which are semantically co-related and share some physical settings or [resents a continuous action performed by the actors



**Methods of segmentation which were proposed before were:**

There are many methods of segmentation which Researchers have tried, some of them are:

1.The first algorithm for video segmentation calculates short-term shot-to-shot coherence using a colour histogram to measure the similarity between new and current shots.

2. Second algorithm is Graph partitioning-based scene segmentation scheme in which each node represents a shot and edges between nodes represents a shot and edges between nodes represent temporal-visual coherence

3.A text-based scene segmentation has been proposed, they analysed specified scripts of the movies and segmented the movie scenes by aligning the common dialogues and timestamps.

The above proposed methods involve a shot-based key-frame selection which have certain limitations, presenting a shot by one keyframe results in missing lots of information for lengthy shots.  Because the shots of a single scene are semantically related, and it is very challenging to represent semantic meaning through single frames.

To tackle these problems, they proposed a scheme to segment a movie into scenes based on the object’s appearance in each shot.

The article adopted a framework with three steps to implement a effective method of scene segmentation

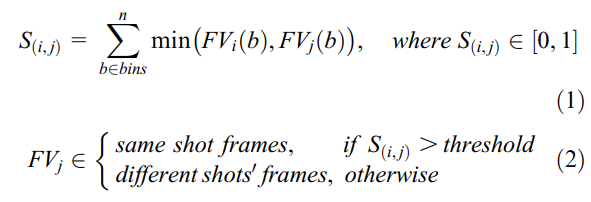
1.The first step segments the input movie into shots.

2.The second step detects objects in the segmented shots.

3.The third step performs object-based shots matching for detecting scene boundaries

**Shot Segmentation:**

Shot segmentation involves grouping multiple frames into shots and is done by using Low level features, such as a colour histogram, Local Binary Patterns (LBPs), and a Histogram of Gradients (HoG) are used to compare frames, frame by frame and group them based on certain similarity score



Above is the grouping algorithm, they calculated the score for two consecutive frames and called it S(i,j) and if it is greater than a threshold value then we say it belongs to the same shot, otherwise we say both frames belong to different shots.

**Object Detection:**

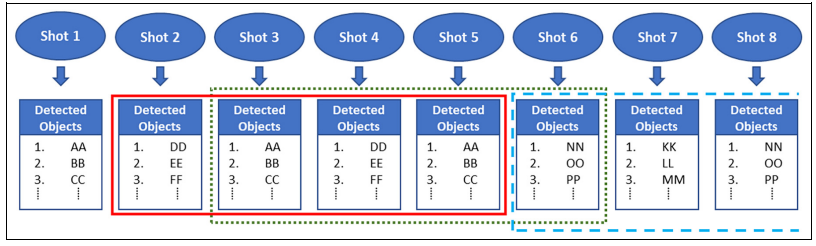
They used a YOLOv3 CNN25 model with DarkNet backend framework, trained on the COCO dataset which contains 80 classes of daily life objects which can be found in movies.

Using this model they detected objects in all frames of the shots to make sure that none of the information is lost.

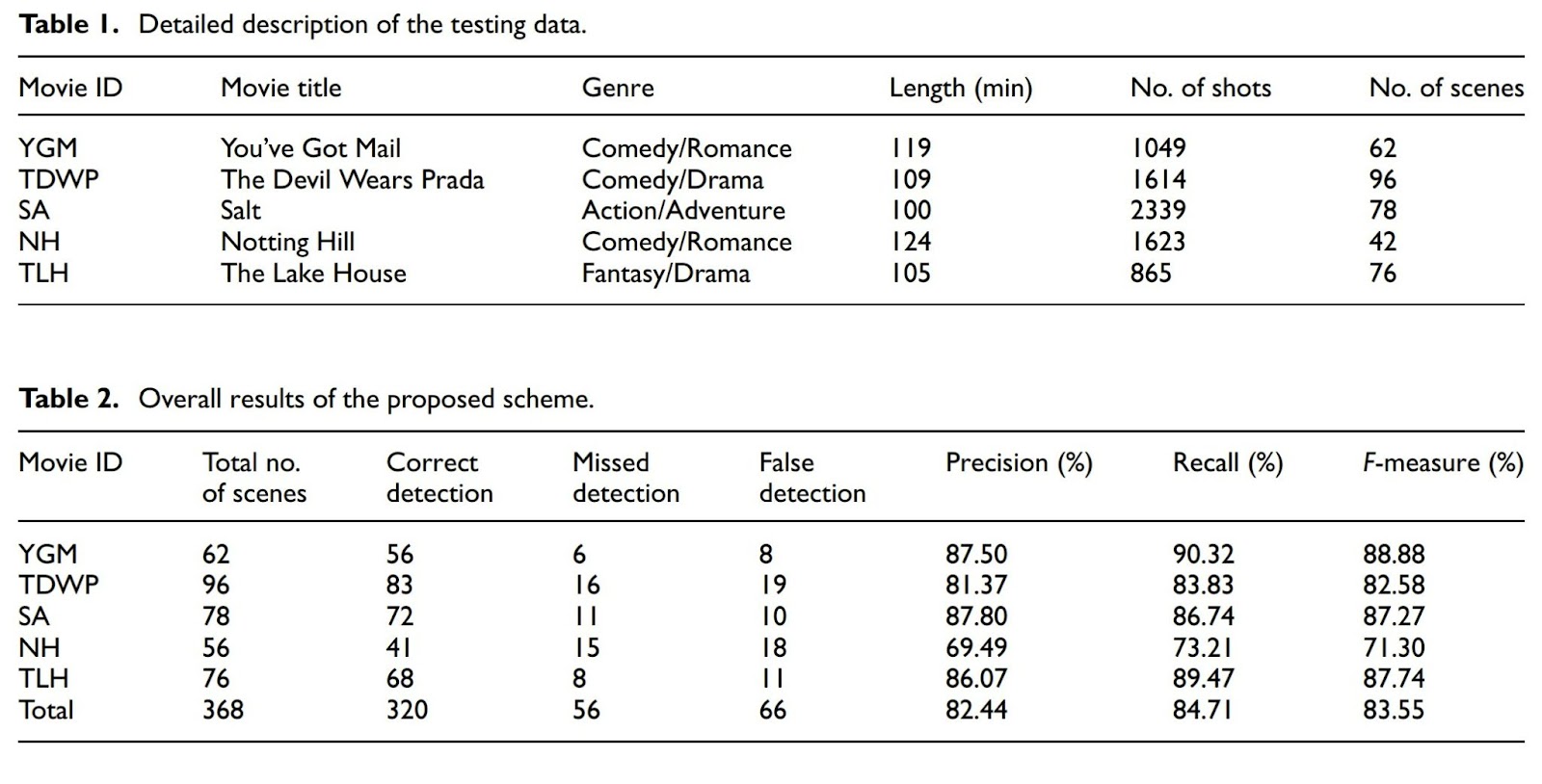
The union of all detected objects of the shot is taken and stored in a set corresponding to that particular shot. Finally, they represent each shot as a set of objects based on the contents present in it with this mechanism.

**Scene Boundary Detection:**

They grouped multiple shots into scenes using set theory and sliding window approach. Let the set of shots 2, 3, and 4 belong to the same scene, the intersection of set 5 is taken with sets 2, 3, and 4, separately. If the number of elements of intersection is greater than or equal to any of the total elements of sets 2, 3, or 4, the shot is considered to be a part of the previous running scene and the window moves to the next shot. Now the intersection of set 6 with the rest of the sets 2,3,4,5 gives an empty set, which means there are zero elements in common with these shots. Hence, a new scene is started, and the window will start from set 6.



**Results Obtained:**



They had considered 5 different movies and obtained the shots and scenes and obtained the above results.Top most table indicates the movie ID (which they assigned), movie title, genre, length in minutes, the total number of shots obtained by following the above mentioned method and finally the scenes obtained by grouping of shots using the Object detection and set theory approach.

In the below table they have mentioned the correct detected and false detected scenes in all the 5 movies, thus calculated Precision and Recall

**4.Implementation**

**4.1 Segmenting into frames:**

We started off by taking the movie and trying segmenting into frames, for that we used OpenCV library and the functions used for this are **cv2VideoCapture**, to obtain the video object and also the **read()** function to obtain the frames and then saving them to a location using imwrite() function. The whole code was written in python and there was only a problem deleting the frames and after using **release()** function we were able to delete the frames.

**4.2 Grouping into Shots:**

After obtaining the frames we then need to group them into shots. We accomplish this by using 2 functions, they are Structural Similarity which gives the similarity values of the two given images using features of the two images and it is done by taking 2 images to be compared and convert them into grayscale images and then pass these grayscale images to the structural similarity function which gives the SSIM value(structural similarity index) which is in range [0,1], 1 implies similarity is maximum.

Next, we have another function which colour histogram of the two images and then calculates the difference between 2 images using the function cv2.compareHist and returns the value in range [0,1] and 0 implying the two given images are very similar.

We do this grouping by taking the consecutive frames and obtaining the score and then if the obtained score is higher than a threshold value then we group the 2nd frame in the shot which already contains the 1st frame.

We initially considered only SSIM values for grouping into shots and thus we fixed a threshold value of SSIM, if above that value then we can add to the same shot else we save the existing shot and create a new shot and repeat the process.

Problem occurred in deciding the threshold value, we had to experiment with different values of the values as setting low value(i.e close to 0) will give shots which overlap each other(which means the shot obtained is the combination of 2 actual shots) and if we take a value which is higher(i.e. close to 1) we get shots which are broken in middle so we experimented different values and first took a low value if shots are overlapped then we take a higher value and then if again obtained shots are broken in middle then we take a mid-value(basically performing bisection using a min and max value).

After obtaining the desired value for which the shot is broken as an actual one, we obtained all the shots and then still we were left with more problems, first we got single frame shots whose duration was less than a second, these shots are created when a frame does not belong to the previous shot or next shot(the score must be lower when compared with the previous frame and also with the next frame), we thought more accurate functions can solve the problem of the single frame shot and thus used another function compareHist to obtain HSV difference scores and thus used a method of both SSIM and HSV values to obtain the shots, the threshold value deciding was quite tough and thus using both methods we had some flexibility on the threshold value, we grouped the shot if SSIM “or” HSV threshold satisfies. We obtained a better result by using this method.

For the problem of 0 length shots we came up with a solution, we took the shots obtained and traversed again and compared the 0 length shots, we compared them to non-0 length shots and then checked to which shot it is similar and then combined it with that shot.

**4.3 Grouping into scenes and Tagging:**

We need to obtain the scenes from the shots grouping them by matching the subtitle file (which has dialogues and timestamps) with the script file using python Regex.

For scene boundary detection we take subtitle file which has:

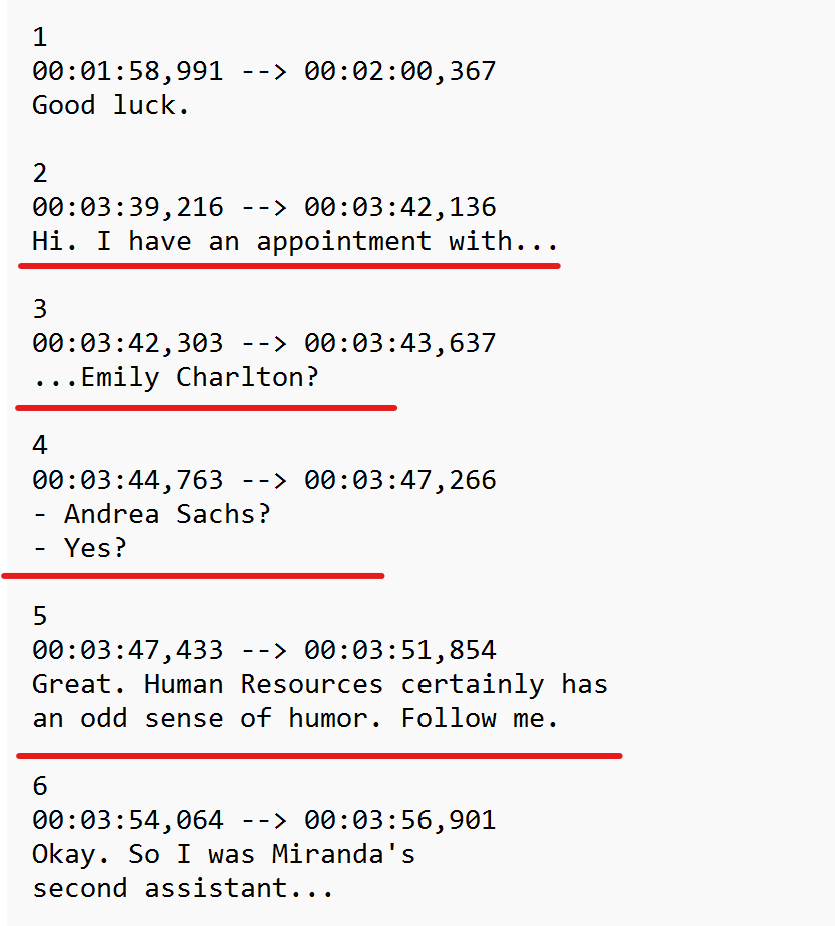
1.Dialogue Number.

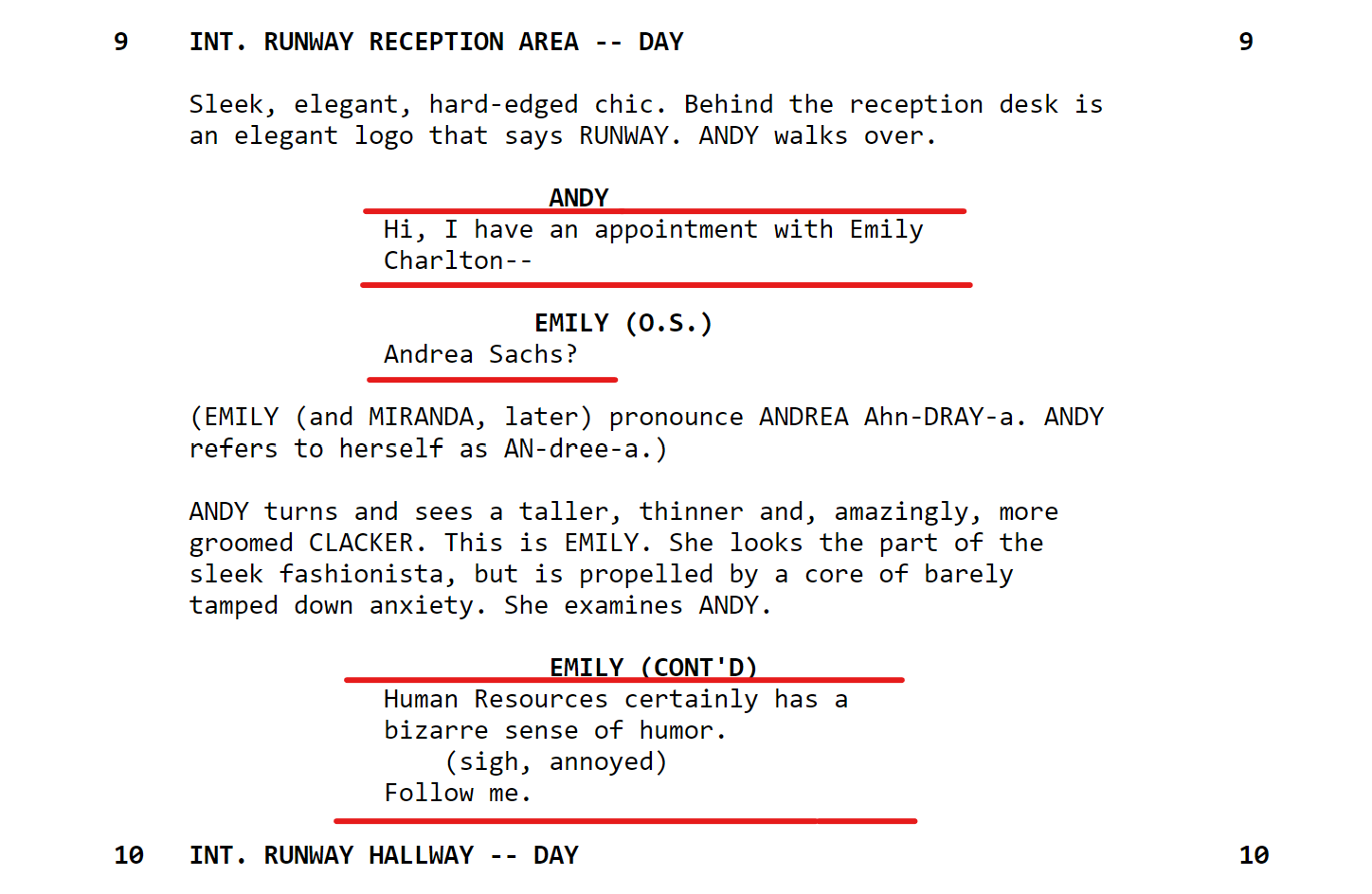
2.Timestamp of the dialogue (Hr: Min: Sec, MSec).

3.Dialogue (String).

Next, we take the Script file which has all the information regarding Scene, Characters and their dialogues and some description about the situation regarding the scene and it has specific indentation for scenes, dialogues and also for characters. Our aim is to extract the scenes using this specific indentation, we match this indentation of scene and when the scene starts we have the Timestamp of the scene as its first dialogue which the script matches with the scene. We take that timestamp as the timestamp of the scene and store the scene number and timestamp and also tags in future using a python dictionary.

Below is the Subtitle and Script file snippets which correspond the same scene.

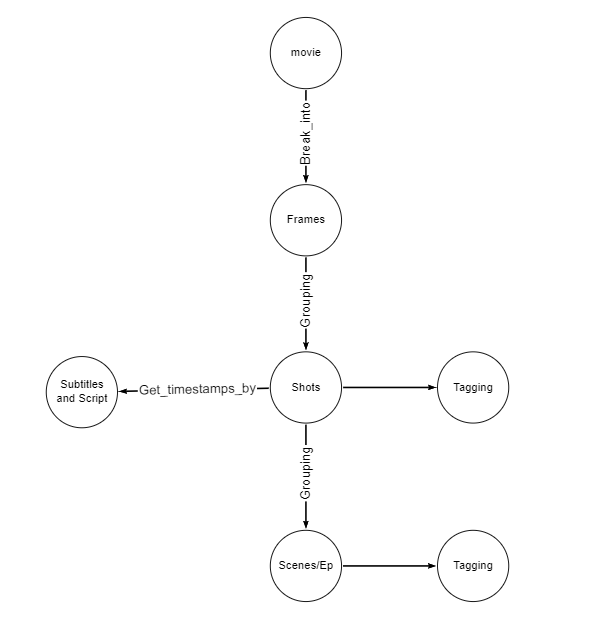




**Figure 4.1 and 4.2 which show snippets of Subtitle and Script file respectively.**

We have used beautiful soup library to segment the HTML script file(extraction HTML format will not lose its indentation by which we can extract the Scene boundaries and Dialogues) and store the results in a list for comparison and also we are able to extract the subtitle file using pysrt module in python, which readily gives us the timestamps and dialogues separately using some pre-defined functions.

Tagging is to be done similarly using NLP(NER specifically) we use NER to get the entities associated with the sentences we parse in the function, we use trained neural network model to retrieve the entities also we tag both the shots and the scene and various tags include: Camera angles, genre of the scene, songs and type of song(intro, duet etc..), characters and the actors, any significant objects in the movie, location of scene(INT/EXT) and finally the Timestamp of the scene.

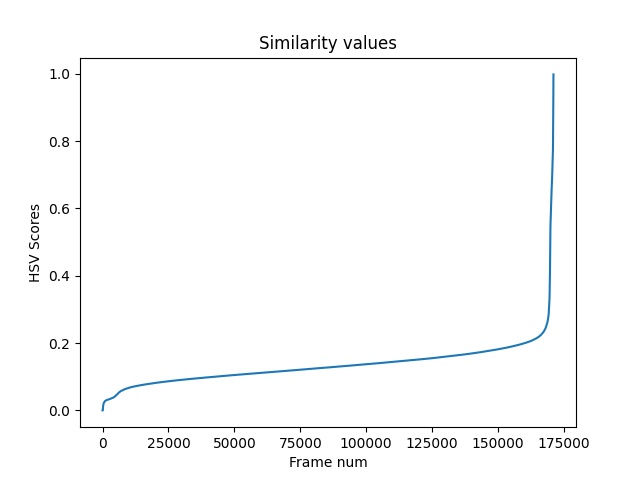


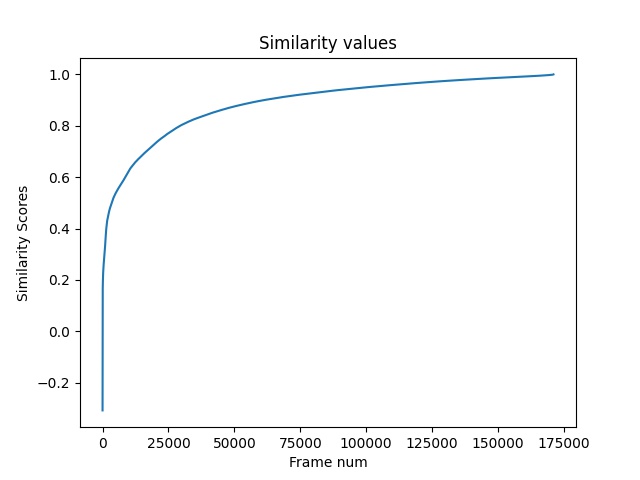
**Above Diagram shows a high-level view of our entire project**

**5.Results**

**5.0 Determining threshold value:**

To Determine a correct threshold value we fixed on a method where we generate all the SSIM values and also the HSV scores for all given frames, for a given movie and sorted them and plotted them as graphs individually(for both SSIM and HSV score) and the point where there is maximum slope change will be the threshold value we have been looking for(because it implies there are many similar frames with the value as the max slope)



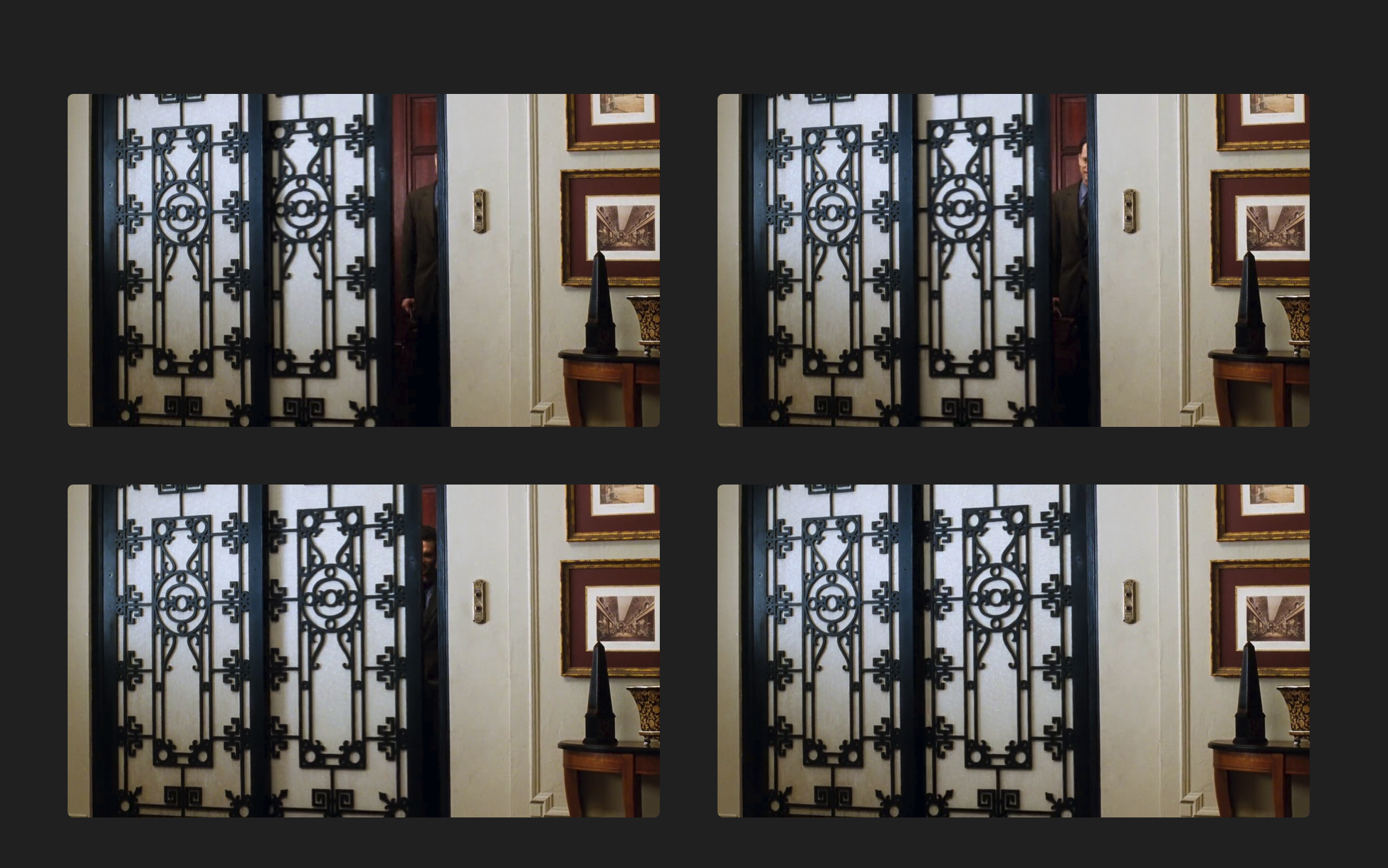


**Fig 5.1 and 5.2 show all values of SSIM and HSV score of movies You’ve Got Mail.**

We also have obtained all the shots for the same movie mentioned above and as we followed the approach of the article, the shots they obtained were **1049** and using our method we obtained **1235** shots for the same movie.

**5.1** Results obtained in Segmenting movie into frames**:**

We will be getting a series of continuous frames by segmenting the movie and there will be some 100K frames obtained from a 2-3GB of film which takes 80-100 GB of space and we did perform this whole operation using the NVIDIA server of our college as the whole process is CPU intensive.

  
**Figure 5.3** shows the some of continuous frames obtained by movie segmentation into frames of the movie You’ve Got Mail.

5.2 Grouping Frames into Shots:

With the obtained frames we then group them into shots (info in implementation section 4.2).

Video 5.1: <https://drive.google.com/file/d/1V46y4MUPSlowStKIPODV1R5k8P2vUV35/view?usp=sharing>

Video 5.2: <https://drive.google.com/file/d/11oUF0oGyNsBbkLD8zVAJDqaCgDvcOQ61/view?usp=sharing>

Video 5.1 and 5.2 are examples of shots obtained by grouping frames.

5.3 Scene Boundary detection:

In this section we obtain the timestamp of the scene and store it in a python dictionary and the results obtained are meant to visualize in a graph database where nodes are scenes and their attributes are the timestamps and tags which are to be added in the future.

Graphical user interface, text, application, email

Description automatically generated

Fig 5.4: Output of scene boundary detection code, where it shows it has found a scene boundary (Line is the line in script file and Diag is the line in subtitle file).

**6.Conclusion**

We have stated different methods for segmenting movie into frames using OpenCV and also for grouping into shots using 2 main functions Structural similarity which uses image features and the compareHist which uses the colour histogram of the images and also we have proposed a method for obtaining better results in grouping of shots and also method to handle the 0 length shots.

In the end we conclude that the results obtained match to some extent to that of those mentioned in the article, as the exact method of implementation was not mentioned and also the methods proposed are not the only approach and there can be better approaches to solve the same problem.

**7.References**

1. [OpenCV Python Course - Learn Computer Vision and AI](https://www.youtube.com/watch?v=P4Z8_qe2Cu0)

2. https://stanfordnlp.github.io/stanza/

3. https://journals.sagepub.com/doi/pdf/10.1177/1550147719845277

4. <https://app.studiobinder.com/company/580e85847e7982164664e844/collab/5fce8be550eb2f12c88a57c8/projects/5fbbc24766461211963c3b52/document/5fbbc257231435110248f01f?filter=%257B%2522l%2522%253A%25225fbbc268680e3a404541b38c%2522%257D&utm_source=blog&utm_medium=sb-app-link&utm_campaign=script&utm_content=1408-full-script-pdf-download&utm_term=best-free-movie-scripts>

5. Stack Overflow for debugging errors.