



Smart Trailer: Automatic generation of movie trailer using only subtitles

by

Bishoy Hani
Mohammad Hesham
Nour Fouad

A dissertation submitted in partial fulfillment of the
requirements for the degree of
Bachelor of computer science

in

Department of Computer Science

in the

Faculty of Computer Science

of the

Misr International University, EGYPT

Thesis advisor:
Dr. Eslam Amer
Eng. Menna Gamil

(June 2018)

Abstract

With the enormous growth rate in user-generated videos, it is becoming increasingly important to be able to navigate them efficiently. Video summarization is considered a promising approach for efficacious realization of video content through Identifying and picking out descriptive frames of the video. In this paper, we propose an adaptive framework called Smart-Trailer (S-Trailer) to automate the process of creating an online trailer for any movie based only on its subtitle. The language used in the subtitle is English. The framework analyzes the movie subtitle file to extract relevant textual features that are used to classify the movie into its corresponding genre(s). Initial experimentation resulted in generating genre-classification corpus. The generated corpus is tested against real movies dataset and showed high classification accuracy rate (0.89) in classifying movies into their corresponding genre(s). The proposed system returned automated trailers that contain on average 47% accuracy in terms of recalling scenes appeared on the original movie trailer for different movie genres. Currently, we employ deep learning techniques to captures user behaviours and opinions in order to adapt our system to provide users with relevant video scenes recommendations that match their preferences.

Acknowledgments

Firstly, We would like to thank our supervisor Dr. Eslam Amer of the faculty of Computer Science at Misr International University. His office was always opened for us and for our researches. He was welcoming any question regarding our graduation. He was helpful throughout the whole year.

We would also like to thank Dr. Ayman Ezzat of the faculty of Computer Science at Misr International University for his great work towards the graduation projects at Misr International University, for his participation with feedback and opinions to remain or to change whatever in the research and for his frequent guidance to reach the optimum solutions.

Our sincere thanks go to Dr. Ashraf Abdelraouf And Dr. Ayman Bahaa of the faculty of Computer Science at Misr International University for his great support, motivation and their knowledge.

Our sincere thanks also go to Dr. Ayman Bahaa Dean of the faculty of Computer Science at Misr International University for his great support and his work for the development of the faculty of Computer Science at Misr International University.

We also thank Prof. Dr. Jiro Tanaka of the school of Information, Production and Systems in WASEDA University for giving his feedback and giving better solutions for better change.

Last but not the least, We would like to thank Dr. Tamer Ikbale of the faculty of Mass Communication at Misr International University for frequently giving his feedback towards the output of our work.

Contents

Abstract	ii
Acknowledgments	iii
List of Tables	5
List of Figures	6
1 Introduction	7
1.1 Introduction	7
1.1.1 Background	7
1.1.2 Motivation	7
1.1.3 Problem Definition	8
1.2 Project Description	8
1.2.1 Objective	8
1.2.2 Scope	8
1.2.3 Project Overview	9
1.3 Project Management and Deliverable	9
1.3.1 Budget and Resource Cost	9
1.3.2 Supportive documents	9
2 Literature Work	11
2.1 Similar system information	11
2.1.1 Similar System Description	14
2.1.2 Comparison with Proposed Project	14
3 System Requirements Specifications	15
3.1 Introduction	15
3.1.1 Purpose of this document	15
3.1.2 Scope of this document	15
3.1.3 Overview	16
3.1.4 Business Context	17
3.2 General Description	17
3.2.1 Product Functions	17

3.2.2	User Characteristics	18
3.2.3	User Problem Statement	18
3.2.4	User Objectives	19
3.2.5	General Constraints	19
3.3	Functional Requirements	19
3.3.1	FR1	19
3.3.2	FR2	20
3.3.3	FR3	20
3.3.4	FR4	20
3.3.5	FR5	20
3.3.6	FR6	21
3.3.7	FR7	21
3.3.8	FR8	21
3.3.9	FR9	22
3.3.10	FR10	22
3.3.11	FR11	22
3.3.12	FR12	22
3.3.13	FR13	23
3.3.14	FR14	23
3.3.15	FR15	24
3.3.16	FR16	24
3.3.17	FR17	24
3.3.18	FR18	24
3.3.19	FR19	25
3.3.20	FR20	25
3.3.21	FR21	25
3.3.22	FR22	26
3.3.23	FR23	26
3.3.24	FR24	26
3.3.25	FR25	27
3.3.26	FR26	27
3.3.27	FR27	27
3.4	Interface Requirements	28
3.4.1	GUI	28
3.4.2	API	28
3.5	Performance Requirements	28
3.6	Other non-functional attributes	29
3.6.1	Security	29
3.6.2	Reliability	29
3.6.3	Portability	30
3.6.4	Maintainability	30
3.7	Preliminary Object-Oriented Domain Analysis	30
3.7.1	User	30
3.7.2	Subtitle	30

3.7.3	Lemmatizer	32
3.7.4	Tokenizer	32
3.7.5	TextFilter	32
3.7.6	FrequencyCounter	33
3.7.7	TFIDF	33
3.7.8	KNNClassifier	34
3.7.9	SWRemoval	34
3.8	Operational Scenarios	35
3.8.1	System Scenario	36
3.8.2	User Scenario	37
3.9	Appendices	37
3.9.1	Definitions, Acronyms, Abbreviations	37
4	System Software Design Document	38
4.1	Introduction	38
4.1.1	Purpose	38
4.1.2	Scope	38
4.1.3	Overview	39
4.1.4	Definitions and Acronyms	39
4.2	System Overview	39
4.3	System Design	41
4.3.1	Architectural Design	41
4.3.2	Decomposition Description	42
4.3.3	Design Rationale	44
4.4	Data Design	45
4.4.1	Data Description	45
4.5	Component Design	45
4.5.1	TF-IDF Term frequency/Inverse document frequency	45
4.5.2	TF-IDF Results	46
4.5.3	KNN: K-Nearest Neighbor	47
4.5.4	KNN Distance Equations	47
4.5.5	KNN Classification Results	48
4.5.6	Text Rank Algorithm	48
4.6	Human Interface Design	48
4.6.1	Overview of User Interface	48
4.6.2	Screen Images	49
4.6.3	Screen Objects and Actions	50
4.7	Requirements Matrix	51
5	Evaluation	54
5.1	Introduction	54
5.2	Experiment 1 Movie Genre Classification	54
5.2.1	Goal	54
5.2.2	Movie Genres Tested	54
5.2.3	Task	55

5.2.4	Results	55
5.3	Experiment 2 Precision and Recall with the manual trailer	55
5.3.1	Goals	55
5.3.2	Movies Tested	55
5.3.3	Results	56
5.4	Expirement 3 - User Evaluation	56
5.4.1	Goal	56
5.4.2	Task	56
5.4.3	Movie Tested	56
5.4.4	Results	56
6	Conclusion and Future Work	57
6.1	Conclusion	57
6.2	Future Work	57
	Bibliography	58

List of Tables

3.1	Total Project compilation time	29
3.2	Preprocessing Module compilation time	29
3.3	TFIDF KNN Module compilation time	29
5.1	Classifiers Accuracy Table	55
5.2	Precision and Recall Table	56

List of Figures

1.1	System Overview.	9
1.2	Gantt Chart for Time Plan.	10
2.1	Similar System Comparison	14
3.1	Block Diagram.	16
3.2	Website Wireframe.	28
3.3	Primitive Class Diagram.	31
3.4	System Scenario.	35
3.5	User Scenario.	36
4.1	System Overview.	40
4.2	Architectural Model	41
4.3	Class Diagram	42
4.4	Decorator design pattern	43
4.5	MVC design pattern	43
4.6	Sequence Diagram for The Classifier	44
4.7	Sequence Diagram for The Ranking	44
4.8	Database	45
4.9	TF-IDF equation for word (i) in document (j)	46
4.10	Weight matrix resulted from TF-IDF algorithm	46
4.11	Euclidean equation where I is the number of K neighbors	47
4.12	Manhattan equation formula	47
4.13	Sign Up Page	49
4.14	Login Page	49
4.16	Resulted Trailer Page	49
4.15	Upload Page	50
4.17	Recommended Shots Page	50
4.18	Requirement Matrix	51
4.19	Requirement Matrix	52
4.20	Requirement Matrix	53

Chapter 1

Introduction

1.1 Introduction

1.1.1 Background

Movie producers around the world spend a lot of money (millions of dollars) on movie trailers. Before watching any movie, people watch the trailer then decide whether they are going to watch it or not, so we can consider the trailer as the base of success of any movie and if the trailer is a failure, some great movies fail because of them. To save a huge amount of money, huge effort of watching a movie more than once and repeating a trailer to get a perfect one, we are going to develop a system that according to the most important subtitles is going to automatically extract a perfect trailer to get the movie success needed. [1].

1.1.2 Motivation

This project is mainly to decrease the huge amount of money spent on trailers per movie. Some systems have implemented this project before but they faced some challenges as taking specific shots to assist the trailer but couldn't extract the trailer from scratch, analysing scenes of low-level visual which isn't 100% accurate and focusing on classifying the movies without extracting a trailer as a result. According to the results of the survey, 59.52% of people are affected with movie trailers, 50.00% have watched movies with bad trailers, 76.19% are females and 85.71% their ages range are from 18-24.

1.1.3 Problem Definition

Movie producers sometimes complain about paying a huge amount of money on trailers which don't achieve the expected success for that movie and this trailers became the reason behind the failing of that movie after spending all this money so that people watch the movies. So, we aim to reduce this huge amount of money paid by by a system that generates a perfect trailer automatically according to detection and classifications done on each movie and analyzing the subtitles by the removal of stop words and the in inflectional endings from words and extracting the most important, catchy and attractive keywords to make a bag of words for each movie category and this bag of words is going to be compared with each subtitle file to classify the movie genre and according to this genre extract shots from the movie and finally generating an interesting movie trailer with least spending.

1.2 Project Description

The system should be given the subtitle file of a specified movie, through the subtitle file the movie genre will be classified and detected. The next phase will be about ranking the most important scenes from the subtitle file with respect to the movie genre, then merging these scenes to get the Trailer.

1.2.1 Objective

The systems purpose is to decrease the financial spending on movie productions and specifically the costs spent on producing an attracting movie trailer also its going to save a lot of effort as cutting a trailer usually takes from 2 to 3 days.

1.2.2 Scope

The system will work on Sub-Rip format (.srt) which is the most used format for subtitles, supported by most software video players, many subtitle editing/creation tools and a lot of hardware media player devices. The subtitle should be written in English(US) language. Natural Language processing techniques will take place on the subtitle file as the file must not contain stop words, less significant words and only contains the catchy keywords or phrases.

1.2.3 Project Overview

The system will be mainly written in Python and some classification algorithms will be used to classify the movie genre and also to rank the most important scenes through the subtitle file, all of these processes will lead to bag of word that will be the reference to the next movie trailers needed to be generated using the system. The classification process is performed using the naive Bayes method as each and every word has a specific weight in respect to the category of the movie.

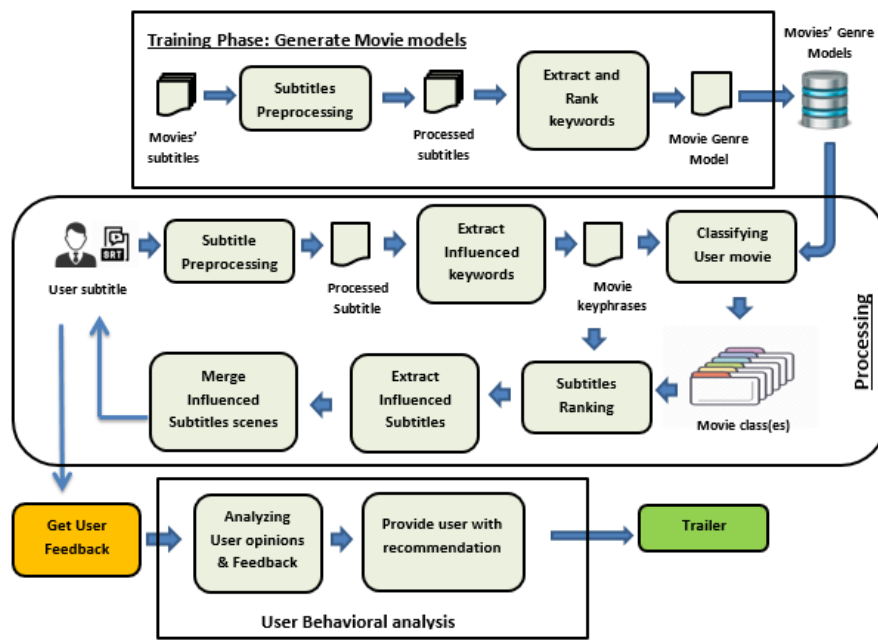


Figure 1.1: System Overview.

1.3 Project Management and Deliverable

1.3.1 Budget and Resource Cost

0.0 EGP

1.3.2 Supportive documents

From the results of a survey, 59.52 percent of people answered 5 which is highly agree to question "How likely does a movie trailer affect your opinion about watching a movie or

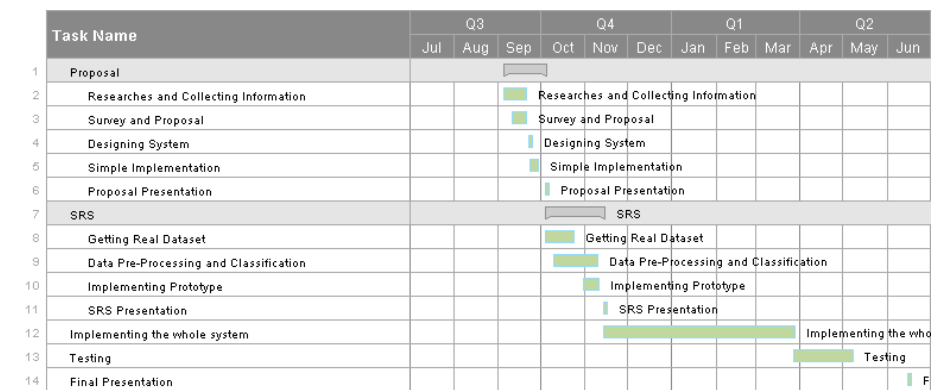


Figure 1.2: Gantt Chart for Time Plan.

not?” and 50 percent answered yes to the question ”Did you watch a movie before with a bad trailer?”.

Chapter 2

Literature Work

2.1 Similar system information

Sinnu Susan Thomas, et.al suggested a new framework for video summarization [2] which they decided to be different from any other video summarization tool which depends on the statistical redundancy to return the summarized video but this technique uses human visualization features to extract only the most significant events from the video without redundancy.

Hyuncheol Kim, et.al proposed a method in [3] to summarize videos using the feature dissimilarities by cutting the video into segments according to the high peak change from one scene to another in the original video and consider them the scenes that have to be put in the summarized video.

Mayu Otani, et.al introduced a video summarization method in [4] based on text. The system takes a video and the text from the blogger to match the text with the frames that has this text and give them the priority to be put in the video according to users' preferences. This method is close to our proposed framework as we also use text to summarize video.

Yitian Yuan, et.al introduced a model of video summarization in [5] which is building a model that takes frames from the video and some extra embedded information like the title, queryetc. The produced video is summarized according to some segments that are selected

from the original video through minimizing the distance to the side information. However, the drawback of this approach is that this side information is never enough as the distance between a frame and title is small while this scene may not add any attractiveness to the summarized video.

Hongteng Xu, et.al proposed a system in [20] that measure the attractiveness of videos through giving the system a lot of trailers to learn from them the most attractive scenes so that this system can generate a trailer automatically according the level of attractiveness. The drawback of this approach is that not all audience get attracted by the same scenes, each one has his own taste in movies, so this attractiveness level measured might not always attract people.

Mayu Otani, et.al suggested a video summarizing technique in [6] that takes some deep semantic features which is extracting from each shot in the original video some objects, actionsetc. Those features from each scene are grouped together to finally generate a summarized video that has the main actions or events in the original video. However, the negative side of this approach is that most important scenes might not only be actions or objects it might be words or sentences that are said in the original video.

Konstantinos Bougiatiotis, et.al [7] demonstrated the extraction of topical representation of movies based totally on subtitles mining. of their work, they tested the life of a similarity correlation between film content material and low stage textual capabilities from respective subtitles. in their technique, they created a subject version browser for films, allowing customers to discover the extraordinary factors of similarities among films. however, the downside of this system is that it offers the consumer the similarity among movies and which is the closest to the other, but it does now not advocate what to look at depends at the input of the user and it does no longer point out the style of the movie.

Amy Pavel, et.al [8] worked on making a digest for videos that afford to browse and skimming by segmenting videos given to sections and giving short summaries of text for every section extracted. The users will navigate by reading the section outline and opt for the section to get or access the present purpose or subject of the video. The resulted system provides a good authoring interface and exploring techniques for mechanically segmenting

and summarizing AN input video from the user. the downside of this technique returns video partitions consistent with titles, chapters or sections however if the title is incomprehensible for any topic in the video, the system wont be ready to summarize it accurately.

J. Nessel, et.al [1] proposed a system that recommends movies to the users by examination users predefined examples with the matter contents of flicks. The system is developed within the context of recursively decidable languages and calculable functions. However, the downside of this system is that it depends solely on extracting words from the user while not victimization any further preferences or opinions from the user except AN example for a moving-picture show name or a word presented in an exceedingly dialog.

Go Irie, et.al [9] presented a content-based trailer generation method named Vid2Trailer (V2T). V2T automatically creates staggering trailers out of original movies via figuring out amazing audiovisual components, as nicely namely accomplishment symbols concerning the film name certain as like the name logo and the affairs music. Results confirmed more appropriateness of V2T method compared after traditional tools. However, the processing effort over the V2T regulation regarded even much as it tends according to filter the address after calculating important words over the entire movie. Nevertheless, this technology can be decreased condition the calculations of top impacted keyword take place after filtering subtitles beyond unneeded words rather than calculating the entire subtitle file.

Howard Zhou, et.al [10] proposed a framework that is based on scene classification. The framework showed that a transiently organized component in light of such middle of the road level portrayal of scenes can enhance the characterization execution over the utilization of low-level visual includes alone. In spite of the fact that the arrangement execution somewhat improved, the disadvantage of this framework utilizes a sack of visual words which thus needs an immense stockpiling to spare a sack of visual expressions of every motion picture class.

Alan F. Smeaton, et.al [11] presented a feeler which automatically selects shoot s from action mechanism movies in parliamentary procedure to assist in the creation of drone s. The approach depends on shot boundary proficiency in society to generate the basic shot-based structure of a movie. The sound rails of a movie are analyzed in social club to detect

the presence of the following categories: speech, music, silence, speech with ground music and other audio. Nowadays movies contain a miscellany of writing style, therefore, any system that generates house trailer should also reflect such mixture. The Major drawback of Alan approach is that it is particular to action shots exist in movies. If the movie contains many genres, the system wont be able to generate a satisfactory trailer for it.

2.1.1 Similar System Description

All over the world, trailer production is a process that is done manually through watching a whole movie, cutting the shots, merging them and adding the beat to get a trailer. All the above tried some systems to automate this process, In [2], this framework takes the important events which is different from one person to another while for example some people may see that the most important event is where the actor they love appear in in a movie trailer, This method [3] is that not all the changes in the scenes indicates an important event it may be for example a change for the places only. While [4] is the closest one to our approach as it also works with text analysis not video and audio.

2.1.2 Comparison with Proposed Project

	Method	Training Sample	Accuracy
Movie Genre Classification via Scene Categorization Our Proposed Model	Through Scenes	1239 movie trailer	71.58%
	Through Subtitles	40 subtitle files	75%

Figure 2.1: Similar System Comparison

Chapter 3

System Requirements Specifications

3.1 Introduction

3.1.1 Purpose of this document

This document is mainly for the full description of the requirements for the project S-Trailer. This document will explain how the cycle of the system will go on, with the assist of the overview, constraints, functional and non-functional requirements which help this document to illustrate what should the user know and how the user will use the system.

3.1.2 Scope of this document

The S-Trailer project is a web application, which helps the production companies to save money, time and effort for making trailers for their movies. This system will help people to make summary for their movies and series as well. The user will provide the system with a subtitle file and a soundtrack audio file as an input; the system will start processing on this subtitle to filter it from the useless data inside, the system will determine which is the most important scenes in this movies depends on that subtitle, then the system will cut the movie into clips depends on the time-stamp that extracted from the subtitles. All these scenes will be merged together and by adding a soundtrack the trailer will be ready, the the system will give the user the chance to add delete and order his shots for the trailer, he

can also add a shot of a sentence that he likes. This software needs internet access and a movie blender for merging clips, System can be used as a video summary as well.

3.1.3 Overview

System is divided into three major phases; The First phase is the training phase which is mainly used for creating the movie genres. In this phase system gets the subtitles of some movies with the same genre and start pre-processing on those subtitles files then well get some bag-of-words of the current genre which will act as a reference to the uploaded subtitles to be classified. The Second phase is the pre-processing of the users subtitle, starts with letting the user uploads his subtitle, system will start pre-processing on this subtitle to get some keywords, then weighting this keywords to know which is more in influencing then movie classification takes place at this moment to determine the genre of the movie, depends on this top ranked keywords system will get the time intervals that contains these keywords, then by splitting these scenes from the movie then merge them together will make the trailer for the user. The Third phase is the recommendation, after extracting the trailer, user can give a feedback or some keywords that he wants to see in his trailer so system will take these keywords in consideration to extract some recommended shots to the user, so he can add it in his trailer.

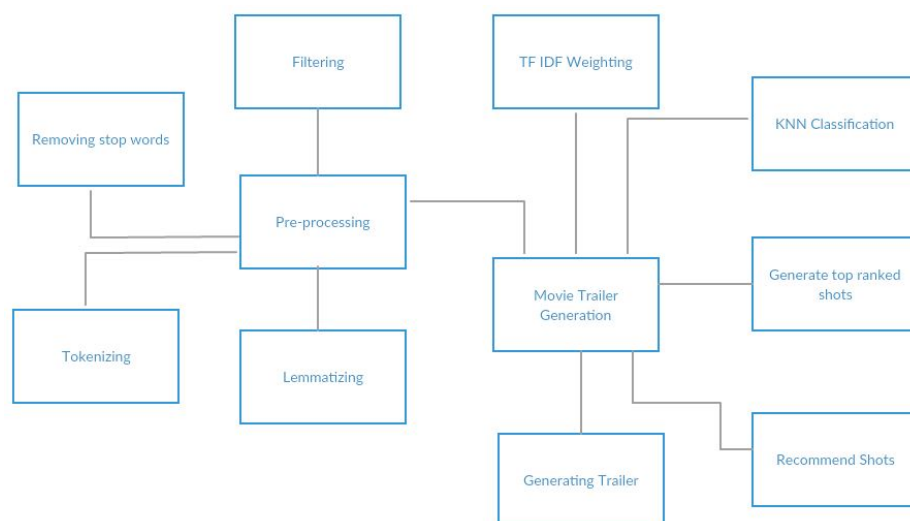


Figure 3.1: Block Diagram.

3.1.4 Business Context

Movie trailers became major events before the release of the movie .A lot of people think that the trailer is made by the director of the movie only but they don't realize that there are agencies with editors that create and work for this trailer. Basically a huge amount of money is paid for these agencies to get the trailer; they don't know the effort and time spent just to extract that trailer [12]. Imagine now that you as a director or a producer have a fixed application that gives you the ability to extract your trailer with a very small amount of effort,time and money. All you have to do is to get the S-Trailer application and give this application the subtitle file you will create for that movie as an input with the soundtrack audio file, after processing, the application will inform the user the category (genre) of the movie then it will extract the trailer as an output.Then user will have the ability to edit his movie trailer from the extracted shots by adding shots, deleting shots or arrange in a specific way.Now compare the effort between: automatically generating subtitle or start dealing with a kind of agencies to create a trailer with paying money, wait for the trailer to be finished and the agency will make some effort to make the trailer.

3.2 General Description

3.2.1 Product Functions

- The user can upload what is needed for extracting the trailer which is upload .srt file.
- The User can upload an audio soundtrack as well to be added to the subtitle.
- System will remove any symbol from the subtitles.
- System will remove any digit from the subtitles.
- System will convert the upper-case letters to lower-case.
- System splits text into separate words.
- System determine the origin of each word (is-be, are-be) to prevent repetition.
- System removes the stop-words from the subtitles file for filtration.
- System calculates the weight matrix of the subtitles file.
- System classifies the movie genre from the subtitles file inserted.
- The User can watch his extracted trailer.
- Then if the user likes the trailer he can download it.

- User can manipulate his trailer as he can add shot which makes the user add some shots from the top ranked shots to his trailer.
- User can delete shots from the extracted trailer if he doesn't like any.
- User has the ability to arrange his shots depends on his desire.
- User can update the soundtrack or delete it if he wants to manually manipulate his sound effects or soundtrack.
- User can search for a word that exists in the subtitles of the movies to add the shot that contains this word or sentence.
- User will be provided with some recommendations invoked by the system after searching the related shots to his search to have the ability to add it to his trailer.
- Admin can manipulate everything related to the bag of words of each genre, it means that admin can handle the genres.
- Admin can create a new bag of words, add file or text to be the new genre.
- Admin can also delete and edit existing bag of words.

3.2.2 User Characteristics

There is no Specific background for the user to deal with the system, as it shouldn't be only the producer who can use the system, it's available for all who are interested to create a trailer. But its important for the user to know how to work on the internet or the web browser. In this system there is the normal user who uploads a srt file with audio file and waiting for results, then can manipulate if he wants. The normal user who uploads the srt is mainly a producer or a director whose waiting for some results and values to be added to their movie trailer.

3.2.3 User Problem Statement

Movie producers sometimes complain about paying a huge amount of money on trailers which don't achieve the expected success for that movie and this trailers became the reason behind the failing of that movie after spending all that amount money. So, we aim to reduce this huge amount of money paid by a system that generates a perfect trailer automatically according to detection and classifications done on each movie and analyzing the subtitles by the removal of stop words and the inflectional endings from words and extracting the most

important, catchy and attractive keywords to make a bag of words for each movie category and this bag of words is going to be compared with each subtitle file that's uploaded to classify the movie genre, according to this genre extract shots from the movie and finally generating an interesting movie trailer with least amount of money. giving the user the opportunity to change and manipulate the shots is one of the most important features.

3.2.4 User Objectives

Users main objective is to save money, time and effort by using this system, but there are some other objectives that user will need such as fast pre-processing as much as the system can, then getting the time intervals and start applying it on the movie to cut and merge should be fast and accurate. Then edit the trailer if he wants some extras. A good interface makes the user satisfied with working on the system, so the good GUI will be important as well.

3.2.5 General Constraints

This system's primary structure constraint is the Network connection consolidation. Since it is a web application, so it needs internet connection for uploading the given srt file and soundtrack audio file, then system will make the processing on that file and get the time intervals of the catchy scenes to be merged.

3.3 Functional Requirements

3.3.1 FR1

- Name : Login
- Description : It lets the user login with his user-name and password
- to start the system functions and the srt will be the file uploading.
- Input : username, password
- Output : Redirection to upload file
- Criticality : High
- Expected risks : Wrong username or password, Database access failure
- Dependencies : -

3.3.2 FR2

- Name : upload subtitle
- Description : It lets the user upload his own .SRT file to start processing on this file.
- Input : .SRT file
- Output : Redirection to the preprocessing of the file
- Criticality : High
- Expected risks : Invalid file type or wrong extension
- Dependencies : 3.3.1 login

3.3.3 FR3

- Name : get wordnet pos
- Description : It determines the wordnet part of speech of the normal part of speech.
- Input : POS (Part-of-Speech)
- Output : WordNet (Part-of-Speech)
- Criticality : High
- Expected risks : If the given POS is invalid
- Dependencies : 3.3.4 text lemmatize

3.3.4 FR4

- Name : text lemmatize
- Description : It determines the origin of each word (lemmatization).
- Input : Array of Words
- Output : Array of Words with its Origin
- Criticality : High
- Expected risks : If it gets confused between words
- Dependencies : 3.3.6 tokenize

3.3.5 FR5

- Name : tokenize text
- Description : It splits any kind of text into separate tokens.

- Input : Text
- Output : Array of words
- Criticality : High
- Expected risks : If it gets text without spaces or breaks
- Dependencies : 3.3.2 upload subtitle

3.3.6 FR6

- Name : remove stopwords
- Description : It removes any stopword from the text as a kind of filtration.
- Input : Text
- Output : Text without stopwords
- Criticality : High
- Expected risks : -
- Dependencies : 3.3.5 tokenize text

3.3.7 FR7

- Name : count word frequency
- Description : It counts the frequency of each word from the given text.
- Input : Text
- Output : print Each word with its frequency
- Criticality : High
- Expected risks : wrong file name or directory
- Dependencies : 3.3.5 tokenize text

3.3.8 FR8

- Name : getIDFValue
- Description : Function to calculate the Inverse Document Value of a single word.
- Input : Word
- Output : IDF value as a variable
- Criticality : High
- Expected risks : invalid input
- Dependencies : -

3.3.9 FR9

- Name : getTFValue
- Description : Function to calculate the Term Frequency of a single word.
- Input : Word
- Output : TF value as a list
- Criticality : High
- Expected risks : invalid input
- Dependencies : -

3.3.10 FR10

- Name : getTfIdf
- Description : Function to calculate the weight matrix(weight of each word = $TFvalue(w) * IDFvalue(w)$).
- Input : Text
- Output : TFIDF weight matrix
- Criticality : High
- Expected risks : invalid input
- Dependencies : 3.3.9 getTFValue 3.3.8 getIDFValue

3.3.11 FR11

- Name : weightClassifiedSubtitle
- Description : Determine the TF matrix of the keywords inside the preprocessed textfile.
- Input : Text
- Output : Weight matrix
- Criticality : High
- Expected risks : Invalid input
- Dependencies : 3.3.13 calculateDistance

3.3.12 FR12

- Name : calculateDistance

- Description : It passes the extracted keywords to the TF-IDF module to get the TF-IDF weightmatrix, Applying the KNN euclidean distance between the TF-IDF weightmatrix and the TF matrix of the keywords determined in the previous function and Getting the shortest distances and calculating the majority of the shortest distances according to the specified movie categories.
- Input : Matrices
- Output : Shortest Distance
- Criticality : High
- Expected risks : The Matrices entered are invalid or corrupted
- Dependencies : 3.3.10 getTfIdf

3.3.13 FR13

- Name : get top shots time intervals
- Description : It gets the time intervals of the most important scenes in the movie from the top ranked words extracted.
- Input : words
- Output : time intervals
- Criticality : High
- Expected risks : cannot find specific word from subtitles file
- Dependencies 3.3.15 get top ranked words

3.3.14 FR14

- Name : get top ranked words
- Description : It gets the top words from the high frequency weighting of the keywords.
- Input : text
- Output : top words
- Criticality : High
- Expected risks : empty text file inserted
- Dependencies : 3.3.11 getTfIdf

3.3.15 FR15

- Name : split shot
- Description : It splits the extracted top shots time intervals from top ranked words in the subtitles file from the movie video file.
- Input : time intervals
- Output : separated splitted shots
- Criticality : High
- Expected risks : invalid time intervals
- Dependencies : 3.3.13 get top shots time intervals

3.3.16 FR16

- Name : merge shot
- Description : It merges the extracted splitted shots time intervals from top ranked words in the subtitles file to generate the system's automatic trailer.
- Input : time intervals of splitted shots
- Output : merged shots (video)
- Criticality : High
- Expected risks : invalid time intervals
- Dependencies : 3.3.15 split shot

3.3.17 FR17

- Name : watch trailer
- Description : It allows the user to watch his generated trailer from the given subtitles with the given soundtrack.
- Input : video (trailer)
- Output : play video
- Criticality : High
- Expected risks : -
- Dependencies : 3.3.17 merge shot

3.3.18 FR18

- Name : download trailer

- Description : It allows the user to download his generated trailer from the given subtitles with the given soundtrack.
- Input : video (trailer)
- Output : download video
- Criticality : High
- Expected risks : network connection drawback
- Dependencies : 3.3.16 merge shot

3.3.19 FR19

- Name : get recommended shots
- Description : It extracts the time intervals that matches with user choices and things that related to the same shots to be recommended to the user.
- Input : searched word
- Output : recommended shots (time intervals)
- Criticality : High
- Expected risks : -
- Dependencies : *

3.3.20 FR20

- Name : show recommended shots
- Description : It allows the user to see the recommended shots depends on the search keys he entered in the search key.
- Input : -
- Output : recommended shots (time intervals)
- Criticality : High
- Expected risks : -
- Dependencies : -

3.3.21 FR21

- Name : remove symbols
- Description : It removes the symbols from the subtitle given (e.g.
- Input : Text

- Output : Text without symbols
- Criticality : Medium
- Expected risks : Nltk library drawback
- Dependencies : 3.3.2 upload subtitle

3.3.22 FR22

- Name : remove digits
- Description : It removes the digits from the subtitle given (e.g. 1, 2, 3,).
- Input : Text
- Output : Text without digits
- Criticality : Medium
- Expected risks : Nltk library drawback
- Dependencies : 3.3.2 upload subtitle

3.3.23 FR23

- Name : to lowercase
- Description : It converts the uppercase letters in the text lowercase letters.
- Input : Text
- Output : Text without uppercase letters
- Criticality : Medium
- Expected risks : Nltk library drawback
- Dependencies : 3.3.2 upload subtitle

3.3.24 FR24

- Name : getFileDetails
- Description : It counts the no of Lines, Words and Character to the given text.
- Input : Text
- Output : No of the words in the given Text
- Criticality : Medium
- Expected risks : wrong file name or directory
- Dependencies : 3.3.2 upload subtitle

3.3.25 FR25

- Name : add shot
- Description : It makes the user has the ability to add his own desired shot to the extracted trailer to meet his satisfaction.
- Input : time interval
- Output : add the shot to the trailer
- Criticality : Medium
- Expected risks : wrong time interval inserted
- Dependencies : 3.3.20 show recommended shots

3.3.26 FR26

- Name : delete shot
- Description : It makes the user has the ability to delete any shot from the automatically extracted trailer to meet his satisfaction.
- Input : time interval
- Output : delete the shot from the trailer
- Criticality : Medium
- Expected risks : wrong time interval inserted
- Dependencies : 3.3.35 show shots

3.3.27 FR27

- Name : upload movie
- Description : It lets the user upload his own .movie to start extracting trailer.
- Input : video file
- Output : Redirection to the preprocessing of the file
- Criticality : High
- Expected risks : Invalid file type or wrong extension
- Dependencies : 3.3.1 login

3.4 Interface Requirements

3.4.1 GUI

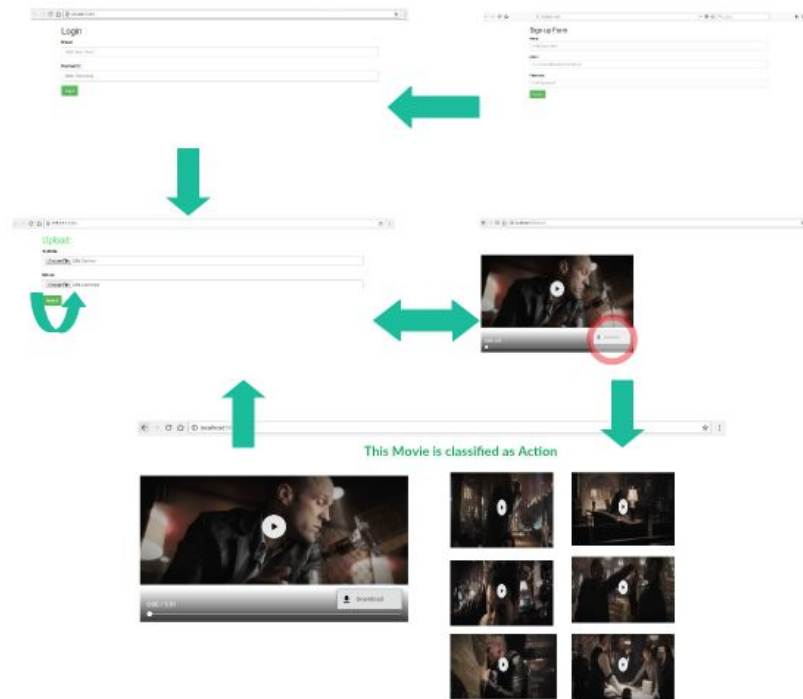


Figure 3.2: Website Wireframe.

3.4.2 API

- Natural Language Tool Kit Library (NLTK): it is used for the text (subtitles) preprocessing for filtration, tokenization and lemmatization.
- Operating System Library (OS): it is used for using the "walk" function.

3.5 Performance Requirements

The System should respond after uploading a certain file with the least time should be. System should be able to handle and process thousands of words in the given subtitle and the bag of words. When having a large SRT file inserted, the fine speed of the system is so important.

- Total Project compilation time:

File Size(KB)	RAM consumed(MB)
46	28.6
96	62
141	100

Table 3.1: Total Project compilation time

- Preprocessing Module compilation time:

File Size before(KB)	File Size after(KB)	Time consumed(ms)
46	7	4300
96	15	4800
141	28	5300

Table 3.2: Preprocessing Module compilation time

- TFIDF KNN Module compilation time:

File Size(KB)	Time consumed(s)
7	35
15	60
28	95

Table 3.3: TFIDF KNN Module compilation time

These statistics were taken on a machine whose RAM is 6.00 GB and Intel processor core i7 @2.4 GHz

3.6 Other non-functional attributes

3.6.1 Security

There is no certain security algorithm but Every user must have an account with user-name and password.(FR1)

3.6.2 Reliability

Classification (FR13) is done after the weighting is ensured by testing some subtitles with classified subtitles for making sure that there is no miss classification [13]. Bag-Of-

Words is extracted by preprocessing 10 movies of the same genre which are already classified. The system should give the best result when handling large SRT files. And if there a failure in a file or in the preprocessing we can get back to the last checkpoint as every file is

3.6.3 Portability

This system's functions and core is written on Python which is portable as it runs on many Unix variants, on the Mac, and on Windows and later [14]. And the system is a web application using HTML Responsive Web Design so it's portable on mobile phones and computer devices software connected to internet. (e.g. Android, Macintosh OS, Microsoft Windows , Linux)

3.6.4 Maintainability

The system is maintainable due to implementing and using the Model-View-Controller (MVC) software architectural pattern, with the decorative design pattern.

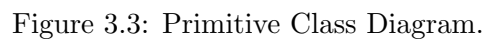
3.7 Preliminary Object-Oriented Domain Analysis

3.7.1 User

- Name : User
- List of superclass : None.
- List of subclass : None.
- Purpose : The user inputs (uploads) the subtitle file to get a trailer. - Collaborations : None.
- Attributes : ID,Username,Password.
- Operations : UploadSubtitle: which a user inputs a subtitle file to the website / Search-File: user searches for the files which were - uploaded before.
- Constraints : None.

3.7.2 Subtitle

- Name : Subtitle
- List of superclass : None.



- List of subclass : None.
- Purpose : This class is responsible for describing the files which are uploaded by the user to extract the trailer from.
- Collaborations : All the preprocessing functions (Lemmatizer,Tokenizer, Stopwords Removal) are applied on it and also weighting by TF IDF and KNN Classifier uses the subtitle files after preprocessing to return the genre of the subtitle.
- Attributes : ID,Name,Date.
- Operations : None
- Constraints : The subtitle files uploaded should be english subtitles.

3.7.3 Lemmatizer

- Name : Lemmatizer
- List of superclass : None.
- List of subclass : None.
- Purpose : This class is to return the part of speech of each word (Noun, Verb, Adjective..etc) to its original to take the same weight without repetition .
- Collaborations : None.
- Attributes : None.
- Operations : GetWordNet: To return each word into noun. /pos tag: gets the word and its lemmatization depending on the pos tagging of each.
- Constraints : None.

3.7.4 Tokenizer

- Name : Tokenizer
- List of superclass : None.
- List of subclass : None.
- Purpose : This class is to split the words in a file into tokens.
- Collaborations : None.
- Attributes : None.
- Operations : TokenizeText: To split the words and return them as tokens.
- Constraints : None.

3.7.5 TextFilter

- Name : TextFilter
- List of superclass : None.
- List of subclass : None.
- Purpose : This class is to filter the input file.
- Collaborations : None.
- Attributes : None.
- Operations : remove symbols: To remove unwanted symbols from file. /remove digits: To

remove digits and timestamps from the file. / to lower: To keep all the words in the file in lowercase.

- Constraints : None.

3.7.6 FrequencyCounter

- Name : FrequencyCounter.

- List of superclass : None.

- List of subclass : None.

- Purpose : This class is to count the number of words in the subtitle files and to count the frequency of each word in the file.

- Collaborations : None.

- Attributes : None.

- Operations : FileDetails:To count the number of words in a file. / WordCountFrequency: To count the frequency of each word in the file.

- Constraints : None.

3.7.7 TFIDF

- Name : TFIDF.

- List of superclass : None.

- List of subclass : None.

- Purpose : This class is to calculate the weight matrix according to TFIDF.

- Collaborations : KNN classification uses the TF IDF weights to classify the file and return the genre of the movie.

- Attributes : Countwords.

- Operations : Get TFValue: To calculate the term frequency and return it. / Get IDF-Value: To calculate the inverse document frequency. / GetTF IDF: To calculate the TF IDF value.

- Constraints : None.

3.7.8 KNNClassifier

- Name : KNNClassifier.
- List of superclass : None.
- List of subclass : None.
- Purpose : This class is to classify the subtitle file to get the genre of the movie.
- Collaborations : None.
- Attributes : WeightObject.
- Operations : GetSubtitle: Receives the preprocessed subtitle / weightClassifiedSubtitle: Determine the TF matrix of the keywords inside the preprocessed text file / calculateDistance: Applying the KNN euclidean distance between the TF-IDF weight matrix and the TF matrix of the keywords and getting the shortest distances and calculating the majority of the shortest distances according to the specified movie categories.
- Constraints : None.

3.7.9 SWRemoval

- Name : SWRemoval
- List of superclass : None.
- List of subclass : None.
- Purpose : This class is to remove the stopwords from a file.
- Collaborations : None.
- Attributes : None.
- Operations : RemoveStopWords: To remove the stopwords from the words returned after lemmatization and tokenization.
- Constraints : None.

3.8 Operational Scenarios

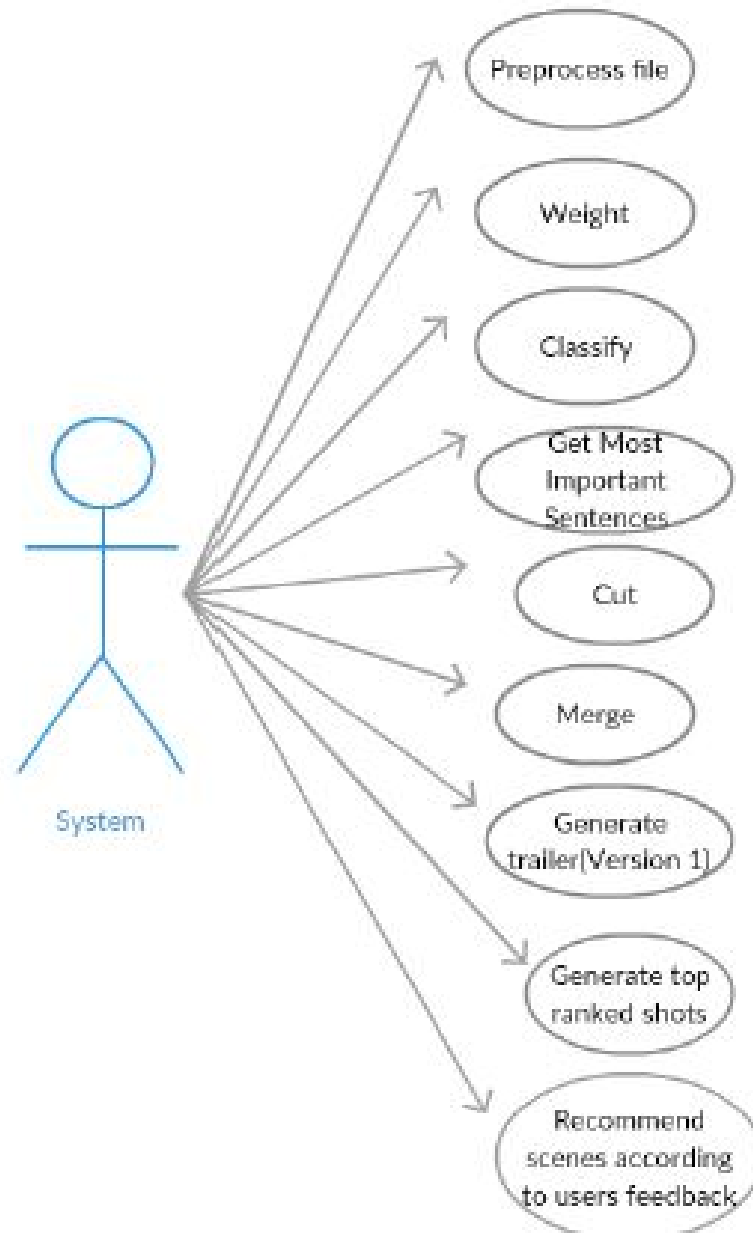


Figure 3.4: System Scenario.

3.8.1 System Scenario

Files are entered to the system to srt be preprocessed for filtration, tokenization and lemmatization. Then file is being weighed by TF-IDF to determine the highly weighted words. Then file is classified to which genre so that the trailer can be generated by knowing the timestamp of the highly weighted words to be splitted and merged together as the trailer then its returned to the user. In case the user wants to change anything about the trailer, he can edit it or send a feedback to the system, so the system analyze it and recommend top ranked scenes according to this feedback and after the user finishes editing, the system generate again the new trailer.

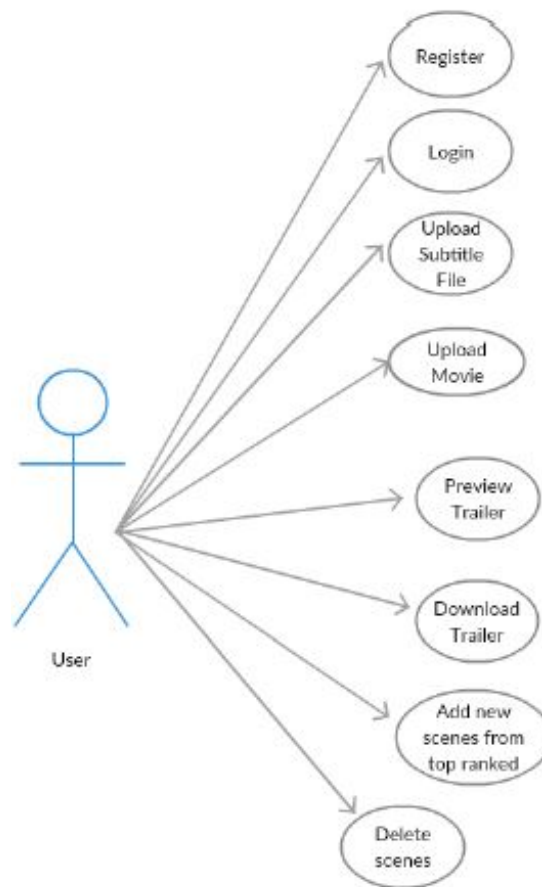


Figure 3.5: User Scenario.

3.8.2 User Scenario

In this system, the user must have an account so the first thing is registration to have a username and password to be able to login anytime. Each user after logging in can upload a movie subtitle file, an audio file and the movie itself so that the system can generate a trailer then user can preview and download the generated trailer. The user may have a point of view that he wants to see it in the trailer so he can edit in the generated trailer by adding new shots from the top ranked scenes appearing, replace or delete another or reorder the trailer as he likes. He also can add a specific sentence to be viewed in the trailer or a feedback so another scenes can be recommended to him that he can choose from.

3.9 Appendices

3.9.1 Definitions, Acronyms, Abbreviations

- NLTK: Natural Language Tool Kit.
- TF-IDF: Term Frequency Inverse Document Frequency.
- POS Tagging: Part-of-Speech Tagging.
- KNN: K-Nearest Neighbours Algorithm.

Chapter 4

System Software Design Document

4.1 Introduction

4.1.1 Purpose

This document is to describe the system architecture and how the system is going to work in details. This document is important because it explains the sequence of the system and the sequence of each function and actions, also, the design patterns used and how each one of them is going to be used and implemented in the system and explaining the detailed database for the whole system. Problem statement: Detecting and classifying movie subtitle file to generate a trailer according to analysis and movie category.

4.1.2 Scope

The goal of this system is to reduce time taken to edit movies and generate a trailer and to generate a trailer in an intelligent way. The benefit of this system is that editors are not going to take a lot of time watching a whole movie over and over again to decide which shots are going to be in a trailer, all they have to do is uploading a subtitle file and the system is going to analyze , return the most important and also the editor can add, reorder or put some preferences so the system recommend some shots according to them and by this system the time taken is for sure less than the time taken to watch a whole movie.

4.1.3 Overview

Generating a trailer is considered as the most important phase of producing a movie where it's the main advertising for the movie which have to be attractive to make the audience buy a ticket and watch the whole movie; and the process of making a trailer takes a lot of time according to [15]. This document proposes "S-Trailer", this system is to generate a trailer automatically from the subtitle file that is uploaded with the movie by a user. The system is based on classification and deep learning in which the system saves the shots chosen by the user to recommend some other shots from the movie according to the user's behavior. The shots that appears to the user depends on the movie genre returned from classification and some preferences entered by the user, he then can delete, re-arrange, add or edit the resulted shots.

4.1.4 Definitions and Acronyms

- NLTK: Natural Language Tool Kit.
- TF-IDF: Term Frequency Inverse Document Frequency.
- POS Tagging: Part-of-Speech Tagging.
- KNN: K-Nearest Neighbours Algorithm.

4.2 System Overview

This system is distributed into 3 phases to maintain the performance. The .srt phase is the training phase, this stage is the .srt one as the system starts with it to get the training dataset. Firstly system gets 10 movies subtitles, System starts pre-processing on these subtitles to filter them and get some keywords which will be named processed subtitles, then ranking these keywords takes place to rank which word is the most important to be used and which is least. After ranking system has a model for each movie genre to be used in the next phase. The second phase is processing phase which is the core of the program, in this stage the user has the ability to upload a SRT subtitles file and The movie required, and system takes this file to start pre-processing on it to get the filtered keywords that represents this movie. System starts classification by comparing this input subtitle with the other genres keywords that the system has to determine the genre of the input subtitle. Then ranking the keywords of the input subtitles depends on the important words of the

movie genre model to extract the influenced key phrases, system merges the scenes that contains these influenced key phrases then give it back to the user to submit his feedback and edits. The third and the final phase is the recommendation and output phase, this phase takes the user feedback and edits as its input to start analyzing this feedback then it provides the user with some recommendations to be added to users trailer, user chooses any of recommendations or not. Then after proceeding, the final movie trailer will be obtained.

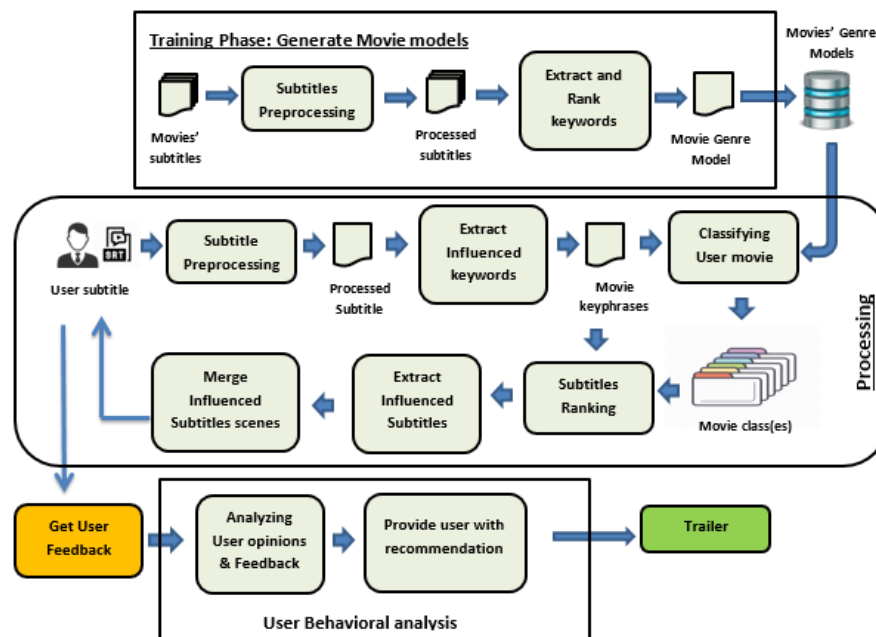


Figure 4.1: System Overview.

4.3 System Design

4.3.1 Architectural Design

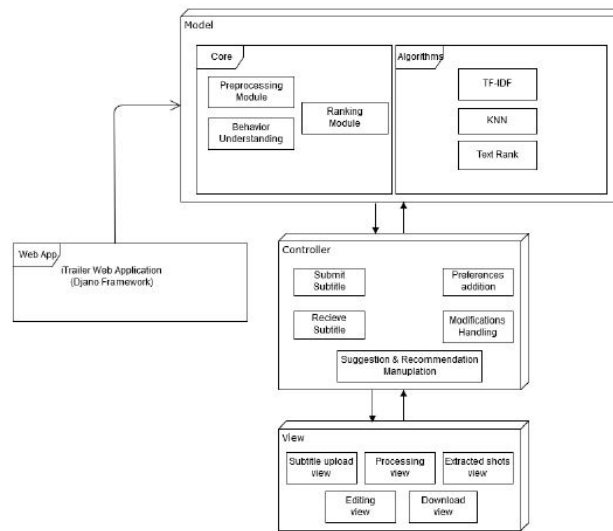


Figure 4.2: Architectural Model

4.3.1.1 Model

- Preprocessing module: It is the primary phase which consists of three sub-phases:

- Stop words removal: Natural language toolkit is used in this process to detect and remove the non-significant words like (and a an)
- Text tokenization: It is the way used to separate every significant word into a single token to substitute the sensitive data from the non-sensitive ones.
- Text lemmatization: The goal of stemming the subtitle file is to reduce inflectional forms

- Ranking module: The system inputs the extracted keywords (subtitle file post-processing) to the ranking module where Text-rank algorithm takes place. The goal of the ranking module is to merge the keywords with the original subtitle and rank from the most significant to the least significant sentences in the subtitle which means that it reaches the most important shots or scenes.

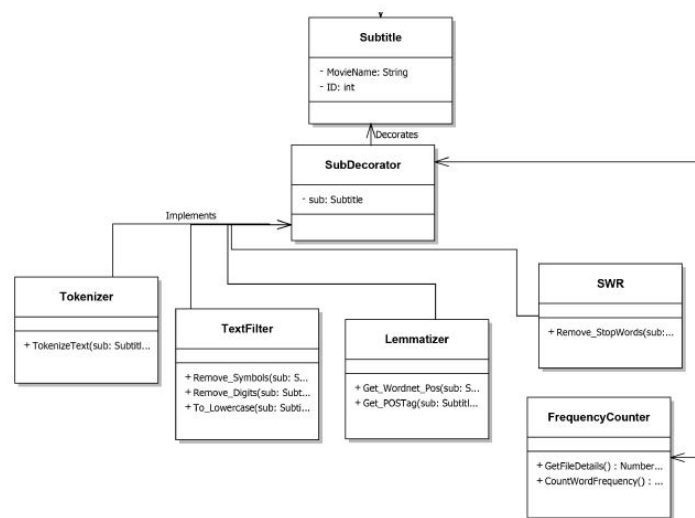


Figure 4.4: Decorator design pattern

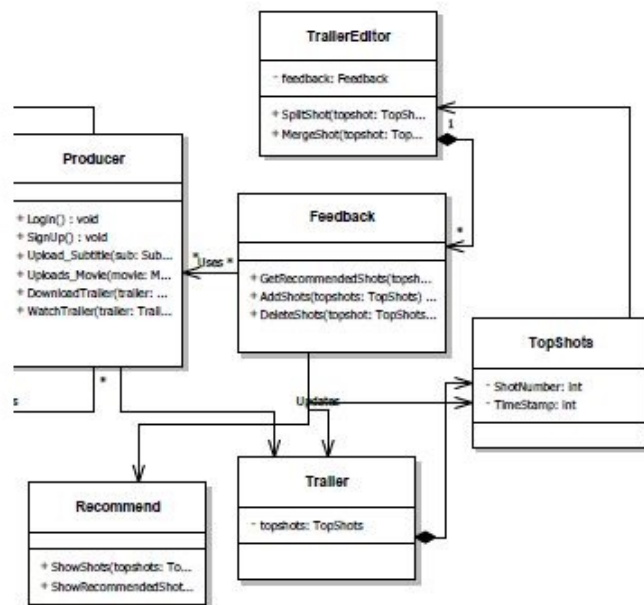


Figure 4.5: MVC design pattern

4.3.2.2 Sequence Diagram

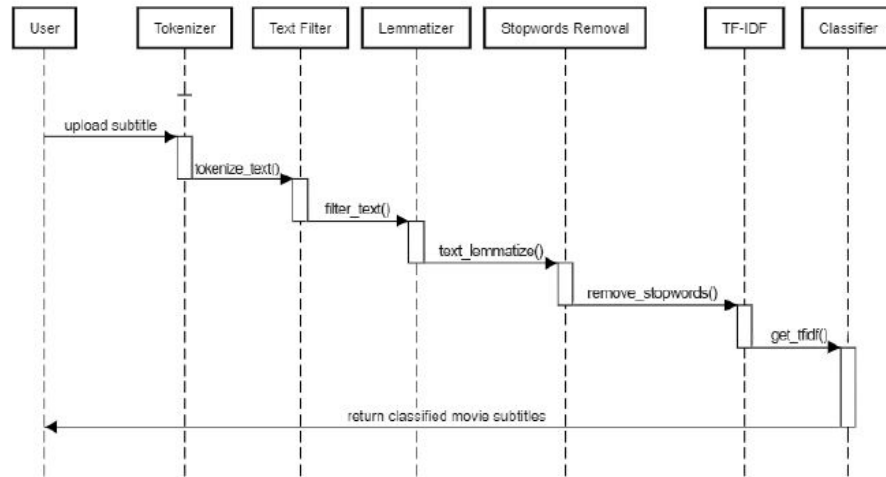


Figure 4.6: Sequence Diagram for The Classifier

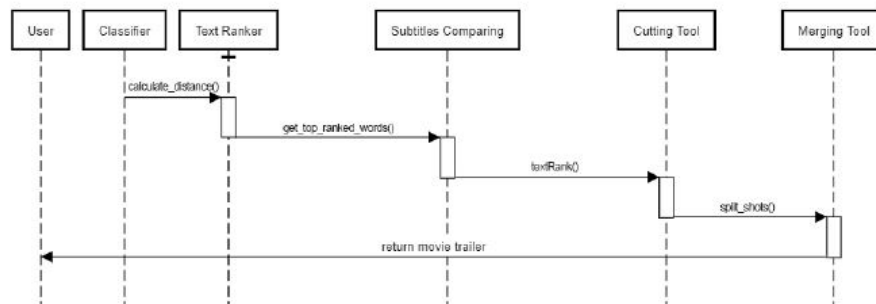


Figure 4.7: Sequence Diagram for The Ranking

4.3.3 Design Rationale

S-Trailer is a web application based on the MVC (Model-View-Controller) design pattern as we represented in this document. The main goal of using the MVC architecture is to provide the system with faster development processes also the entire model won't be affected with the future work.

4.4 Data Design

4.4.1 Data Description

- User: This table has the details of the user in which he uses to login to the system and identifies his privileges
- Subtitle file: This table has the movie name, timestamps and it stores the original file before preprocessing to get from the top ranked shots subtitles according to the timestamp.
- Behavior: It has the ID of each user to save his behavior while working with the system, so the system can recommend to him another shots according to this behavior and it will be deleted the next time he uses the system.

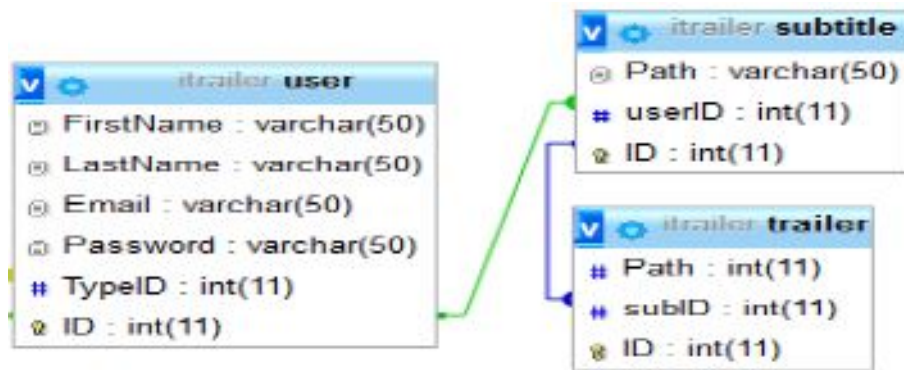


Figure 4.8: Database

4.5 Component Design

4.5.1 TF-IDF Term frequency/Inverse document frequency

Used to introduce a two aspect term frequency normalization scheme as it combines relative term-frequency weight and the term frequency normalization based on the document length.[19] The algorithm is divided into two separated equations performed on each and every word included in the document. The

$$w_{i,j} = tf_{i,j} \times \log \left(\frac{N}{df_i} \right)$$

Figure 4.9: TF-IDF equation for word (i) in document (j)

algorithm process is completed when the two equations of TF and IDF are combined together as shown above (figure 8) and then we a weighted-matrix for every word (i) among all documents (j) as shown below in Figure 9:

$$\begin{pmatrix} & T_1 & T_2 & \dots & T_t \\ D_1 & w_{11} & w_{21} & \dots & w_{t1} \\ D_2 & w_{12} & w_{22} & \dots & w_{t2} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ D_n & w_{1n} & w_{2n} & \dots & w_{tn} \end{pmatrix}$$

Figure 4.10: Weight matrix resulted from TF-IDF algorithm

4.5.2 TF-IDF Results

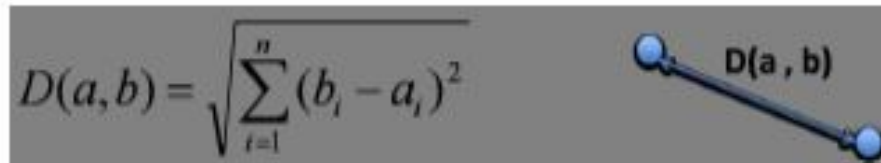
The algorithm showed an efficient weighting schemes tested on more than ten thousand subtitle file[16] and the efficiency of the weights were reflected later in the classification process.

4.5.3 KNN: K-Nearest Neighbor

KNN is a non-parametric algorithm used for classification and regression. In this software, we've used the KNN in text classification process (subtitle classification) where the output is a movie genre class. An object is classified by a majority vote of its neighbors and being assigned to the genre with the most neighbors. KNN algorithm has three different distance equation which are: Euclidean, Manhattan and Cosine Similarity distances.[17]

4.5.4 KNN Distance Equations

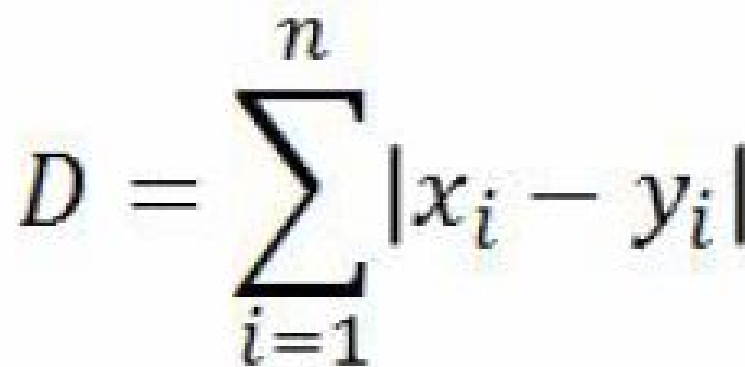
- Euclidean Distance It calculates the square root of the subtracted square of two different words from the given data. (Between point and B in Figure 10)



$$D(a,b) = \sqrt{\sum_{i=1}^n (b_i - a_i)^2}$$

Figure 4.11: Euclidean equation where I is the number of K neighbors

- Manhattan Distance It is the sum of the absolute values of the horizontal and vertical axes (each word and every document) in figure 11.



$$D = \sum_{i=1}^n |x_i - y_i|$$

Figure 4.12: Manhattan equation formula

4.5.5 KNN Classification Results

The KNN classifier algorithm was tested on a sample from a huge dataset (29,000 subtitle files from different movie genres) and the results have recorded an average accuracy of 79%.

4.5.6 Text Rank Algorithm

Graph-based ranking model enabled to be used on natural language texts, the main idea of the algorithm is that is built upon voting or recommendation as when a one vertex (word or key phrase) in the text file links to another vertex, its basically casting a vote for this vertex.[18] The higher the number of votes:The higher the importance of the vertex.[15]

4.6 Human Interface Design

4.6.1 Overview of User Interface

The applications .srt interface that user sees will be describing the system and how it work giving the user some hints how this system works and a button Create your Trailer now to make the user excited to click and enter the next page.After proceeding, the login page will appear to make user enter his username and password and asking if new user to sign up. when user signs in it redirects to the srt file upload page which allows the user to upload his .srt file and the movie to start pre-processing on then after the loading of the processing finishes the .srt generated trailer will be shown to the user. The trailer will be in front of the user with two choices even Finish which accepting the obtained trailer or Edit which redirects to another page. If the user chose Edit the page of editing and recommendations will appear, in this page the user has the ability to add,move or delete any of the generated shots, after editing any of the shots the system gives the user some recommendations of shots that might satisfy the user and his trailer. Finally, a page thanking the user for using this system.An extra page will be available only for the admin who can manipulate the bag-of-words of each genre as he can add, delete or update.

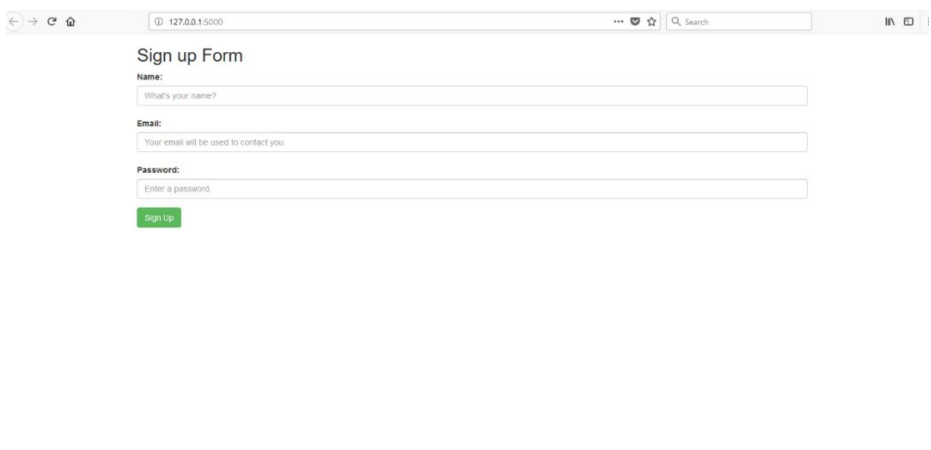


Figure 4.13: Sign Up Page

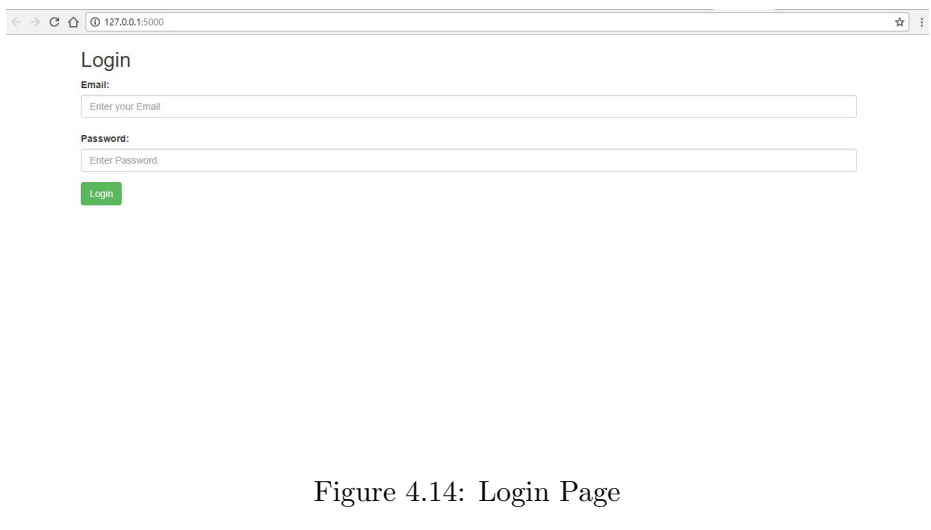


Figure 4.14: Login Page

4.6.2 Screen Images

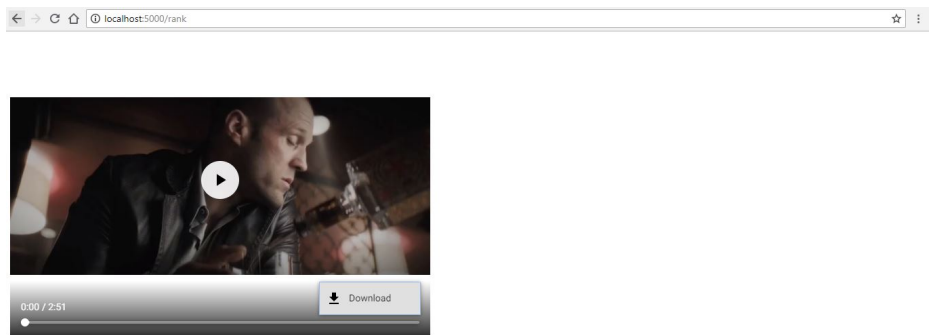


Figure 4.16: Resulted Trailer Page

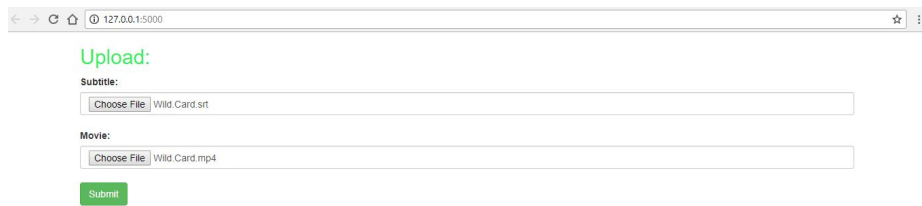


Figure 4.15: Upload Page

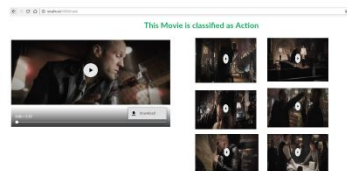


Figure 4.17: Recommended Shots Page

4.6.3 Screen Objects and Actions

- Figure 12 (Sign-up Page): this screen is for sign-up which makes user register to the system by filling his data in the text boxes then press done to create the new account.
- Figure 13 (Login Page) :in this screen there is only one button which is used to accept the username and password to redirect to the upload page.
- Figure 14 (Upload Page): in this screen there is only two fields to upload the srt subtitles file and the mp4 movie file.
- Figure 15 (Extracted Trailer Page): in this page there is the trailer which generated by the system, a button download and it redirects the user to start downloading immediately.
- Figure 16: in this screen the video appears with it's category and the recommended shots according to users choices.

4.7 Requirements Matrix

Requirement ID	Requirement Type	Requirement Description	Test Strategy	Source
FR1 Login	Required	It lets the user login with his username and password to start the system functions and the first will be the file uploading.	Valid username and password	Brainstorming
FR2 Upload Subtitle	Required	It lets the user upload his own .SRT file to start processing on this file.	Successful login	Brainstorming
FR3 get Wordnet pos	Required	It determines the wordnet part of speech of the normal part of speech.	Using classifier, if results were true	Brainstorming
FR4 text lemmatize	Required	It determines the origin of each word (lemmatization).	Using classifier, if results were true	Brainstorming
FR5 tokenize text	Required	It splits any kind of text into separate tokens.	Using classifier, if results were true	Brainstorming
FR6 Remove stop words	Required	It removes any stop word from the text as a kind of filtration.	Using classifier, if results were true	Brainstorming
FR7 count word frequency	Required	It counts the frequency of each word from the given text.	Using classifier, if weighted right the results will be right	Brainstorming
FR8 get IDF value	Required	Function to calculate the Inverse Document Value of a single word.	Using classifier, if weighted right the results will be right	Brainstorming
FR9 get TF value	Required	Function to calculate the Term Frequency of a single word.	Using classifier, if weighted right the results will be right	Brainstorming
FR10 get TFIDF	Required	Function to calculate the weight matrix (weight of each word = $TFvalue(w) * IDFvalue(w)$).	Using classifier, if weighted right the results will be right	Brainstorming
FR11 Weight Classified subtitle	Required	Determine the TF matrix of the keywords inside the preprocessed text file.	Using classifier	Brainstorming

Figure 4.18: Requirement Matrix

FR12 Calculate Distance	Required	It passes the extracted keywords to the TF-IDF module to get the TF-IDF weight matrix, Applying the KNN Euclidian distance between the TF-IDF weight matrix and the TF matrix of the keywords determined in the previous function and Getting the shortest distances and calculating the majority of the shortest distances according to the specified movie categories.	Classifier returns accurate results	Brainstorming
FR13 get top shots time intervals	Required	It gets the time intervals of the most important scenes in the movie from the top ranked words extracted.	Have the top ranked subtitles	Brainstorming
FR14 get top ranked words	Required	It gets the top words from the high frequency weighting of the keywords.	Return an accurate result after classifying and the important shots	Brainstorming
FR15 split shot	Required	It splits the extracted top shots time intervals from top ranked words in the subtitles _le from the movie video _le.		Brainstorming
FR16 merge shot	Required	It merges the extracted splitted shots time intervals from top ranked words in the subtitles _le to generate the system's automatic trailer.		Brainstorming
FR17 download trailer	Required	It allows the user to download his generated trailer from the given subtitles with the given soundtrack.	Downloaded successfully	Brainstorming

Figure 4.19: Requirement Matrix

FR18 get recommended shots	Required	It extracts the time intervals that matches with the user choices and things that related to the same shots to be recommended to the user.	They are close to user's behavior	Brainstorming
FR17 show recommended shots	Required	It allows the user to see the recommended shots depends on the search keys he entered in the search key.		Brainstorming
FR18 remove symbols	Required	It removes the symbols from the subtitle given (e.g. %, \$, !,).	Classification process is accurate	Brainstorming
FR19 remove digits	Required	It removes the digits from the subtitle given (e.g. 1, 2, 3,).	Classification process is accurate	Brainstorming
FR20 to lowercase	Required	It converts the uppercase letters in the text to lower case letters.	Classification process is accurate	Brainstorming
FR21 get File Details	Required	It counts the no of Lines, Words and Character to the given text.		Brainstorming
FR22 add shot	Required	It makes the user has the ability to add his own desired shot to the extracted trailer to meet his satisfaction.		According to interview with <u>Dr.Tamer Iqbal</u>
FR23 delete shot	Required	It makes the user has the ability to delete any shot from the automatically extracted trailer to meet his satisfaction.		According to interview with <u>Dr.Tamer Iqbal</u>
FR24 upload movie	Required	It lets the user upload his own. movie to start extracting trailer.	Successful login	Brainstorming

Figure 4.20: Requirement Matrix

Chapter 5

Evaluation

5.1 Introduction

After implementing all the main functionalities of the proposed system, the smart trailer system was passed through multiple experiments to test and identify the points of strength of the system proposed. The first experiment was mainly to test the main classifier of the system and compare its accuracy with the rest of the classifiers on the same dataset, the second experiment aims to compare the automated trailer generated from the system proposed with the manual trailer produced by the movie editors and clarify the common shots between both trailers. The third experiment goal was to get the trailer edited by a number of volunteers and compare it with our automated trailer along with the volunteers feedback on our proposed system.

5.2 Experiment 1 Movie Genre Classification

5.2.1 Goal

Decide which classifier gives the system the highest accuracy among the top four movie genres.

5.2.2 Movie Genres Tested

- 1- Action
- 2- Comedy

3- Fantasy

4- Romance

5.2.3 Task

First step is to receive the submitted subtitle file from the user, preprocess and filter it and finally pass it through the module of the classification to identify the genre of the target movie. We have examined our bag of words model with several classifier and the table in section 5.2.4 shows the results of the top three classifiers (top according to accuracies).

5.2.4 Results

Genre	KNN	CNN	ANN	SVM	Naive Bayes	Cosine Similarity	Decision Tree
Action	75.5%	10%	60%	50%	70%	70%	66%
Comedy	66.5%	50%	60%	20%	50%	60%	20%
Fantasy	47.5%	10%	30%	40%	40%	80%	20%
Romance	55.5%	40%	10%	20%	50%	90%	70%

Table 5.1: Classifiers Accuracy Table

5.3 Experiment 2 Precision and Recall with the manual trailer

5.3.1 Goals

- 1- Determine how the proposed system is far/close from the original trailer produced by the movie editors.
- 2- Measure the similarity between the automated trailer and the manual trailer according to the number of common shots.

5.3.2 Movies Tested

- 1- Wild Card (Action/Drama)
- 2- Endless Love (Romance)

3- The Hangover (Comedy)

5.3.3 Results

Movie	Precision 10 Shots	Precision 30 Shots	Recall 10 Shots	Recall 30 Shots
Wild Card	2/13	6/13	2/10	6/30
Endless Love	1/4	1/4	1/10	1/30
The HangoverII	0	3/5	0	3/30

Table 5.2: Precision and Recall Table

5.4 Experiment 3 - User Evaluation

5.4.1 Goal

The goal in this experiment is to prove that the proposed system takes less time to produce a movie trailer.

5.4.2 Task

Giving a 10 minutes movie to 10 video editors to produce a 30 seconds trailer from those 10 minutes.

5.4.3 Movie Tested

- Big Sick

5.4.4 Results

The editors took an average time 1.5 - 2 hours to generate those 30 seconds while our system only take 5 minutes to produce same trailer.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

We're presenting our system which is natural language processing model incorporated with machine learning techniques. This system is going to be used in the domain of movie production to produce a movie trailer in minutes rather than days or may be weeks. Through the presented phases which were described in details training phase, processing phase and recommendation phase. As we mentioned training phase is to filter and extract the unique words, processing phase is to get movie genre and produce an attractive trailer through ranking and the recommendation phase is to recommend shots according to users' behaviors while editing. According to experiments we succeeded in producing an attractive trailer that is good enough to be used in advertising.

6.2 Future Work

The future work is adding object recognition to this system, so that the scenes with explosions for example can be detected and added to the trailer to increase the attractiveness of the output and decrease the editing and the user interference.

Bibliography

- [1] J. Nessel and B. Cimpa, The movieoracle - content based movie recommendations, in Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011 IEEE/WIC/ACM International Conference on, 22-27 Aug. 2011, Lyon, France. IEEE, October 2011. [Online]. Available:<http://ieeexplore.ieee.org/document/6040879/>
- [2] S. S. Thomas, S. Gupta, and V. K. Subramanian, Perceptual video summarization: a new framework for video summarization, IEEE Transactions on Circuits and Systems for Video Technology, vol. 27, no. 8, pp. 17901802, 2016. [Online]. Available: [10.1109/TCSVT.2016.2556558](https://doi.org/10.1109/TCSVT.2016.2556558);<https://ieeexplore.ieee.org/document/7457239/>
- [3] H. Kim, I. Yoon, T. Kim, and J. Paik, Video summarization using feature dissimilarity, in Electronics, Information, and Communications (ICEIC), 2016 International Conference , 27-30 Jan. 2016, Da Nang, Vietnam. IEEE, September 2016. [Online]. Available: <https://ieeexplore.ieee.org/document/7562973/>
- [4] M. Otani, Y. Nakashima, T. Sato, and N. Yokoya, Video summarization using textual descriptions for authoring video blogs, Multimedia Tools and Applications, vol. 76, no. 9, 2017. [Online]. Available:<https://doi.org/10.1007/s11042-016-4061-3>;
<https://link.springer.com/article/10.1007/s11042-016-4061-3>
- [5] Y. Yuan, T. Mei, P. Cui, and W. Zhu, Video summarization by learning deep side semantic embedding, IEEE Transactions on Circuits and Systems for Video Technology, pp. 11, 2017. [Online].
Available: [10.1109/TCSVT.2017.2771247](https://doi.org/10.1109/TCSVT.2017.2771247);<https://ieeexplore.ieee.org/document/8101557/>
- [6] Y. Nakashima, M. Otani, E. Rahtu, J. Heikkil, and N. Yokoya, Video summarization using deep semantic features, in the 13th Asian Conference on Computer Vision (ACCV16), March 2017, Taipei, Taiwan. Springer International Publishing, 2017, pp. 361 377. [Online]. Available: <https://www.springerprofessional.de/en/videosummarization-using-deep-semantic-features/12134100>
- [7] R. Ren, H. Misra, and J. Jose. Semantic based adaptive movie summarisation. In S. Boll, Q. Tian, L. Zhang, Z. Zhang, and Y.-P. Chen, editors, Advances in Multimedia Mod-

eling, volume 5916 of Lecture Notes in Computer Science, pages 389399. Springer Berlin Heidelberg, 2010.

[8] A. Pavel, C. Reed, B. Hartmann, and B. Hartmann, Video dig ests: a browsable, skimmable format for informational lecture videos, in UISTUser Interface Software and Technology, SIGGRAPH ACM Special Interest Group on Computer Graphics and Interactive Techniques. NY, USA: ACM New York, October 2014, pp. 573582. [Online]. Available: <https://dl.acm.org/citation.cfm?id=2647400>.

[9] G. Irie, T. Satou, A. Kojima, T. Yamasaki, and K. Aizawa, Automatic trailer generation, in MM 10 Proceedings of the 18th ACM international conference on Multimedia, Firenze, Italy, SIGMULTIMEDIA ACM Special Interest Group on Multimedia. ACM New York, NY, October 2010, pp. 839842. [Online]. Available: <https://dl.acm.org/citation.cfm?id=1874092>.

[10] H. Zhou, T. Hermans, A. V. Karandikar, and J. M. Rehg, Movie genre classification via scene categorization, in MM 10 Proceedings of the 18th ACM international conference on Multimedia, October 25 - 29, 2010 , Firenze, Italy, SIGMULTIMEDIA ACM Special Interest Group on Multimedia. New York,USA: ACM New York, NY, October 2010, pp. 747750. [Online]. Available: <https://dl.acm.org/citation.cfm?id=1874068>.

[11] A. F. Smeaton, B. Lehan, N. E. OConnor, C. Brady, and G. Craig, Automatically selecting shots for action movie trailers, in MIR 06 Proceedings of the 8th ACM international workshop on Multimedia information retrieval, Santa Barbara, California, USA, SIGGRAPH ACM Special Interest Group on Computer Graphics and Interactive Techniques. New York,USA: ACM New York, NY, October 2006, pp. 231238. [Online]. Available: <https://dl.acm.org/citation.cfm?id=1178709>.

[12] Christopher Hooton, "We spoke to the people who make film trailers" 17 Jan 2017, <http://www.independent.co.uk/arts-entertainment/films/features/film-trailers-editors-interview-create-teasers-tv-spots-a7531076.html>.

[13] Kaggle "Datasets" <https://www.kaggle.com/datasets>.

[14] Python Software Foundation "General Python FAQ" 14 Nov 2017, <https://docs.python.org/3/faq/general.html#general-information>.

[15] K. Sankar and L. Sobha, An approach to text summarization, in CLIAWS3 09 Proceedings of the Third International Workshop on Cross Lingual Information Access: Addressing the Information Need of Multilingual Societies, June 04 - 04, 2009, Boulder, 3 Colorado. Stroudsburg, PA, USA: Association for Computational Linguistics, June 2009, pp. 5360. [Online]. Available: <https://dl.acm.org/citation.cfm?id=1572441>

- [16] Rounak Banik "The Movies Dataset", Metadata on over 45,000 movies.26 million ratings from over 270,000 users. November 2017. [Online]. Available: <https://www.kaggle.com/rounakbanik/the-movies-dataset>
- [17] I. Batal and M. Hauskrecht, Boosting knn text classification accuracy by using supervised term weighting schemes, in CIKM Conference on Information and Knowledge Management, November 02 - 06, 2009 , Hong Kong, China,SIGWEB ACM Special Interest Group on Hypertext, Hypermedia, and Web.NY,USA: ACM New York, November 2009, pp. 20412044. [Online]. Available: <https://dl.acm.org/citation.cfm?id=1646296>
- [18] R. Mihalcea and P. Tarau, Textrank: Bringing order into texts, in Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain.Association for Computational Linguistics, July 2004, pp. 404411. [Online].Available: <http://clair.eecs.umich.edu/aan/paper.php?paper id=W04-3252bibtex>
- [19] J. H. Paik, A novel tf-idf weighting scheme for effective ranking, in IR Research and Development in Information Retrieval, July 28 - August 01,2013 , Dublin, Ireland, SIGIR ACM Special Interest Group on Information Retrieval. NY,USA: ACM New York, July 2013, pp. 343 352. [Online]. Available: <https://dl.acm.org/citation.cfm?id=2484070>.
- [20] H. Xu, Y. Zhen, and H. Zha, Trailer generation via a point processbased visual attractiveness model, in Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, July 25 - 31,2015 , Buenos Aires, Argentina, The International Joint Conferences on Artificial Intelligence, Inc. (IJCAI). AAAI Press, July 2015, pp.21982204. [Online]. Available: <https://www.ijcai.org/Proceedings/15/Papers/311.pdf>.