Smart Trailer : Automatic generation of movie trailer using only subtitles

Mohammad Hesham
Faculty of Computer Science
Misr International University
Cairo, Egypt
Mohammad1407752@miuegypt.edu.eg

Bishoy Hani
Faculty of Computer Science
Misr International University
Cairo, Egypt
Bishoy1409973@miuegypt.edu.eg

Nour Fouad
Faculty of Computer Science
Misr International University
Cairo, Egypt
Nour1404954@miuegypt.edu.eg

Eslam Amer
Faculty of Computer Science
Misr International University
Cairo, Egypt
eslam.amer@miuegypt.edu.eg

***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 behaviors and opinions in order to adapt our system to provide users with relevant video scenes recommendations that match their preferences.

Index Terms—movie trailer; natural language processing; classification

I. INTRODUCTION

The widespread availability of high-speed Internet causes the video to become familiar information medium on the Web. The rapidly increasing rate of supply and demand of video content creates a compelling challenge for producers and consumers. Although the amounts of videos that are produced and indexed are growing at a rapid rate, the amount of time that is required to watch these videos still limited. This explains human inability to keep up with the massive amount of video data. Consequently, human needs an automatic video summarizing framework that generates effective video summaries.

Movie trailers can be seen as an application of video summarization; the purpose of a trailer is to summarize a whole movie (about 2-3 hours) into (2-3) minutes. The main obstacle of video summarization is how to find segments that are considered significant or interesting, and avoid other segments that are neither significant nor interesting to viewers.

There are many applications currently used by humans to select and merge the video scenes easily like Apple iMovie [1], Microsoft Windows Movie Maker [2], as well as online applications like Movavi [3], MakeWebVideo [4], and Morgan created by IBM. However, such applications usually require loading the whole movie to perform their task. Nevertheless, producers are also required to go through lots of phases which are watching the movie, selecting the best scenes that describe the movie, aggregating such scenes in some order,..etc.

Generating a trailer manually is considered a cumbersome and time-consuming task; according to Independent article [5]; a movie trailer could take about three to four months to be completed. Most of the companies that create movie trailers usually attempt to find attractive scenes through some keywords that they think are going to be the reason why people buy a ticket and watch a movie.

Amy Paval, et.al. introduced an approach in [6] to trim a video into partitions and providing short text summaries and thumbnails for each partition. Viewers can read and navigate to the partitions of interest through browsing the summary. However, there isn't an effective way in the context of movies as there is no way in which a trailer creator enters a word and get back the scenes with this word yet.

978-1-5386-5083-7/18/$31.00 ©2018 IEEE

Another approach proposed by Zhe Xu, et.al in [7] to generate a trailer automatically using frames and shots from the movie. The proposed system was given some movies trailers as a training set to learn some features. The resulted features are utilized when giving a new movie to produce a trailer as a result. However, still, it might not be accurate as they may not contain all needed important shots from that movie to be put in the trailer.

The negative impact of current approaches is that it required fully user involvement which causes and consumes great time and efforts taken by movie editors while working on a movie to generate a trailer.

In this paper, we propose a system that automatically generates a movie trailer using only its subtitle. The proposed system utilizes Natural Language Processing, Machine Learning, and Deep learning to analyze the text included in the subtitle file. The analysis includes classifying the movie into its corresponding genre(s), ranking significant keywords according to that genre(s). The system also utilizes PageRank algorithm proposed by [8] to rank influential subtitles to get their corresponding time frame. The system can adapt to users' opinions and behaviors as it can provide recommendations and suggestions with similar scenes based on users' interest in a particular type of scenes that matches their preferences.

This paper is organized as follows. Section II describes the related works. The proposed approach is presented in Section III. Initial results are presented in section IV, and finally, section V presents the conclusion and future work.

II. Related Works

In this section, a very brief overview of touching approaches is presented since the specific way of generating automatic movie trailers has only little-related work so far. One can say that area of automatic trailer generation is a rather untouched field of research. However, the more general task of summarizing video content is a wide field of research.

Konstantinos Bougiatiotis, et.al [10] demonstrated the extraction of topical representation of movies based on subtitles mining. In their work, they examined the existence of a similarity correlation between movie content and low-level textual features from respective subtitles. In their approach, they created a topic model browser for movies, allowing users to explore the different aspects of similarities between movies. However, the drawback of this system is that it gives the user the similarity between movies and which is the closest to the other, but it does not recommend what to watch depends on the input of the user and it does not mention the genre of the movie.

Amy Pavel, et.al [6] worked on creating a digest for videos that afford to browse and skimming by segmenting videos given to sections and giving short summaries of text for each section extracted. The users can navigate by reading the section summary and choose the section to get or access the current point or subject of the video.

The resulted system provides a decent authoring interface and exploring techniques for automatically segmenting and summarizing an input video from the user. The drawback of this system returns video partitions according to titles, chapters or sections but if the title is missed for any topic in the video, the system won't be able to summarize it accurately.

J. Nessel, et.al [11] proposed a system that recommends movies to the users by comparing user's predefined examples with the textual contents of movies. The system is developed in the context of recursively decidable languages and computable functions. However, the drawback of this system is that it relies only on extracting words from the user without using any extra preferences or opinions from the user except an example for a movie name or a word presented in a dialog.

Go Irie, et.al [12] presented a content-based trailer generation method named Vid2Trailer (V2T). V2T automatically creates impressive trailers from original movies by identifying impressive audiovisual components, as well as key symbols of the movie title such as the title logo and the theme music. Results showed more appropriateness of V2T approach compared to conventional tools. However, the processing effort of the V2T system considered too much as it tends to filter the speech after calculating important words of the whole movie. Nevertheless, this processing can be reduced if the calculations of top impacted keyword take place after filtering subtitles from unneeded words rather than calculating the whole subtitle file.

Howard Zhou, et.al [13] suggested a system that is based on scene categorization. The system demonstrated that a temporally-structured feature based on such intermediate level representation of scenes can help to improve the classification performance over the use of low- level visual features alone. Although the classification performance slightly enhanced, the drawback of this system uses a bag of visual words which in turn needs a huge storage to save a bag of visual words of each movie category.

Alan F. Smeaton, et.al [14] presents an approach which automatically selects shots from action movies in order to assist in the creation of trailers. The approach depends on shot boundary technique in order to generate the basic shot-based structure of a movie. The audio track of a movie is analyzed in order to detect the presence of the following categories: speech, music, silence, speech with background music and other audio. Nowadays movies contain a mixture of genres, therefore, any system that generates trailer should also reflect such mixture. The major drawback of Alan approach is that it is specific to action shots exist in movies. If the movie contains many genres, the system won't be able to generate a satisfactory trailer for it.

The approaches discussed in this section showed an effort in understanding textual content, audio-video contents, or both together. However, as movies and trailers exist in a variety of forms according to different cultural environments, personal preferences such approaches didn't care about such preferences or opinions.

III. Proposed System

Smart trailer or (s-trailer) as shown in Figure (1), has two main phases namely, training, and processing-output phase.

Each phase will be described in details in the following sections.

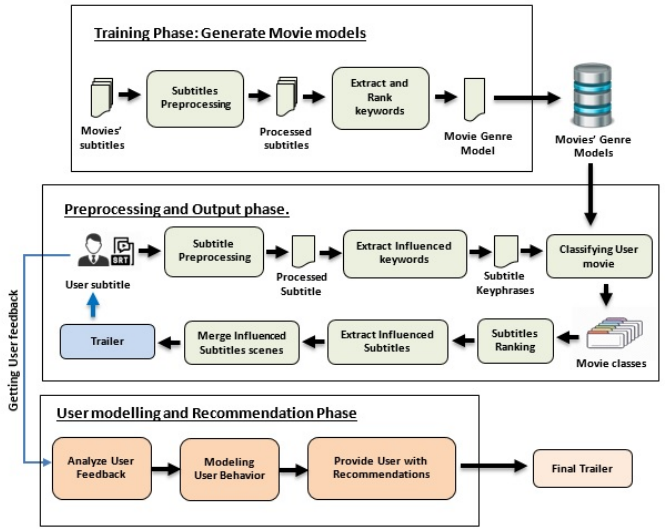

Figure 1. Smart-Trailer Model Architecture

A. Training Phase

The objective of the training phase is to create a bag of words or corpus that depicts the most frequent keywords or sentences that occur in each movie genre. This is done by collecting English subtitles for top-20 movies for each genre. The collection of top-20 movies were classified as a top for each genre according to IMDB Top rated movies by genre [15].

The subtitle file consists of four parts, all in text

1- A number indicating which subtitle it is in the sequence.
2- The time duration that tells when subtitle should appear on the screen, and when it disappears.
3- The subtitle itself.
4- A blank line indicating the start of a new subtitle.
Figure (2) is showing an example of sample subtitles:

```
1
00:02:17,440 – 00:02:20,375
Senator, we're making our final approach into
Coruscant.
2
00:02:20,476 – 00:02:22,501
Very good, Lieutenant.
```

Figure 2. Sample subtitle sequences

Where (1,2) indicating which subtitle is in the sequence, 00:02:17,440 – 00:02:20,375 depict the time duration that the subtitle should appear on the screen, and when it disappears, text "Senator, we're making our final approach into Coruscant" is the subtitle itself.
The subtitle files are going through preprocessing step to eliminate unnecessary trivial characters and words that are totally insignificant to the movie genre. The main focus is to extract significant words that describe the movie.

According to experimental observations, keywords that are annotated by humans are nouns [19-20]. We are giving concerns to extract all nouns and adjectives or any patterns like an adjective + noun and prune the rest.
The preprocessing step relied on part-of-speech tagger in NLTK library to find tags of interest that are marked as nouns or adjective.

In Training phase, we intend to build genres models, for each genre; build hierarchical n-grams of the distinct word and/or co-occurrence with other words in the processed documents along with their frequencies. This is done through employing the methodology presented in [16-17].
Keyphrases are ranked using our keyphrase ranking equation algorithm presented in [25] with minor modification.

$$Rank(i) = \log\left(p(i)\frac{TF_i + TI_i}{L}\right) \qquad (1)$$

Where *pi* is the position of dictionary entry *i*. The position is calculated as *(L–Ls)* where *L* is total lines in the document, *Ls* is the first sentence where dictionary entry *i* occurs. *TFi , TIi* denotes the term frequency, influence weight for dictionary entry *i* respectively.

The output of training phase is set of different genre-lists, each list contains a set of weighted words along with their co-occurrence with other words in the training text. Each list is considered as a signature for movie specific genre.
The final outcome is a set of lists are stored in a genre dictionary.

B. Processing-Output Phase

The processing phase considered the core phase of the proposed model. In this phase, the user provides the model with the subtitle file for the movie that he wants to create a trailer for. The user subtitle will be processed in a way similar to the training phase to generate subtitle keyphrases for user subtitle.

The generated keyphrases are used to detect the genre(s) of user movie. This is done through utilizing K-nearest neighbor (*KNN*) classifier where *K* is the number of genres stored in genre dictionary. Generally, movies can have several genres, such genres arranged in a descending order depending on number of scenes related to each genre. For example, according to IMDB classification, Titanic[1] movie has two genres (Drama, and Romance). Therefore, the role of classifier is to return ordered set of classification genre(s) along with their percentages that are more closely related to user movie subtitle (i.e., 60% - Drama, 40% - Romance) The values returned by classifier will be used as a guideline to the proposed system when it comes to filling the trailer with movie scenes; as it should fill scenes within the resulted classification percentages.

[1] http://www.imdb.com/title/tt0120338/

The proposed model utilizes PageRank algorithm proposed by [8] to rank influential subtitles to get their corresponding time frames.

This ranking step considered one of the core steps in that phase. In the ranking module, we initially build a graph of subtitle sequences through building an adjacency matrix $N \times N$, where N denoted the number of sequences in the subtitle file. The values of rows and columns in the matrix are calculated using the following equation:

$$A(i,j) = \begin{cases} 0 & if\ (i = j) \\ Cosine\ Similarity\ (i,j) & if\ (i \neq j) \end{cases}$$

Where $A(i,j)$ denotes the item of the adjacency matrix, and cosine similarity measure the similarity between two sequences in the subtitle file. The value of $A(i,j)$ will depict the degree of similarity between two subtitles [16]. The resultant matrix represents vertices and edges linking them associated with some weights.

When the adjacency matrix become ready, the role of PageRank algorithm comes to rank nodes according to their popularity.

The output if the rank module is a set of ranked sequences. The proposed system selects the *top-K* sequences to represent the trailer scenes. The value of *K* is depending on the time duration required for the trailer.

The time frames of *top-K* sequences are passed to video editing library in python which fetch the corresponding scenes from original movie. Such scenes are aggregated, merged, and passed to the user as a final trailer.

C. User modelling and Recommendation Phase

In this phase, we aggregate user feedback and opinions about initial trailer to provide him/her with suggestions and recommendations to produce the final trailer. User can edit the trailer to add or modify sequence of scenes. The purpose of this phase is to build a recommender system that capture user behavior to provide the user with scenes from the movie that matches his preferences. This phase is not finished yet in this version of paper.

IV. EXPERMENTATION AND RESULTS

In order to prove the validity of the proposed framework. The resulted bag-of-words were tested against kaggle movie data-set downloaded from kaggle [18]. It contains about 5000 movies of different genres. Each movie in the dataset provided with its IMDB associated genre(s).

The first outcome of the proposed framework is a set of lists that contains set of weighted words and phrases for each movie genre. Through utilizing those bag of words, the proposed framework can classify any movie subtitle to its corresponding genre(s). The following table (Table 1) presents the classification accuracy for different movie genre resulted from the proposed framework

Table 1: Evaluating S-trailer classification accuracy

Genre	Classification Accuracy
Action	89%
Comedy	75%
Drama	72%
Romance	60%
Fantasy	55%

In our experiment, the order of genre is taken into consideration when we want to evaluate the classification accuracy. For example, if the movie genre is: Action, Crime, and Romance; the classifier should return the same order of genre appearance or the movie is considered as misclassified.

The results can be considered as a seed corpus for movies' classification, hence there is no standard golden corpus to be used to classify a movie.

The generated Trailer evaluation will be based on Precession, recall, and F-measure metrics:

Where PRECISION is the ratio of the number of relevant scenes retrieved to the total number of original trailer scenes [21-22], it is calculated as:

$$Precision = \frac{Number\ of\ relevant\ scenes\ retrieved}{total\ number\ of\ original\ trailer\ scenes}$$

RECALL is the ratio of the number of correct or relevant scenes retrieved to the total number of relevant scenes indexed in the generated trailer database [21], it is calculated as:

$$Recall = \frac{Number\ of\ relevant\ scenes\ retrieved}{total\ number\ of\ generated\ trailer\ scenes}$$

F-MEASURE is the weighted harmonic mean of precision and recall [21], it is calculated as:

$$F - measure = \frac{2\ x\ Precesion\ x\ Recall}{Precesion + Recall}$$

For the purpose of evaluation, 50 movies are used to evaluate the performance of the proposed model. Table (2) show the average performance of S-trailer in retrieving similar scenes presents in the original trailer.

Table 2: Evaluating S-trailer Performance

Genre	Average Precision	Average Recall	Average F-measure
Action	0.38	0.60	0.47
Comedy	0.35	0.42	0.38
Drama	0.44	0.46	0.45
Romance	0.31	0.34	0.32
Fantasy	0.26	0.30	0.32

Experimental results showing a promising result returned by S-trailer. A major drawback of the proposed method is that, it cannot fetch silence senses, which are scenes where there is no speech in it. We noted that producers of original trailers tend to use such silence scenes for the purpose of attraction or surprising. However, such drawback will be pruned in the future work.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed S-Trailer framework which is a natural language processing model integrated with machine learning techniques. The framework showed how the model can be successfully used in the field of movie marketing throughout the phases mentioned before to extract an attractive trailer for the audience. The proposed model builds a golden corpus that can be used to classify movies. The major contribution of the proposed model is its ability to generate a trailer without any human intervention with an accuracy degree up to 47%. Concerning the future work planned, we aim to model the silence scenes to enhance the efficiency of the proposed model. Also, we also intend to build a recommendation module that capture user behavior, therefore it can suggest user with scenes that matches his preferences.

REFERENCES

[1] iMovie - Apple. https://www.apple.com/lae/imovie/

[2] Windows Movie Maker https://www.windowsmovie-maker.org/

[3] Movie Trailer Maker—How to Make a Movie Trailer Movavi. https://www.movavi.com/support/how-to/how-tomake-a-movie-trailer.html.

[4] Create Your Own Movie Trailer With Our Online Video Maker."https://www.makewebvideo.com/en/make/movie-trailervideo.

[5] The Independent. "We spoke to the people who make film trailers " 17 Jan.2017, http://www.independent.co.uk/artsentertainment/films/features/film-trailers-editors-interviewcreate-teasers-tv-spots-a7531076.html.

[6] 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. 573-582.

[7] Z. Xu and Y. Zhang, Automatic generated recommendation for movie trailers, in Broadband Multimedia Systems and Broadcasting (BMSB), 5-7 June 2013, London, UK. IEEE, October 2013.

[8] Ying Ding, et.al. PageRank for Ranking Authors in Co-citation Networks. JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE AND TECHNOLOGY, 60(11):2229–2243, 2009

[9] K. Bougiatiotis and T. Giannakopoulos, Content representation and similarity of movies based on topic extraction from subtitles, in SETN 16 Proceedings of the 9th Hellenic Conference on Artificial Intelligence, May 18 - 20, 2016.

[10] R. Ren, H. Misra, and J. Jose. Semantic based adaptive movie summarization. In S. Boll, Q. Tian,L. Zhang, Z. Zhang, and Y.-P. Chen, editors, Advances in Multimedia Modeling, volume 5916 of Lecture Notes in Computer Science, pages 389-399.Springer Berlin Heidelberg, 2010.

[11] 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.

[12] 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. 839-842

[13] 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. 747-750

[14] A. F. Smeaton, B. Lehane, 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. 231-238

[15] www.imdb.com/

[16] Eslam Amer Enhancing Efficiency of Web Search Engines through Ontology Learning from unstructured information sources, Proceeding of 16th IEEE International conference of Information Integration and Reuse (IRI2015), PP.542- 549, 13-15 August 2015. San Francisco, USA.

[17] Aliaa A.A. Youssif, Atef Z. Ghalwash, and Eslam Amer. KPE: An Automatic Keyphrase Extraction Algorithm, Proceeding of IEEE International Conference on Information Systems and Computational Intelligence (ICISCI 2011), pp. 103 -107, 2011.

[18] "Kaggle." https://www.kaggle.com/

[19] Hulth, A.: Improved automatic keyword extraction given more linguistic knowledge. In: Collins, M., Steedman, M. (eds.) Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, pp. 216 –223 (2003).

[20] Wan, X., Xiao, J.: Single document keyphrase extraction using neighborhood knowledge. In: Proceedings of the 23rd National Conference on Artificial intelligence, AAAI 2008, vol. 2, pp. 855 –860. AAAI Press (2008)

[21] Amer, Eslam, and Khaled Foad. "Akea: an Arabic keyphrase extraction algorithm." In International Conference on Advanced Intelligent Systems and Informatics, pp. 137-146. Springer, Cham, 2016

[22] Amer, Eslam, and Khaled M. Fouad. "Keyphrase extraction methodology from short abstracts of medical documents." In Biomedical Engineering Conference (CIBEC), 2016 8th Cairo International, pp. 23-26. IEEE, 2016.