Segmentation and Labeling of Video Checkpoints

Yoshee Jain*
yosheej2@illinois.edu
Siebel School of Computing and Data Science, University of Illinois
Urbana, IL, USA

Ronit Anandani*
ronita2@illinois.edu
Siebel School of Computing and Data Science, University of Illinois
Urbana, IL, USA

Arpit Bansal*
arpitb2@illinois.edu
Siebel School of Computing and Data Science, University of Illinois
Urbana, IL, USA

Jiya Chachan*
chachan2@illinois.edu
Siebel School of Computing and Data Science, University of Illinois
Urbana, IL, USA

ACM Reference Format:
Yoshee Jain, Ronit Anandani, Arpit Bansal, and Jiya Chachan. 2024. Segmentation and Labeling of Video Checkpoints. In . ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Summary

The increase in the usage of video content in the digital era, with the introduction of largely online and recorded classes, has revealed a number of challenges for students in effectively browsing and retrieving meaningful information from long videos. In this project, a video-summarizing system was designed, which used video captions to identify topic changes and label sections in the video, enabling learners to navigate the video based on the assigned labels. The YouTube8M dataset was used for training and testing, where a sample of videos was obtained, and pre-processing was performed using stemming, inverse-document frequency, and other methods. After data preparation, unigram LM was created for the transcript, the transcript of the video was splitted into segments using a pre-defined threshold and then unigram LMs were created for each segment. Then, the top-*n* words for a segment were identified by maximizing the probability of the topic word being in the specific time segment (using the unigram language model of the segment) while minimizing the probability of it capturing the topic of the entire video (using the unigram language model of the entire transcript). Using these words, an LLM (OpenAI's ChatGPT API (GPT-4o-mini)) was queried to generate a coherent topic sentence. This analysis was performed, and a CSV file was created to map videos to their labels. An evaluation of these generated labels was conducted against the annotated labels in the YouTube 8M dataset using metrics such as BLEU and ROUGE, and the model's performance was reported. The complete code repository is available on GitHub[1]. The strengths and limitations of the implementation, along with key areas for future improvement, were also discussed.

2 Introduction

Videos have become a major medium of education, entertainment, and communication. Various websites, such as YouTube and MediaSpace, related to education contain immense amounts of video content on different topics. However, it is difficult for users to access specific information or get an overview in a short amount of time. Traditional methods of summarization rely on intense manual annotation, which is labor-intensive. Captions and transcripts provide textual information about words spoken in a video and are considered a valuable resource for understanding the content of the video. The project detects topic changes within the video using the captions and label different segments. The system finally labels the video with a meaningful topic sentence to enable users to navigate and access the content easily.

3 Description

In this section, we describe the steps that we used to build our model including the tech stack. In general, all code was written in Python. Figure 1 summarizes our approach starting from data collection to model evaluation.

3.1 Data Collection

For the video summarization project, the YouTube8M dataset was utilized along with captions retrieved via the YouTubeTranscriptAPI to access pre-segmented videos and their transcripts. This dataset enabled the avoidance of the labor-intensive manual segmentation process, allowing a focus on applying text information retrieval techniques to label pre-defined video segments..

3.2 Data Preprocessing

A two-stage pipeline was implemented for pre-processing the text in the transcripts to achieve higher accuracy in the model's implementation:

(1) The text was preprocessed by removing stop words and punctuation, applying stemming and tokenizing the transcripts to reduce noise and preserve semantically important information.

[*] All authors contributed equally to this research.

[1] https://github.com/yosheejain/CS410-topic-summarization-and-labeling

CS 410 Final Project, December 2024, Champaign, IL, USA
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

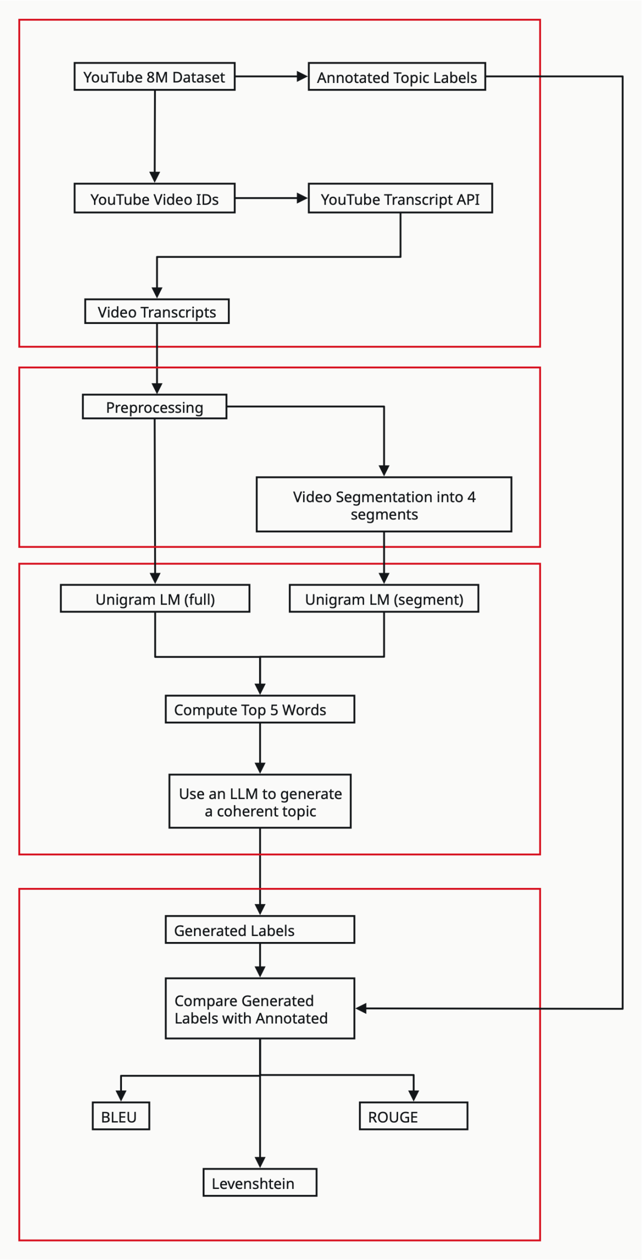

Figure 1: Details of the model and the pipeline including data collection, pre.

(2) Preprocessing was also implemented using inverse document frequency within each transcript. This approach allowed for the removal of words that were common throughout the entire video, as they would not accurately capture the specific topic of a particular video segment.

(3) The data was organized into a structured table mapping each video's ID to its cleaned transcript, annotated labels, and a placeholder for the generated labels. This structure facilitated the implementation of the model and evaluation strategies.

3.3 Data Preparation

For each video, a threshold of four segments was manually defined. The video was split into four equal segments to proceed with the model implementation. From that point, a dictionary was maintained for each video, mapping the complete transcript of the video to a list of the transcripts of its four segments.

3.4 Model Implementation for Text Summarization

A multi-stage pipeline was utilized for the model described below:

(1) For each video, a unigram language model was created, referred to as Full-LM from here on, using the complete transcript. This unigram language model represented the words used both frequently and infrequently throughout the video..

(2) Each segment transcript was accessed, and a unigram language model was created for each segment, referred to as Segment-LM. This model captured the frequencies of words used specifically within that segment.

(3) A threshold was then manually defined to choose the top n words which described the segment. For this experiment, top five words were used.

(4) For getting the top five words, the following pipeline was implemented that applied an algorithm designed to define a metric evaluating both the Full-LM and Segment-LM.

(5) Every word in the transcript of the segment was iterated through, and the probability of its occurrence was calculated both throughout the video using the Full-LM and within the segment using the Segment-LM. The objective was to maximize the probability from the Segment-LM while minimizing the probability from the Full-LM. The following formula was used for this calculation:

$$\text{Score} = -\text{Probability}_{\text{Full-LM}} + \text{Probability}_{\text{Segment-LM}}$$

(6) The top five words were then chosen for each segment as representative words for its label.

(7) At this point, a dictionary of keys were maintained as full transcripts and values as another dictionary. The second dictionary mapped each segment to each five descriptive words.

3.5 Label Generation

The dictionary of dictionaries was iterated through to access each set of descriptive labels for each segment of every video. OpenAI's ChatGPT API (GPT-4o-mini)[2] model was used to generate a cohesive topic sentence from the descriptive words.

To ensure scalability, we first tried using the Mistral model, which is less computationally heavy than some larger LLMs. We also thought that Mistral would capture the topic better because its open-source and can generate uncensored data contributing to the generalizability of our approach across domains. But, it did not

[2] https://openai.com/index/hello-gpt-4o/

generate accurate topic sentences as there was a lot of noise in the data, so we used the Open AI model for more accurate results.

At the end of this pipeline, each video was mapped to its segments, with each segment assigned a cohesive topic label describing its content.

4 Evaluation

At this stage, the generated labels for the videos in the YouTube 8M dataset were collected. The topic labels for each video were retrieved from the dataset, and semantic analysis was performed to compare the content of the generated labels with the annotated labels.

4.1 Metric Details

Three metrics were used for the semantic analysis. The overall model performance and accuracy will be reported in 4.2. Below, the three metrics are described, along with the rationale for their use in semantic similarity analysis..

(1) Edit Distance, specifically Levenshtein Distance: This metric provides a direct measure of how similar two text strings are.
(2) BLEU Scores: BLEU is widely used in natural language processing for comparing the similarity of machine-generated text to human-written reference text. It captures the overlap in phrases and is especially effective in ensuring that the generated labels preserve the content and structure of the ground truth labels.
(3) ROUGE Scores: ROUGE is commonly used for evaluating summarization and translation tasks, making it ideal for semantic similarity tasks. It emphasizes coverage, ensuring that the generated labels capture as much information as possible from the ground truth labels.

4.2 Implementation Details

A two-stage pipeline was implemented to report results and assess the model's performance:

(1) The mean scores of Levenshtein, BLEU, and ROUGE were first calculated and presented in Figure 2.
(2) The similarity scores were then converted into a boolean variable to indicate whether the text was similar to the label or not. A manually set threshold of scores > 0.4 was used, with values above the threshold labeled as $True$ and others as $False$.
(3) Based on these boolean results, the model's performance was evaluated and reported using $F1$-Score, Precision, and Recall accuracies for each metric as shown in Figure 3.

5 Discussion

5.1 Model Performance and Results

We used three metrics, namely Levenshtein distance, BLEU, and ROUGE scores for our comparative semantic analysis of the generated labels with the annotated labels.

Figure 2 is a histogram with the mean scores of BLEU, ROUGE and Levenshtein between annotated and generated labels. We see that across metrics, the mean similarity is lower than 0.3. In contrast to BLEU and ROUGE, we see a higher Levenshtein distance between the labels which can be attributed to two key reasons: (1) We asked ChatGPT to use the top few words to generate topic sentence in a few phrases leading to a higher chance of it being structurally similar (2) The annotated labels also broadly classify the video (for example game video or educational video) and ChatGPT is more likely to use common words leading to a higher chance of structural similarity.

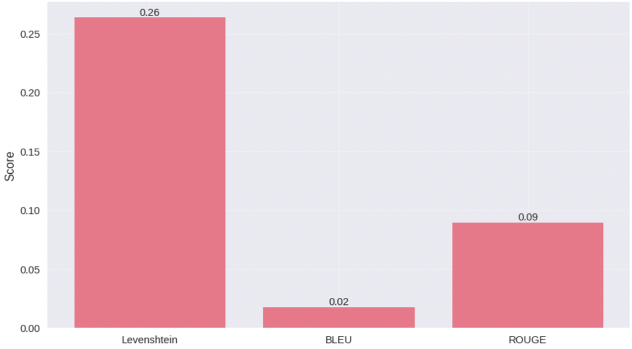

Figure 2: Mean scores of the semantic similarity between the generated and annotated labels using Levenshtein distance, BLEU, and ROUGE scores.

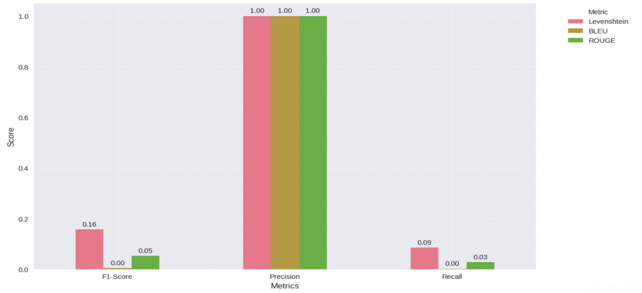

Figure 3: F-1, precision, and recall scores of the semantic similarity between the generated and annotated labels using Levenshtein distance, BLEU, and ROUGE scores.

Figure 3 describes the histograms of the F1, Precision, and Recall scores of BLEU, ROUGE, and Levenshtein between annotated and generated labels. We note low scores for the metrics except for precision. Precision remains constant because the system is accurate in a limited number of cases, but misses broader matches leading to lack of generalizability.

5.2 Limitations and Future Work

5.2.1 *Data Collection.*

First, the dataset provides only encoded identifiers for YouTube videos, so we had to decode these to retrieve the actual video IDs. This added an additional preprocessing step to access the necessary video transcripts and captions. Second, the dataset uses word indexes to represent vocabulary, requiring us to reverse search these indexes to map them back to the actual words. Other datasets that reduce the tedium in data collection would optimize the pipeline further.

5.2.2 Topic Summarization. A threshold for splitting the videos was pre-defined to simplify computation and focus on implementing class-specific concepts. However, these pre-defined thresholds may not effectively capture topic changes in realistic scenarios. A more dynamic approach to splitting videos is needed and could be explored as part of future work. One potential idea involves detecting topic changes using term frequency analysis. By identifying keywords that occur more frequently in specific segments, it would be possible to determine coherent topical segments within the video.

5.2.3 Evaluation. It was observed that while using edit distance for computing semantic similarity, Levenshtein distance is more suited for measuring syntactic similarity. It computes the distance, or the number of changes needed, to convert one string to the other. English is a diverse language where there are multiple ways of communicating one subject. Thus, this method of evaluation did not capture the semantic differences between the labels. Using BLEU and ROUGE was more accurate and gave more insight into the performance of the model.

6 Conclusion

The results of this project demonstrate a promising step toward automated segmentation and labeling of video content using textual information from transcripts. Although the model's performance is not yet fully optimized, the initial results are encouraging, showing that even with a limited number of high-probability keywords, meaningful topic labels can be generated. This highlights the potential for scaling and refining the approach to improve performance and applicability across diverse datasets.

Future enhancements, such as dynamic segmentation and advanced measures of semantic similarity, could further increase the system's ability to identify and label topics effectively. This makes it a valuable tool for navigating and summarizing video content in education and beyond.