Automatic Curation of Content Tables for Educational Videos

Arpan Mukherjee, Shubhi Tiwari, Tanya Chowdhury, Tanmoy Chakraborty
IIIT-Delhi, India
{arpan17007,shubhi17057,tanya14109,tanmoy}@iiitd.ac.in

ABSTRACT

Traditional forms of education are increasingly being replaced by online forms of learning. With many degrees being awarded without the requirement of co-location, it becomes necessary to build tools to enhance online learning interfaces. Online educational videos are often long and do not have enough metadata. Viewers trying to learn about a particular topic have to go through the entire video to find suitable content. We present a novel architecture to curate content tables for educational videos. We harvest text and acoustic properties of the videos to form a hierarchical content table (similar to a table of contents available in a textbook). We allow users to browse the video smartly by skipping to a particular portion rather than going through the entire video. We consider other text-based approaches as our baselines. We find that our approach beats the macro F1-score and micro F1-score of baseline by 39.45% and 35.76% respectively. We present our demo as an independent web page where the user can paste the URL of the video to obtain a generated hierarchical table of contents and navigate to the required content. In the spirit of reproducibility, we make our code public at https://goo.gl/Qzku9d and provide a screen cast to be viewed at https://goo.gl/4HSV1v.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**; • **Applied computing** → *Education*.

KEYWORDS

index curation; Youtube; video tagging; information retrieval

ACM Reference Format:
Arpan Mukherjee, Shubhi Tiwari, Tanya Chowdhury, Tanmoy Chakraborty. 2019. Automatic Curation of Content Tables for Educational Videos. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331400

1 INTRODUCTION

In the last couple of years, the popularity of online education has increased significantly. Video hosting forums like Youtube, Coursera and NPTEL make a rich information base for people willing to study a particular subject. Reputed universities are giving options of online degrees making it possible for a large international audience to take up their courses. It has enabled education to reach remote areas where it is difficult to have a traditional school set up. It has significantly reduced the cost of education making it affordable to a large section of the society.

Motivation: There are hundreds of videos on some popular topics making it difficult for a user to reach to the video which best suits his requirement. Also every user has a different requirement of depth while understanding a topic. While one user may be looking for an introductory course, another may be pursuing advanced research in the topic. But both users end up searching for videos with the same query. In such a case, a content table gives quick insight into the contents of the video and its extent of detail.

These videos are often classroom recordings and are an hour or longer in duration. They may be filled with discussions irrelevant to the video topic. For example, often a professor discusses the homework, upcoming deadlines and mid term results. A content table allows a user to skip unnecessary material and jump to the required section.

Often a user starts a video with the pursuit of gaining knowledge on a particular topic. In such cases, it becomes increasingly tiresome for a viewer to linearly search and navigate through the video to understand its contents. Content tables provide a summary of the video at a glance. This cuts down the time spent by the user browsing online, significantly. It thus becomes a necessary task for video hosting forums to have in-built content tables for educational videos.

Related Work: Attempts to automatically generate content tables from educational videos are less in number. Recently, Mahapatra et al. [7] used visual, audio and text features to generate a table of contents built on top of phrases. They identified key frames in the videos and used an Optical Character Reader (OCR) to retrieve text in bold/larger font from projected slides. They also used a Speech to Text - Audio Speech Recognition (ASR) module to identify important phrases. They introduced a phrase cloud based moderation approach. However, it focuses only for lectures where slides are the primary mode of communication. Previously, Biswas et al. [2] defined and quantified word saliency for visual words extracted from the slides and spoken words obtained from the speech transcript. They then ranked these words and formed a topic segmentation cost function which is later optimized by dynamic programming. However, the state-of-the-art methods in both OCR and ASR are known to have considerably less precision. Also OCR is sensitive to font and font size while ASR is sensitive to the dialect being spoken. This decreases the quality of the system significantly. As a result, we decided to go with a text and raw audio feature based approach.

System Architecture: We propose a novel architecture (Fig 1) that harvests text and acoustic features of these videos to generate hierarchical content tables. We extract the readily available subtitle and audio files from target videos. We use a rule based approach for

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331400

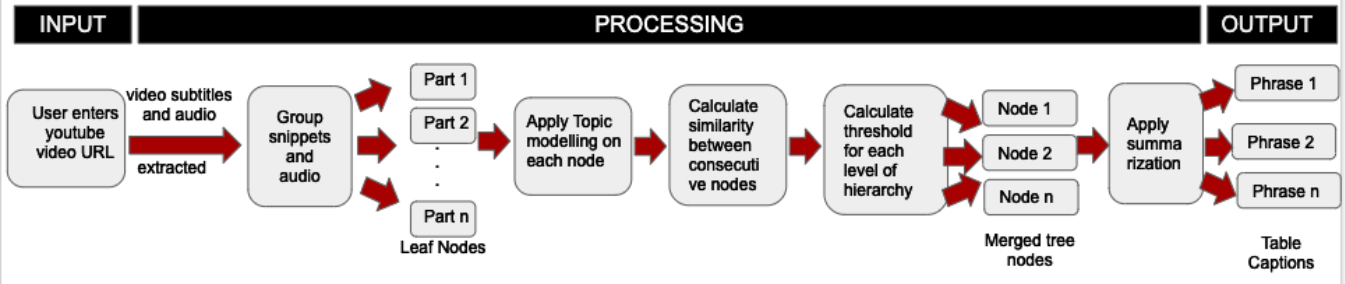

Figure 1: Architecture to auto generate content table for educational videos.

topic segmentation as our baseline. We experiment with various topic segregation techniques like LDA and Text Tiling and introduce a dynamic threshold based method to obtain optimal topic segments in videos. We beat our baseline significantly by 39.45% of macro F1-score and 35.76% of micro F1-score. After topic segmentation, we use text summarization techniques to generate headlines for video segments. We present a comparative analysis of summarization modules and report what works the best in our case.

2 PRELIMINARIES

Topic Modeling: We primarily experiment with two topic modelling approaches – Text Tiling and Latent Dirichlet Allocation (LDA). In **Text Tiling** [6], text is broken into tokens, and inflected nouns and verbs are reduced to their roots. The text is then divided into pseudo-sentences, and a dictionary of the token roots is built to store the indices of the token sequences they occur in along with their respective counts. A score is assigned to every token sequence gap based on the number of new words introduced in the interval considering the sequence gap as the mid point. The score stands for how strongly the cues of sub-topic change in sequences on either side of the sequence gap. Whenever the score exceeds a threshold, we create a new tile. **Latent Dirichlet Allocation (LDA)** [3] is a generative statistical model. It assumes that documents are produced from a mixture of topics which generate words based on their probability distributions [1]. For a given document, LDA backtracks to find the topics which could give origin to the document. It creates a document-term matrix, mapping each document to its vocabulary, with each cell giving the frequency of the corresponding word in the document. LDA then uses matrix factorization to convert it into two lower dimensional (document-topic and topic-term) matrices. It then iterates through each word for each document, trying to alter the present topic-word allocation. LDA iterates until a steady-state is reached with a consistent topic-term distribution.

Acoustics Modeling: Mel Frequency Cepstral Coefficients (MFCCs) are features that can be used to extract acoustic information from video data. They are computed by breaking the audio signal into small duration frames and computing the periodogram estimate for each frame. Next, the mel filterbank is applied to the power spectra and a log followed by Discrete Cosine Transform (DCT) is taken from the filterbank energies. DCT coefficients of 2-13 are kept while the rest are discarded.

Headline Phrase Selection: A number of text summarization methods are used to caption text articles. We experiment with two popular methods – LexRank [4] and TextRank [9]. In **LexRank**, a fully connected graph is built with selected phrases as vertices and a similarity function between these phrases is used to decide edge-weights. Edges with weights below a certain threshold are dropped. Vertices with most remaining edges contribute to the summary. **TextRank** extends PageRank and chooses phrases which collectively best represent the original documents.

3 PROPOSED METHODOLOGY

We now discuss our proposed architecture to generate hierarchical content tables for educational videos (see Fig 1).

3.1 Data Acquisition

We use youtube-dl [5] – a command line python library to download videos, audios and related metadata. We limit our test sets to videos with pre-uploaded subtitles. This is done to steer away from youtube auto-captioning speech recognition errors. We find that most popular Massive open online courses (MOOCs) and reputed university lectures have separately uploaded subtitle files.

3.2 Preprocessing

A subtitle file is built from small text snippets, each 3-4 seconds long. Each snippet is accompanied by a snippet index and its time stamp in the video. We concatenate the text corresponding to every 50 snippets into a string. We name these combined entities 'leaf nodes'. We use text processing techniques such as replacing common abbreviations with their full forms on these strings. We experiment with combining different number of snippets to form leaf nodes and find 50 to be an optimal number. Snippets much lesser than 50 significantly add to the computational latency in the architecture ahead while numbers significantly larger contribute to unwanted merging of separate topics.

3.3 Segregating Topics

In this step, we begin with an array of leaf nodes. Our goal is to take a decision on whether a particular leaf node depicts a topic change or not. We separately use rule based, text and acoustic features to model topics and later do a weighted merging.

- **Rule based**: We find intervals in the subtitle file with no text for more than 10 seconds. We assign an intersection score of zero in case such a gap exists and one otherwise.
- **Text features**: We initialize LDA on every leaf node with 10 features. As a result, LDA returns a list of 10 constituent topics for each leaf node. We compute Jaccard Similarity between every consecutive leaf nodes and report the scores. Separately, we initialize a text tiling model on our leaf nodes

and obtain a depth score for each leaf node intersection. Unlike Jaccard, since this score is a dissimilarity score, we subtract this score from 1 to get a similarity score.

- **Audio features**: We use librosa [8] - a python audio analytics package - to compute MFCC features for the extracted mp3 files. We feed our audio time series information to the MFCC spectral feature function of librosa. The function returns a MFCC sequence. We find probabilistic features like the mean, max, median, variance, skewness and kurtosis of the returned sequence and concatenate them to form a time series feature vector for each leaf node. We then calculate the cosine similarity between audio feature vectors of different leaf nodes and report them.

To combine these leaf nodes topically, we consider the intersection scores obtained from the above three methods: rule based, text and acoustics. We experiment by assigning different weights to each feature and compute a weighted sum. We define a threshold score to demarcate topic similarity. If our weighted sum happens to be more than this threshold, we merge our leaf nodes to form a single tree node. After this step, we obtain a list of tree nodes, each representing a different topic in the video.

3.4 Topic Node Merging

In the previous section, we define a threshold and merge leaf nodes that have similarity score more than this threshold. However, we find that different videos require different thresholds to reach optimal number of topics. Hence we propose a **dynamic thresholding algorithm** to decide whether to merge two leaf nodes or not.

From the previous section, we receive an array of similarity scores between every two leaf nodes. We find the largest and smallest values among these scores and find the difference. Thus our threshold must lie between these two scores. We experiment with different threshold values in this range and find that for a content table with n-level hierarchy, thresholds can be modeled well using an exponential function described below.

Let us denote the similarity score between leafNode$_i$ and leafNode$_{i+1}$ as sim_i. Let $\max_{sim}$ = max(sim) and $\min_{sim}$ = min(sim), and sim_diff = $\max_{sim}$ - $\min_{sim}$. Then we can denote the n^{th} hierarchical threshold as:

$$\text{threshold}_n = \max_{sim} - \text{sim\_diff} \times e^{-n}$$

It is a monotonically increasing function in terms of n, where the n^{th} level will have the highest threshold value.

$$\text{For } \lim_{n \longrightarrow \infty}, \text{threshold}_n \longrightarrow \max_{sim}$$

To maintain neatness of our UI, we limit our content tables to at most two levels of hierarchy.

3.5 Headline Generation

To build textbook like content tables, we need coherent phrases that represent the topics in a particular tree node. For this, we use text summarization methods. We feed the concatenated text of each tree node to a summarization module. We use the sumy python library for this purpose. We set the summary length to top three sentences. We then look for most common trigrams in these sentences to assign a syntactically valid phrase as our topic headline. We experiment with different summarization models such as KL Divergence, SumBasic, LexRank [4] and TextRank [9] . We present a comparative analysis of these methods in Section 4.

4 EVALUATION

4.1 Dataset and Annotation

We select a set of 100 educational videos from Youtube and NPTEL. On an average the videos are 50 minutes long (length of college lectures). Most videos have professors teaching computer science concepts. There is little to no external noise in these videos.

For evaluating our architecture, we manually annotate these videos. Our annotators segment the videos into major sections and records the start and end timestamp for each section. They then classify each section into subsections, if possible. Two annotators segmented the videos and we use average of both scores while computing accuracy of our topic segregation algorithms. For every section and subsection in the videos, our annotators suggest three primary keywords and a caption that best describes that segment. Three annotators, who have domain knowledge of the video manually tagged these segments, and the inter annotator agreement score (Kappa score) was 0.68.

4.2 Topic Modeling Evaluation

For calculating leaf node similarity score, we give weight to rule based:acoustic:text features in the ratio 2:3:5. We use this ratio as it gives the highest precision on our validation set. We then use above described leaf node merging algorithm to get a list of tree nodes. To calculate precision and recall, for each tree node we find the closest section in the manually annotated list and establish a one-to-one mapping between generated sections and the ground-truth sections. We consider the rule-based approach for topic segregation as our baseline. We use micro and macro statistics of precision and recall (see Table 1) to finally compute F1-score (see Table 2) and evaluate the performance of different algorithms for topic modeling. We achieve 0.410 macro F1-Score and 0.391 micro F1-score, beating the rule-based baseline by 39.45% and 35.76% respectively.

Table 1: Performance of LDA and Text Tiling for for topic segregation w.r.t Micro Precison (Mi-P), Macro Precision (Ma-P), Micro Recall (Mi-R) and Macro Recall (Ma-R).

Model	Mi-P	Ma-P	Mi-R	Ma-R
Text Tiling	0.39	0.41	0.29	0.31
LDA	0.47	0.46	0.34	0.36

Table 2: Macro (Ma) and Micro (Mi) F1-Score of topic segregation module after the addition of each feature.

Sl. No.	Features used	Ma-F	Mi-F
(1)	Rule Based	0.294	0.288
+ (2)	Textual Features (LDA)	0.391	0.373
+ (3)	Acoustic Features	0.410	0.391

4.3 Headline Generation Evaluation

We use ROUGE-1 and BLEU to evaluate caption generation algorithm. We compare the generated and manually annotated headlines from the one-to-one mapping of video sections established in

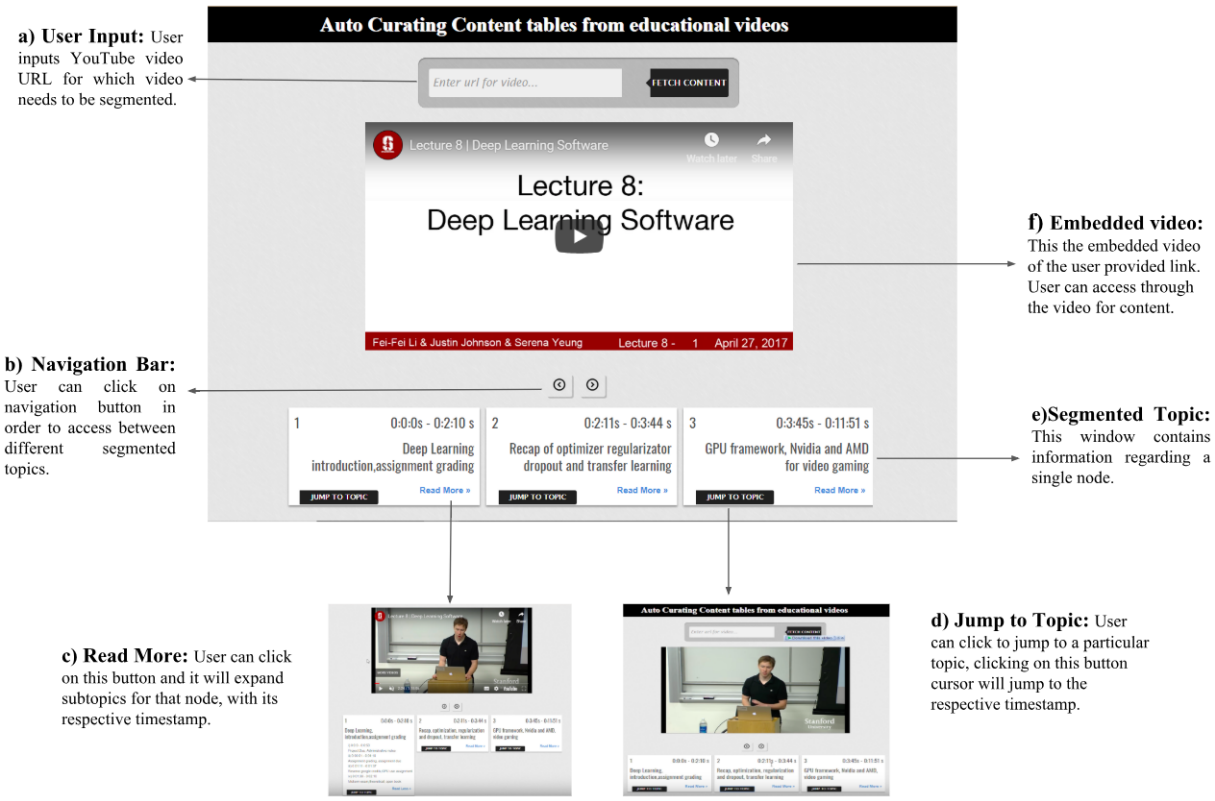

Figure 2: A screen out of our video content table generation user interface.

Table 3: Comparative evaluation of headline generation algorithms.

Algorithm	ROUGE-1	BLEU
KL Divergence (Greedy)	0.16	0.19
SumBasic	0.21	0.23
LexRank	0.34	0.39
TextRank	0.31	0.41

the previous section. Although ROUGE is the traditional metric to evaluate summarization, we choose ROUGE-1 and BLEU due to the short length of our summaries and our annotations. While LexRank achieves a BLEU score of 0.39, TextRank achieves 0.41 (see Table 3).

5 DEMONSTRATION

Figure 2 represents our system architecture. **Input:** Users can paste urls of any publicly available videos from video hosting forums. **Output:** We show a content table using the generated captions for the topmost level of the hierarchy.

6 CONCLUSION

We propose a novel architecture to automatically curate content tables for educational videos. Our system gives a textbook content table like interface for online educational videos. This can be used by educational video stores like Coursera or NPTEL to enhance user engagement on their platforms. We would like to extend this work to support intra-video search. Our ultimate goal is to make a comprehensive video handling interface with acceptable latency.

ACKNOWLEDGEMENT

The work was partially supported by the Ramanujan Fellowship (SERB) and the Infosys Centre for AI, IIIT-Delhi, India.

REFERENCES

[1] Shivam Bansal. 2016. Beginners Guide to Topic Modeling in Python. (2016). https://www.analyticsvidhya.com/blog/2016/08/beginners-guide-to-topic-modeling-in-python/
[2] Arijit Biswas, Ankit Gandhi, and Om Deshmukh. 2015. Mmtoc: A multimodal method for table of content creation in educational videos. In *ACM MM*. 621–630.
[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *JMLR* 3, Jan (2003), 993–1022.
[4] Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research* 22 (2004), 457–479.
[5] Ricardo Garcia Gonzalez. 2006. YouTube-dl: download videos from youtube. com. (2006).
[6] Marti A Hearst. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics* 23, 1 (1997), 33–64.
[7] Debabrata Mahapatra, Ragunathan Mariappan, Vaibhav Rajan, Kuldeep Yadav, Sudeshna Roy, et al. 2018. VideoKen: Automatic Video Summarization and Course Curation to Support Learning. In *WWW*. 239–242.
[8] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. 2015. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*. 18–25.
[9] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *EMNLP*. 1–8.