2021 IEEE International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS) | 978-1-6654-0610-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/CSITSS54238.2021.9683609

Video Based Transcript Summarizer for Online Courses using Natural Language Processing

Krishna Kulkarni
Department of Computer Science and Engineering
RV College of Engineering
Bangalore, India
krishnak.scs20@rvce.edu.in

Rushikesh Padaki
Department of Computer Science and Engineering
RV College of Engineering
Bangalore, India
rushikeshap.scs20@rvce.edu.in

***Abstract*—Online education has become an effective way to deliver quality education to students. They have become more popular because of their high graphical and pictorial content, delivered by experts in the subjects and convenient for learning at anytime and anywhere. But sometimes, students may not be able to go through the course content due to shortage of time. Video transcript summarizer has got a lot scope in this situation. It highlights the important topics from the video. The idea of summarizing the videos can be extended to online courses videos. This will help students save a lot of time as they can understand the gist of the class within less time without actually watching the video and by just going through the summary. Our system focuses on the development of a module using Natural Language Processing with python to summarize an online class video. The methodology adopted in this project uses Natural Language Processing (NLP) algorithms such as Term Frequency- Inverse Document Frequency (TF-IDF) and Gensim to obtain the summary of video of online course. The model takes URL of a video from user as input. We have implemented summarization process with the help of two algorithms. TF-IDF is an information retrieval algorithm which uses frequency of a term and its inverse document frequency. Gensim is a NLP package that deals with topic modeling. The model also gives the flexibility to the user to decide on as to what percentage of summary is needed compared to the original lecture. The summarization technique is a subjective process. We have incorporated two prominent methods. One is cosine similarity and the other one is ROUGE score. The former does not require human generated summary for reference, whereas latter requires it. The efficiency obtained using Cosine similarity is greater than 90% in both the cases: TF-IDF and Gensim. The efficiency obtained in case of ROUGE score is in between 40-50%.**

***Index Terms*—Natural Language Processing, Summarizer, TF-IDF, Gensim, Cosine Similarity, ROUGE**

I. Introduction

The Covid-19 pandemic has become the biggest threat to all sections of the society. Education system is no exception in this case. Education has been the most affected sections due to this pandemic. The only savior for reaching out students in this situation is online classes. Most of the schools and colleges in India have adapted to this method since the pandemic. They use different modes of online classes such as live classes and/or pre-recorded classes. Many platforms such as Google Meet, Zoom, Microsoft teams, Cisco WebEx and YouTube videos have come to their rescue. National Programme on Technology Enhanced Learning (NPTEL) is one such programme by Ministry of Education, Government of India. It existed even before the pandemic. Many students enroll into these courses to gain their knowledge or to fulfill their academic criteria. Sometimes, students find it difficult to manage both academic work and these online courses, may be due to shortage of time. In such a situation they need a system that reduces their burden by explaining them the gist of the video lecture without actually watching the video or reading a machine-generated transcript. Video transcript summarizer has got a lot scope in this situation. It highlights the important topics from the video. The idea of summarizing the videos can be extended to online courses videos. This will help students save a lot of time as they can understand the gist of the class within less time without actually watching the video and by just going through the summary. Our system focuses on the development of a module using Natural Language Processing with python to summarize an online class video.

II. Related Work

The core of our project is summarization. We have a video of an online class. We need to make a text summary out of the video. We have done an extensive literature survey on text summarization, its types and various techniques to incorporate the text summarization process.

Text summarization is the process of creating a short, concise, accurate and fluent summary of a long text document [9]. Text summarization can be extractive or abstractive. Extractive text summarization is a method in which the phrases and sentences are selected from the original document to make a summary [5]. Abstractive text summarization is a method that involves the generation of new phrases and sentences to make a summary [6]. In our project we use, extractive text summarization technique.

Hritvik Gupta et al [1] proposed a model that used three prominent algorithms. They used Latent Semantic Analysis (LSA) for topic modeling, TF-IDF for keyword extraction and Bidirectional Encoder Representations from Transformers (BERT) for encoding the sentences. LSA uses truncated Singular Value Decomposition (SVD) to extract all the relevant topics from a text. TF-IDF extracts key words from each sentence of a document. BERT model is used to encode the

978-1-6654-0610-9/21/$31.00 ©2021 IEEE

sentences for retrieving positional embedding of topics. This paper provides a clear understanding of the TF-IDF algorithm. Swaranjali Jugran et al [3] proposed an extractive text summarization-based model. They demonstrated the text summarization with the help of NLTK and Spacy Tools. NLTK is a natural/human language toolkit in Python. Spacy is an open-source library for NLP. Their results show that Spacy is a better option for text summarization when compared to NLTK. So, in our project we use Spacy.

Surabhi Adhikari et al [4] gave a complete idea of various text summarization techniques in their work. They gave the idea of different summarization techniques in NLP. Few of them to be mentioned here are- K-means clustering, K- nearest neighbors, TF-IDF, LSA, Sentence Ranking and BERT model. Supreetha D et al [10] provides an extensive survey of the abstractive text summarization techniques available. They conducted research on various methodologies and represented them in terms of their advantages and disadvantages. Vari- ous methods include- Template based, Ontology based, Rule based, Multimodal semantic model and so on.

Divya Santwani et al [11] designed a text rank architecture for extractive text summarization of a document. They used Concept net numbers to represent the words in the document. They worked on Akbar and Birbal Stories for summarization. The model performed only on a small subset of data.

J. N. Madhuri et al [14] explained a model of extractive text summarization of a text document into an audio. The model used sentence ranking and NLTK in the summarization process. The major drawback of the system was that it did not work on a large document and also the process of conversion of summary to audio was missing.

Luca Cagliero et al [15] proposed a model called as ELSA. It is a multilingual document summarization algorithm. It is based on Frequent Itemsets and LSA. From this work, we understood the concept of evaluation metrics and also the concept of frequent itemsets.

Rahim Khan et al [16] implemented an automatic text summarization model using K-means algorithm and TF-IDF algorithm. The model worked on a well formed document. The selection of value of K in K-means algorithm is arbitrary in this case.

Kaiz Merchant et al [19] proposed a NLP based Latent Semantic Analysis for Automatic Text Summarization. They worked on Legal judgments from courts and tried to summarize the long judgment into short and useful summaries. The major drawback of this system was that, the model evaluation was done on similar words and not on entire concept.

III. TF-IDF Algorithm

TF-IDF algorithm is the combination of two algorithms, which are multiplied together. One is Term Frequency (TF) and another one is Inverse Document Frequency (IDF). TF is defined as how often a word appears in a document, divided by how many words there are. IDF defines how unique or rare a word is [8].

The steps used to implement this algorithm are-

1. Tokenization of the sentences: We tokenize the sentences here instead of the words. Next we give weight to these sentences.
2. Creation of the Frequency matrix of the words in each sentence: We calculate the frequency of the words in each sentence. The result is a key value pair where, key is sentence and value is a dictionary of word frequency.
3. Calculation of TF and Generation a matrix: We find the TF for each word in a paragraph.
4. Creation of table for documents per words: Here, we calculate how many sentences contain a word. We call this as Documents per words matrix.
5. Calculate IDF and generate a matrix: Here, we find IDF for each word in a paragraph.
6. Calculate TF-IDF and generate a matrix: Now that we have both the matrices and the next step is very easy. We multiply the values from both the matrices TF and IDF and generate a new matrix.
7. Score the sentences: Scoring of a sentence done using TF-IDF score of words in a sentence to give weight to the paragraph.
8. Find the threshold: We calculate the average the sentence score in this step.
9. Generate the summary: Select a sentence for a summarization whose sentence score is greater than the average score.

IV. TextRank Algorithm

We implement the textrank algorithm using the Gensim library in Python. It is an open-source vector space modeling and topic modeling toolkit. It is implemented using packages like NumPy, SciPy and Cython. We use the summarization.summarizer from gensim library. The summarization is based on ranks of text sentences using a variation of the TextRank algorithm which is a general purpose, graph based ranking algorithm for NLP. It is an automatic summarization technique. Graph-based ranking algorithms are a method for deciding the importance of a vertex within a graph, based on global information recursively drawn from the entire graph.

The steps used to implement this algorithm are-

1. Pre-processing of the corpus: This includes stemming, removing stop words and punctuation marks.
2. Make a graph where the vertices are the sentences of the corpus.
3. In the graph the edge between two vertices denotes the similarity between them.
4. Run PageRank algorithm on the graph generated after including the edge weights i.e. the weighted graphs.
5. Pick the vertices with highest score and add the corresponding sentences of those vertices to the summary.
6. Based on the fraction value, the number of vertices to be chosen is decided.

V. Methodology

The proposed system for text summarization of online course videos is implemented using two algorithms. They are TF-IDF and Gensim. The model takes URL of a video from

user as input. We have implemented summarization process with the help of two algorithms. TF-IDF is an information retrieval algorithm which uses frequency of a term and its inverse document frequency. Gensim is an NLP package that deals with topic modeling. The model also gives the flexibility to the user to decide on as to what percentage of summary is needed compared to the original lecture.

The URL of the video lecture from YouTube is fed as input to the system. Along with the URL, the user has to choose the model to be used for summarization i.e., either TF- IDF or Gensim, the frequency ranging from 0.1 to 1 as what percentage of summary is desired from original video lecture and local drive location i.e., where the output summary needs to be stored. The video summarizer begins the process. Once the summarization process is done, the output summary file is automatically stored in the location mentioned by the user. The GUI provides Open Folder button for user to open the local folder and see the output.

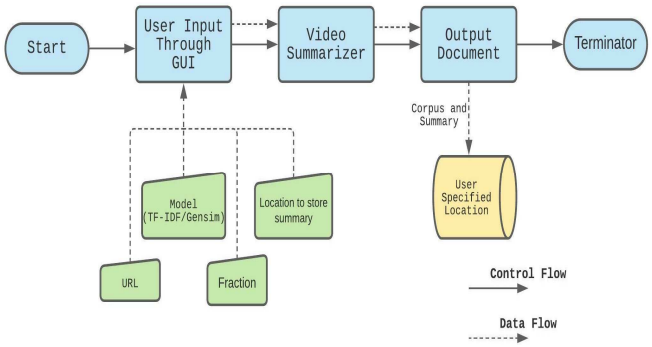

Fig. 1. System Architecture of the Proposed System

VI. Implementation

We have implemented the proposed system using Python Programming language. This section describes the modules used to implement the proposed system.

A. The on-submit Module

The on-submit module is where the interface of the project lies. The interface takes in inputs from the user. The input consists of the 'URL', the Summarization 'Module' selected, the required size of the summary in terms of a 'Fraction' of the corpus and the 'Location' i.e., the path to save the corpus and summary. On clicking 'Submit', this module calls 3 other modules sequentially, which we shall see next. Once the last module, i.e., the save-summary module executes successfully, the on-submit module enables the "Open Folder" button in the UI which enables the user to open the Location specified directly.

B. The get-caption Module

The first module that the on-submit module calls is get-caption. The URL is passed to this module. This module in turn verifies the correctness of the URL. If the URL is incorrect, an error message is thrown to the user. If the URL is valid, the module proceeds to extract the vtt file, short for "Video Text Track", which is a popular subtitle and caption format. The module also converts the vtt file to a readable text file which we will refer to as "corpus", as this is the file that we will use for a summary.

C. The summarizer Module

The second module that on-submit module calls is the summarizer module. The inputs passed to this module are the summarization module chosen by the user, the corpus generated in the get-caption module and the fraction which will determine the size of the summary in terms of a fraction of the module. Based on the choice of summarization module i.e., "TFIDF-based Module" or "Gensim-based Module", the summarizer calls either of the two modules i.e., tfidf-based or genism-based. The corpus and the fraction are passed to the module selected. Once the summarizer module receives the summary from either of these modules, it passes the summary back to the on-submit module.

D. The tfidf-based Module

The input passed to this module is the corpus and the fraction which determines size of the summary. This module is where the summary generation happens based on the tf-idf algorithm. The module first vectorizes the corpus using the pandas library. The vectors are assigned TF values and IDF values. TF*IDF is calculated, and each sentence is assigned a weight. This weight is used to choose which sentence is to be selected to be included in the summary. The fraction here determines the numbers of sentences we need to select and based on priority sentences are chosen. The summary is returned to the summarizer module.

E. The genism-based Module

The input passed to this module is the corpus and the fraction which determines size of the summary. This module is where the summary generation happens based on the genism- based PageRank algorithm. Gensim is a Natural Language Processing package that helps in Topic Modeling.

Gensim implements the PageRank summarization using the summarize() function in the summarization module. The module first pre-processes the corpus by removing stop words, punctuations and stemming. The module converts the corpus into a graph by considering sentences as vertices of the graph. The edge between the two vertices denotes the similarity between them. The PageRank algorithm is run on this weighted graph. The vertices i.e., sentences with the highest score are used in the summary. Based on the fraction value, the number of vertices to be chosen is decided. Once the summary is generated, it is passed back to the summarizer module.

F. The save-summary Module

The third module that on-submit module calls is the save-summary module. The generated summary and the Location

are forwarded to the save-summary module. The module uses os libraries to check permissions of the given folder path and saves the summary at the location in a ".txt" format. The name of the summary is given like "[Title of the Video] [Summarization Module Used]" for example, "Introduction to Data Science Tfidf-Based" or "What is Civil Engineer- ing Gensim-Based". Once the summary is saved, the save-summary module indicates the same to the on-submit module.

VII. Experimental Results

This section provides a description of the evaluation metrics used to analyze performance of the built system and the results obtained based on which the conclusions can be drawn.

We have used Cosine similarity and ROUGE metrics to evaluate our system.

A. Cosine Similarity

Cosine similarity is a metric used to determine how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. To compute the co- sine similarity, we use the en-core-web-sm language model provided by spaCy. We compute cosine similarity between the corpus and the generated summary. The value of cosine similarity can range between 0 and 1 which scales to 0 to 100 percent.

The cosine similarity for both models is almost the same. The graph with both plots is as below:

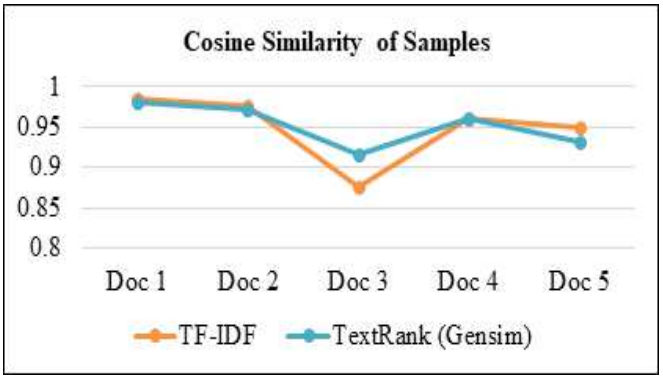

Fig. 2. Cosine Similarity of Samples

As we can see from the plot, the cosine similarity of both values is nearly equal, which on average is 95%. So we can say that according to cosine similarity the efficiency of the model is 95%.

B. ROUGE

ROUGE (Recall Oriented Understudy for Gisting Evaluation) is a set of metrics used for the evaluation of a text summarizations. The metrics compare automatically generated summary with reference summary or multiple reference summaries [2].

ROUGE-N: It measures the overlap of n-grams between the generated summary and reference summary.

ROUGE-L: Compute the Longest common subsequence between reference summary and the generated summary. Each sentence in a summary is considered as a sequence of words. Two summaries which have longer common sequence of words are more similar to each other.

We use Rouge-1, Rouge-2 and Rouge-L to compare the generated summary to a reference summary obtained from a commercial text summarizer. Rouge metrics involve precision, recall and f1-score. All 3 metrics range from 0 to 1 and can be scaled to 0% to 100%.

1) Recall: The recall is the ratio between the number of overlapping n-grams found in both the model output and reference, to the total number of n-grams in the reference. While this is good for capturing all of the information contained in the reference — it does not ensure that the model is not just pushing out a huge number of words to increase the recall score.

$$Recall = \frac{\text{No. of n-grams found in model and reference}}{\text{No. of n-grams in reference}} \quad (1)$$

2) Precision: To avoid this issue faced in recall metric mentioned above, we use the precision metric - which is calculated in a similar way, but rather than dividing by the reference n-gram count, it is divided by the model n-gram count.

$$Precision = \frac{\text{No. of n-grams found in model and reference}}{\text{No. of n-grams in model}} \quad (2)$$

3) F1-Score: Once we have the precision and recall values, they can be used to calculate our ROUGE F1 score.

$$F1\text{-}Score = (2 * \text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall}) \quad (3)$$

The average rouge metrics for the given TF-IDF model when compared to a commercial text summarizer is given in the Fig. 3.

According to rouge-1 metrics, where the evaluation parameter is 1-gram, the precision is 0.47, recall is 0.36 and F1-Score is 0.4. Similarly values for Rouge-2 and Rouge-L can be seen.

The average rouge metrics for the given Gensim model when compared to a commercial text summarizer is given in Fig. 4.

According to rouge-1 metrics, where the evaluation parameter is 1-gram, the precision is 0.44, recall is 0.35 and F1- Score is 0.39. Similarly values for Rouge-2 and Rouge-L can be seen.

VIII. Conclusion

The proposed project aimed to create a summarizer for online lectures found on YouTube by extracting the closed captions of the videos. The project was performed as a way to

TF-IDF Summarizer			
Metric	**Recall**	**Precision**	**F1-Score**
Rouge-1	0.3646	0.4666	0.4094
Rouge-2	0.1666	0.2449	0.1983
Rouge-L	0.3333	0.4266	0.3743

Fig. 3. Rouge metrics of TF-IDF Model

Gensim Summarizer			
Metric	**Recall**	**Precision**	**F1-Score**
Rouge-1	0.4444	0.3478	0.3902
Rouge-2	0.1363	0.1176	0.1263
Rouge-L	0.3888	0.3043	0.3414

Fig. 4. Rouge metrics of Gensim Model

make it easier to find a brief summary of long lecture videos that capture the gist of the video. The project also aimed to compare the performance of two summarization algorithms, TF-IDF and Textrank. Both models performed as expected with an efficiency of 95% on the basis of cosine similarity.

When both models are compared to a commercial text summarizer, we see that, on an average, the rouge metrics for TF-IDF are 0.3 precision, 0.4 recall and F1-score of 0.3. The average rouge metrics for Gensim are 0.3 precision, 0.25 recall and F1-score of 0.28. The TF-IDF model performs slightly better here. This could be due to the fact that the commercial summarizer may be using an internal TF-IDF model for summarization. The Rouge score in this case is subjective and the cosine similarity evaluation takes higher precedence.

REFERENCES

[1]H. Gupta and M. Patel, "Method Of Text Summarization Using Lsa And Sentence Based Topic Modelling With Bert," 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), 2021, pp. 511-517, doi: 10.1109/ICAIS50930.2021.9395976.

[2]I. Awasthi, K. Gupta, P. S. Bhogal, S. S. Anand and P. K. Soni, "Natural Language Processing (NLP) based Text Summarization - A Survey," 2021 6th International Conference on Inven- tive Computation Technologies (ICICT), 2021, pp. 1310-1317, doi: 10.1109/ICICT50816.2021.9358703.

[3]S. JUGRAN, A. KUMAR, B. S. TYAGI and V. ANAND, "Extractive Automatic Text Summarization using SpaCy in Python NLP," 2021 International Conference on Advance Computing and Innova- tive Technologies in Engineering (ICACITE), 2021, pp. 582-585, doi: 10.1109/ICACITE51222.2021.9404712.

[4] Rahul, S. Adhikari and Monika, "NLP based Machine Learning Approaches for Text Summarization," 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), 2020, pp. 535-538, doi: 10.1109/ICCMC48092.2020.ICCMC-00099.

[5] Verma, Pradeepika Verma, Anshul, "Accountability of NLP Tools in Text Summarization for Indian Languages" Journal of scientific research, 2020, 64, 258-263. 10.37398/JSR.2020.640149.

[6]P. Batra, S. Chaudhary, K. Bhatt, S. Varshney and S. Verma, "A Review: Abstractive Text Summarization Techniques using NLP," 2020 International Conference on Advances in Computing, Communication Materials (ICACCM), 2020, pp. 23-28, doi: 10.1109/ICACCM50413.2020.9213079.

[7]W. H. Ong, K. G. Tay, C. C. Chew and A. Huong, "A Com- parative Study of Extractive Summary Algorithms Using Nat- ural Language Processing," 2020 IEEE Student Conference on Research and Development (SCOReD), 2020, pp. 406-410, doi: 10.1109/SCOReD50371.2020.9251032.

[8]R. Boorugu and G. Ramesh, "A Survey on NLP based Text Summariza- tion for Summarizing Product Reviews," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020, pp. 352-356, doi: 10.1109/ICIRCA48905.2020.9183355.

[9] Supreetha D, Rajeshwari S B and Jagadish S Kallimani, "Abstractive Text Summarization Techniques", International Journal of Engineering, Science and Computing, Vol. 10, Issue No. 7, 2020

[10] Divya Santwani, Akash Bedi and Mohit Bahrani, "Textizer", International Journal of Engineering Research and Technology, ISSN: 2278-0181, Vol. 9 Issue 07, July-2020

[11]J. N. Madhuri and R. Ganesh Kumar, "Extractive Text Summarization Using Sentence Ranking," 2019 International Conference on Data Science and Communication (IconDSC), 2019, pp. 1-3, doi: 10.1109/IconDSC.2019.8817040.

[12] Luca Cagliero, Paolo Garza, and Elena Baralis. 2019. ELSA: A Multilingual Document Summarization Algorithm Based on Frequent Itemsets and Latent Semantic Analysis. ACM Trans. Inf. Syst. 37, 2, Article 21, March 2019. doi: https://doi.org/10.1145/3298987.

[13] Rahim Khan, Yurong Qian, Sajid Naeem, "Extractive based Text Summarization Using KMeans and TF-IDF", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.11, No.3, pp. 33-44, 2019. doi: 10.5815/ijieeb.2019.03.05.

[14]K. Merchant and Y. Pande, "NLP Based Latent Semantic Analysis for Legal Text Summarization," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2018, pp. 1803-1807, doi: 10.1109/ICACCI.2018.8554831.