International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-6, August, 2019

An Optimized E-Lecture Video Retrieval based on Machine Learning Classification

Lakshmi Haritha Medida, Kasarapu Ramani

Abstract: ***The advent of internet has lead to colossal development of e-learning frameworks. The efficiency of such systems however relies on the effectiveness and fast content based retrieval approaches. This paper presents a methodology for efficient search and retrieval of lecture videos based on Machine Learning (ML) text classification algorithm. The text transcript is generated exclusively from the audio content extracted from the video lectures. This content is utilized for the summary and keyword extraction which is used for training the ML text classification model. An optimized search is achieved based on the trained ML model. The performance of the system is compared by training the system using Naive Bayes, Support Vector Machine and Logistic Regression algorithms. Performance evaluation was done by precision, recall, F-score and accuracy of the search for each of the classifiers. It is observed that the system trained on Naive Bayes classification algorithm achieved better performance both in terms of time and also with respect to relevancy of the search results.***

Keywords : Logistic Regression, Machine Learning, Naive Bayes, Support Vector Machine, Text Classification, Video retrieval

I. INTRODUCTION

The demand for e-learning is constantly rising due to its numerous benefits over the traditional classroom learning practices. Recorded video lectures are used more commonly by the students for e-learning purpose. As these video lectures contribute to a very efficient and flexible learning the demand for e-learning through video lectures is rapidly increasing. This demand has lead to the huge volumes of lecture video files in the web. So, it is cumbersome for a student to search a particular area of interest or a particular video in these numerous repositories. Hence, the need for a more proficient method of video retrieval in the web or within huge lecture video collections arose. The textual or content based metadata facilitates easy search and browsing through the video lecture corpus [1]. This metadata helps a user to identify and understand the lecture video contents more effectively, and the learning efficiency can accordingly be enriched. The appropriate metadata can be automatically collected from lecture videos by using suitable analysis methods.

Revised Manuscript Received on October 30, 2019.
* Correspondence Author
Lakshmi Haritha Medida*, Research Scholar, CSE, JNTUA, Ananthapuramu, India.
Kasarapu Ramani, Intelligent Computing Research Centre, Department of IT, Sree Vidyanikethan Engg College(Autonomous), Tirupati, India.

Text is a superior semantic feature which has often been used for content-based information retrieval. In lecture videos, text serves as an outline for the lecture and is very important for understanding the content. The textual information can thus be generated by using OCR (Optical Character Recognition) and ASR (Automatic Speech Recognition) methods, which unwraps the content of the video lectures. The current work is carried out on the textual information extracted based on ASR [2].

The textual metadata can be used for the goal of text categorization. It is the classification of documents into a fixed number of predefined categories. This allows users to obtain more relevant information faster by searching only the pertinent categories and not the entire video corpus. The significance of text classification is more evident when the lecture video space is huge such as the World Wide Web [3].

Machine learning offers capable tools for significant classification of documents [4]. A technique's performance depends not only on the algorithm in use, but also on the characteristics of the data in use. Hence for a specific data and a particular task, it is important to identify the right technique for the task. As such, evaluating how different techniques perform in classifying text is of much value. In this paper, we proposed a system that retrieves the lecture videos based on the text classification model trained on the summarized text transcript data of the speech content. Based on the trained classifier model, the retrieval system could quickly retrieve the most relevant videos by predicting the category of the search query. During the video retrieval, the search is built upon the predictions made by the classifier model rather than going through the entire database. Therefore, an optimized solution is provided for retrieving the e-lecture videos by improving the relevancy of the results in a reduced time.

The rest of the paper is organized as: Section II analyses the related work on the lecture video retrieval and Section III designates the text classifier model developed for the effective lecture video retrieval. Section IV, provides the implementation details of the refined search mechanism. Section V evaluates the proposed methodology. Finally, Section VI concludes the paper with an outlook on future work.

II. LITERATURE REVIEW

H Yang et al. presented a system that allows search for the temporal scope of a specific lecture topic within a lecture video [5]. This approach implemented the extraction of content-based metadata automatically from the video as well as the audio resource of lecture videos. For textual extraction purpose the OCR and ASR techniques were used. Stephan Repp et al.

proposed a system that allows browsing in sections of videos by creating a chain index for a lecture video [6]. The implemented index structure and the assessment of the keywords present within the lecture video and the user interface for dynamic browsing were presented for the e-learning content.

M A Parveenet al. offered a technique for content-based lecture video segmenting and retrieval in large lecture video collection [7].

Subsequently, extracted textual metadata by applying OCR on the key-frames and ASR on the lecture audio tracks for a more efficient retrieval of the videos and improved recognition rate.

Naphade et al. [8], proposed a framework for video indexing and retrieval using semantic unit "multijects" (i.e., "multiple objects"). Different feature sets such as color, texture, edges, shape, and motion were extracted from each frame and passed through different classifiers that check the multijects individually and combine the results to arrive at the final decision. Similarly, Smith et al in [9] proposed a problem of feature fusion when integrating features, models, and semantics for TREC video retrieval. They represented and retained the results from different feature sets in GMMs (Gaussian mixture models), instead of combining different feature sets to one.

Miha Grcar et al. presented a semi-automatic categorization of video-recorded lectures into taxonomy through ML task [10]. This categorizer combined information present in texts related with lectures and information extracted from various links between lectures in a unified ML framework for efficient browsing. Rajendra K R et al. used Term Frequency Inverse Document Frequency (TFIDF) to form clusters of similar documents and ranking mechanism is implemented to rank the documents of each cluster [11]. Their approach helped the user to acquire all the important documents in one place and can then limit the search functionality only to the selected top most documents.

The search methodology proposed restricts the search only to a particular category thus reducing the retrieval time and increasing the relevancy of the items. This is achieved by ML text classification model trained on the summary and keywords of the textual transcript generated from the audio content. The ranking mechanism implemented on TFIDF and cosine similarity further improved the relevance of documents fetched.

III. TEXT CLASSIFIER MODEL

This work, presented a methodology for lecture video retrieval based on ML text classification model trained on the summary and keywords extracted from the text transcript. The text transcript is generated from the audio component of the lecture videos using ASR. In this context, training the appropriate text classification model is a crucial step as the effectiveness of the video retrieval system depends wholly on the predictions made by the trained classification model. For training the model on the summary and keywords, we implemented the Naive Bayes, Support Vector Machine and Logistic Regression algorithms. The efficiency of classification achieved by each of these techniques is compared. The workflow that is followed is represented by Fig. 1.

A. Audio Component Extraction

Initially a database is created with the attributes such as name, size, storage location, category of lecture video file etc. Lecture videos of variable length and distinctive internal structure are collected from different internet sources. The video files gathered are of MP4 format. The audio file is extracted from the given lecture video file and is stored in database. The database created is therefore well organized with the inserted lecture video content such that it provides an effective retrieval of the content during the search operation.

The extraction of audio track from the video lecture recording is implemented using Fast Forward Moving Pictures Expert Group (FFmpeg) with Python subprocess. The audio track is extracted in WAV format. The extracted WAV file is subjected to resampling. It is resampled to a 16KHz 16-bit mono little-endian format as this format is proved to have given higher recognition efficiency with different ASR tools.

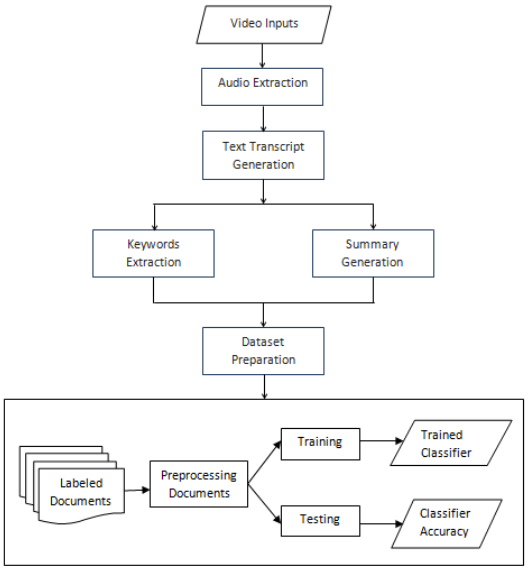

Fig. 1.Workflow for building the ML text classification model.

B. Audio to Text Translation

Segmentation is the process of splitting a heterogeneous input audio stream into composite audio segments or audio clips. The segmentation of audio files is exclusively a considerable step as it provides scope for much easier and also more accurate analysis. The resampled WAV file is subjected to segmentation. The duration of each WAV file is calculated by extracting the number of frames and the frame rate of the WAV file. Then the entire duration is split into number of segments with each segment of 20 seconds duration. These segments are further processed for extracting the audio content. Each segment obtained by the audio segmentation is subjected to speech to text translation. The Google Speech Recognition(GSR) library of Python is used for the text transcript generation. It relays on the neural network implementation and analyses tons of voice spectrogram patterns to predict the new patterns. This requires an active internet connection to work. The extracted text from all the segments can further subjected to summary and keywords extraction.

C. Automatic Summary and Keyword Extraction

Automatic summarization technique automatically summarizes the given text [12], by extracting one or more important sentences from the text. Automatic keyword extraction technique identifies the terms that represent the most relevant information contained in the text. Therefore automatic summary and keyword extraction techniques will diminish the complexity and length of the document, while retaining the essential qualities of the original text content. Instead of training the text classification model directly on the text transcript data, we train the model on the summary and keywords extracted. This step is critical as it reduces the time required for training the text classification model while not effecting the efficiency of the model.

Python's gensim library is used for generating the summary and keywords from the text transcript documents. The gensim implementation for summary and keyword extraction is based on the accepted "TextRank" algorithm. The output summary and keywords are added as attribute to the dataset that is being used for training the text classification model. A predefined category is added for each instance to perform multiclass classification. This dataset is used for training the model that classifies the video files based on the assigned category.

D. Training the Text Classifier Model

Classification is a supervised learning task whose objective is to deduce a prediction model by using a training dataset containing instances whose category is known, and then using the deduced model to assign class labels to test instances whose categories are unknown. Text classification is a well defined process of classifying text into categories. Using ML to automate the text classification tasks, makes the process efficient and fast [13]. Text classification in this context of video retrieval helps in achieving the most appropriate results in reduced time. The text classifier model is trained on the prepared dataset using the popular classifier models viz. Naive Bayes, Support Vector Machine and Logistic Regression in order to assess the classifier model that best fits the data. From result evaluation, Naive Bayes classifier provided the best model for the considered data.

Naive Bayes(NB): NB classifiers are a genre of classifiers that are based on the Bayes' probability theorem. NB classifier is a simple yet well performing model, particularly in the field of text classification. The NB classifier is based on the assumption that the features in a dataset are mutually independent. NB classifier still tends to perform very well under this impractical assumption [14]. It is relatively robust, fast, accurate and easy to implement. A part from this NB can be trained with limited resources. NB therefore can be considered as a simple text classification algorithm that is based on the probability laws and works reasonably well in complex real world text classification problems.

Support Vector Machine(SVM): SVM Classifiers attempt to separate the data by using linear or non-linear separations between the different classes. The crucial part of such classifiers is to decide the optimal boundaries between the different classes and use them for the determination of classification [15]. The features of text that make SVMs work for text classification are very few number of irrelevant features present in the text data. A part from this majority of the text classification problems is linearly separable.

Logistic Regression(LR): LR is a discriminative classifier model. Since LR gives a linear classifier, it best suits for the binary classification. LR can easily be generalized to multiple classes.

The overall implementation of the text classifier model is achieved through algorithm 1 and algorithm 2 gives the detailed implementation of the ML text classification.

Algorithm 1

Input

{ v } : Set of lecture video files of MP4 format

Output

Mcc_model.sav : Trained ML classification model

Procedure

1. *for each* v_in ϵ { v } *do*
    1.1 Extract audio from the video using ffmpeg
        a ← ffmpeg (v_in)
        a : audio of 16KHz 16- bit mono little endian WAV file
    1.2 Divide the audio into segments of 20 seconds duration
        a ← { a_i } where $1 \leq i \leq n$ where n: no. of segments
    1.3 Generate the text transcript for the audio using GSR
        for each i ϵ { a_i } *do*
            text ← transcript_gen(i)
            trans ← trans + text
            trans : text file of transcript generated from audio
        end for
    1.4 Extract keywords from the transcript using gensim library
        keywords ← keyword_ext(trans)
    1.5 Extract summary from the transcript using gensim library
        summary ← summary_ext(trans)
    1.6 Combine the keywords with summary
        ke_summ ← keywords + summary
end for
2. Prepare the dataset from ke_summ
    summ_mc.csv : dataset with the summary and class labels for categories
3. Train the ML classification model on the dataset
4. Save the model as mcc_model.sav

Algorithm 2

Input

summ_mc.csv : dataset

Output

mcc_model.sav : trained ML classification model

Procedure

1. Set X to summary and Y to category from the dataset

X ← summary

Y ← category

2. Perform oversampling to generate a balanced dataset

3. Preprocessing the text by removing stopwords and symbols

X ← clean_text(X)

4. Split the data into train and test sets

X_train, X_test, Y_train, Y_test ← train_test_split(X, Y)

5. Build the pipeline for the classifier

nb : classifier pipeline

6. Train the model

nb.fit(X_train, Y_train)

7. Obtain the accuracy of the trained model

accuracy_score(nb.predict(X_test), Y_test)

8. Save the model as mcc_model.sav

IV. SEARCH SYSTEM

Search procedure discovers the matched documents from the corpus. Once the matched documents are retrieved the documents are ranked for relevance and displayed in that order. The workflow of the search and ranking system is shown in Fig. 2. When a user query is given, the search category and the keyword are fetched from the form. The categories considered include 'All', 'Computers', 'Electronics', 'Aerospace', 'Civil', 'Mechanics', 'Irrigation', 'Datascience', 'Health and Safety'. If the specific category is not specified by the user, the trained ML text classification model is run for predicting the category of the search keyword. The text transcripts of the predicted category are then consulted to fetch the matched documents for the search expression. The appropriate documents are then ranked based on their degree of relevance, considering the TFIDF [16] and Cosine Similarity scores. The resultant videos of the search expression are displayed in the order of their relevance. The entire search process performed is given in algorithm 3.

Algorithm 3

Input

search_term : search query given

search_category : search category chosen

Output

Lecture videos that matches the search

Procedure

1. *If* search_category = 'All' *then*

Predict the category of ML classification model mcc_model.sav

category ← nb.predict([search_term])

end if

else

Set category as the search_category

category ← search_category

2. Perform search in the specific category

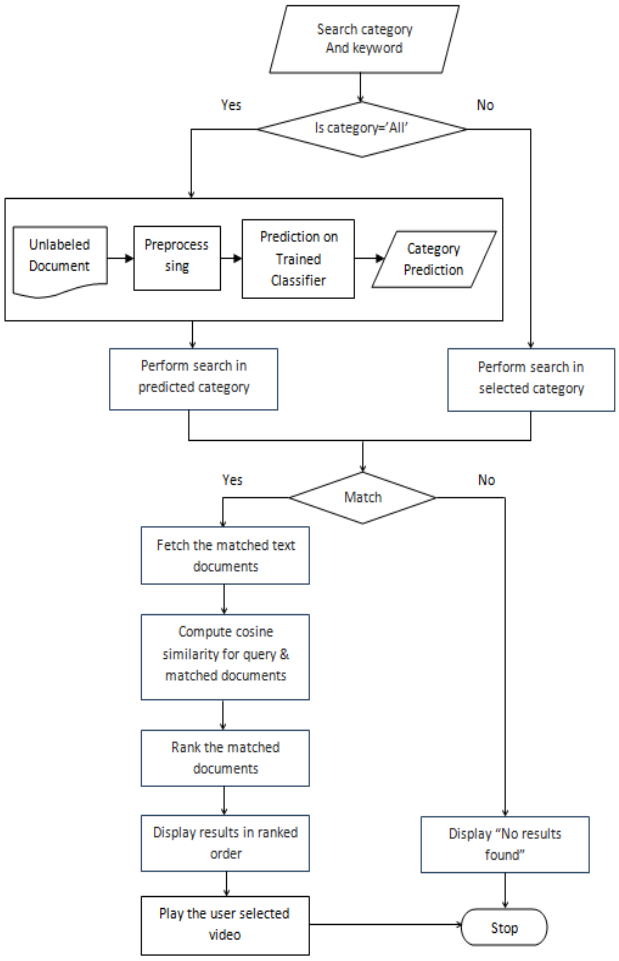

Fig. 2.Architecture of the implemented search system

3. *If* match found *then*

Fetch the matched documents

docs : list of matched documents

for each file ϵ docs *do*

calculate the cosine similarity score

tf_idf ← Tfidfvectorizer().fit_transform(file)

cosine ← cosine_similarity(tfidf[0:1], tf_idf)

cosine_values ← cosine_values + [cosine]

end for

Display results in sorted order of relevance

end if

else

Display "No results found"

V. RESULT EVALUATION

A. Evaluation of the Text Classifier Model

In order to finalize whether a text classification model is accurately capturing a pattern, we must evaluate that model. The outcome of this evaluation is important for deciding the reliability of the model. The elementary metric that can be used to evaluate a classifier is the accuracy of the model i.e. the percentage of test inputs that the classifier correctly labeled. Performance evaluation of NB, SVM and LR classifiers was undertaken by measuring the accuracy metric as tabulated in Table I. Fig. 3 gives the graphical representation for the same.

Table- I: Comparison of predictions obtained using ML Classification algorithms

Prediction / Search Term	Algorithm		
	Naive Bayes	*Support Vector Machine*	*Logistic Regression*
Safety management	True	False	False
Water resource management	True	True	False
Raster graphics	True	True	True
Lift	True	True	True
Normalization	True	True	True

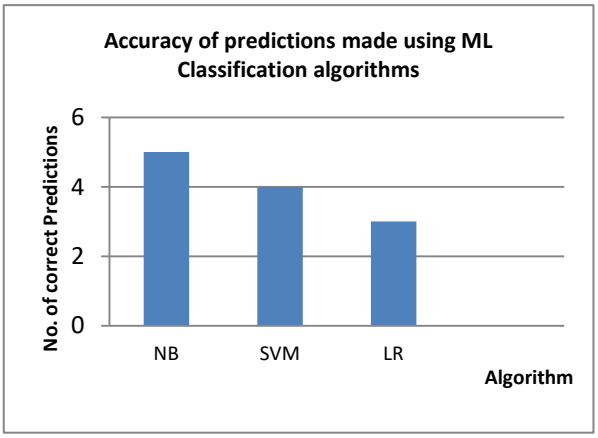

Fig. 3. Accuracy of predictions made by ML text classification models.

B. Evaluation of the Search Efficiency

The efficiency of the search technique implemented is evaluated based on the metrics of recall, precision, F-score and accuracy [17]. Recall is the extent to which retrieval of desired items occurred. Precision refers to the proportion of items retrieved that are appropriate. The F-score measure binds the recall and precision into a single metric. Accuracy is an obvious metric of judge the search efficiency which is given by the fraction of items that are correct. Each of these metrics is given by

$$Recall = \frac{No.\ of\ retrieved\ relevant\ items}{No.of\ relevant\ items} \quad (1)$$

$$Precision = \frac{No.\ of\ retrieved\ relevant\ items}{No.of\ retrieved\ items} \quad (2)$$

$$F-score = \frac{2*Precision*Recall}{(Precision+Recall)} \quad (3)$$

$$Accuracy = \frac{(No.\ of\ retrieved\ relevant\ items + No.of\ non\ retrieved\ irrelevant\ items)}{Total\ no.of\ items} \quad (4)$$

The values of recall, precision, F-score and accuracy achieved for different search terms on the implemented NB, SVM and LR models are tabulated below in Table II, Table III, Table IV and Table V. Fig. 4, Fig. 5, Fig.6 and Fig.7 gives the graphical representation of the obtained results. These search terms are considered in order to support testing different stages of the proposed system and doesn't form part of the test dataset. The dataset is composed of summary and keyword attributes generated for a video corpus comprising of 100 video lectures.

Table- II: Recall achieved with ML Classification algorithms

Recall / Search Term	Algorithm		
	Naive Bayes	*Support Vector Machine*	*Logistic Regression*
Safety management	1	0.5	0
Water resource management	1	1	0.67
Raster graphics	1	1	1
Lift	1	1	1
Normalization	1	1	1

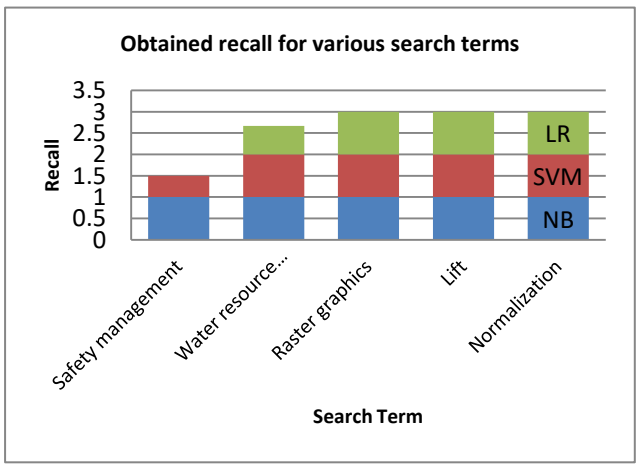

Fig. 4. Recall achieved with ML Classification algorithms

Table- III: Precision achieved with ML Classification algorithms

Precision / Search Term	Algorithm		
	Naive Bayes	*Support Vector Machine*	*Logistic Regression*
Safety management	1	0.125	0
Water resource management	1	1	0.133
Raster graphics	1	0.5	0.5
Lift	1	1	1
Normalization	0.5	0.5	0.5

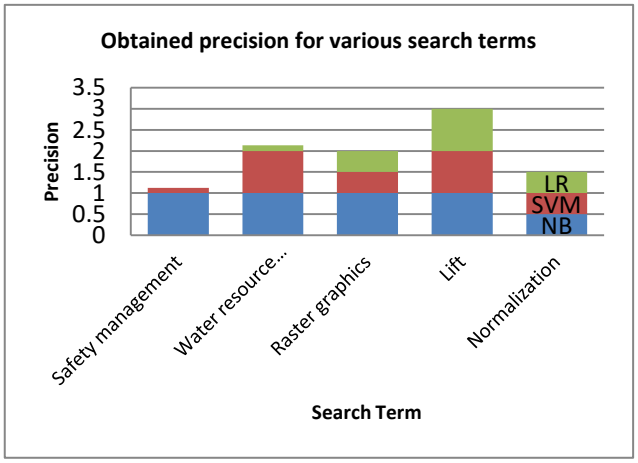

Fig. 5. Precision achieved with ML Classification algorithms

Table-IV: F-Score achieved with ML Classification algorithms

F-Score / Search Term	Algorithm		
	Naive Bayes	*Support Vector Machine*	*Logistic Regression*
Safety management	1	0.22	0
Water resource management	1	1	0.178
Raster graphics	1	0.67	0.67
Lift	1	1	1
Normalization	0.67	0.67	0.67

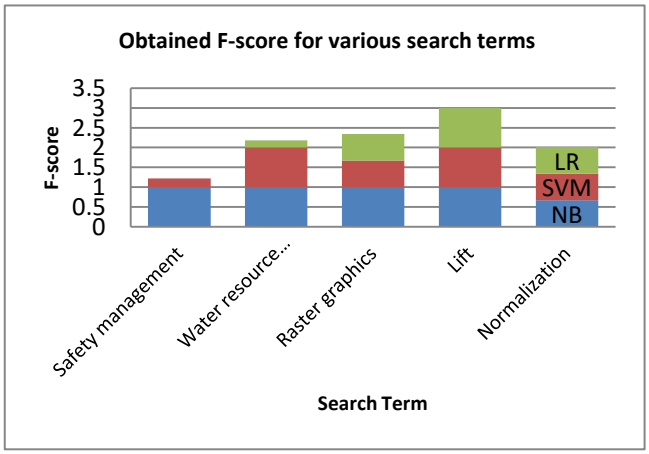

Fig. 6. F-score achieved with ML Classification algorithms

Table- V: Accuracy achieved with ML Classification algorithms

Accuracy / Search Term	Algorithm		
	Naive Bayes	*Support Vector Machine*	*Logistic Regression*
Safety management	1	0.125	0
Water resource management	1	1	0.125
Raster graphics	1	0.5	0.5
Lift	1	1	1
Normalization	0.5	0.5	0.5

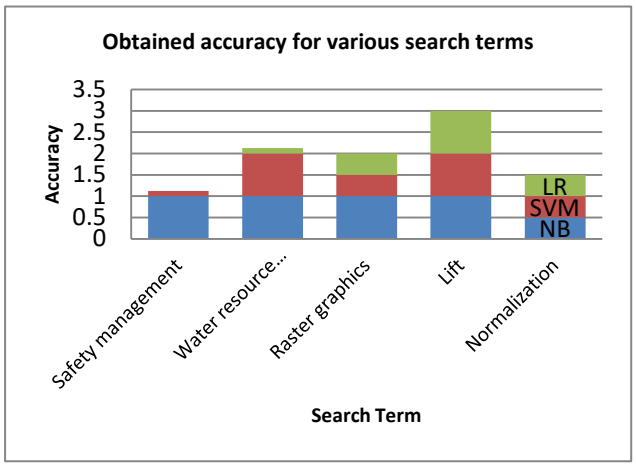

Fig. 7. Accuracy achieved with ML Classification algorithms

The results illustrate that the NB algorithm achieved state of art results. Being a simple probabilistic classifier model, it makes an assumption of independence of features so that predictions can be made independently and also very promptly. NB gives better results when there is less overlapping between the class variables as it assumes the independence of class variables [18]. SVM on the other hand is geometric in nature and looks at interactions between the features. LR assumes the log of the target variable is a linear function of parameters and is good for binary classification. Therefore, for the particular features of the data considered NB is the champ. The performance of the search implemented on the NB ML text classification model is compared with the regular search with respect to the time taken for displaying the search results. The search time taken by the implemented system drastically reduced by around 5-7 times when compared with the normal search operation. The results of which are illustrated in Table VI and Fig 8.

Table- VI: Time taken for search

Time taken(msec) / Search Term	Normal Search	Search based on NB model
Safety management	792.37	259.60
Water resource management	1573.99	296.87
Raster graphics	1896.43	319.88

Lift	855.99	284.87
Normalization	2183.00	343.72

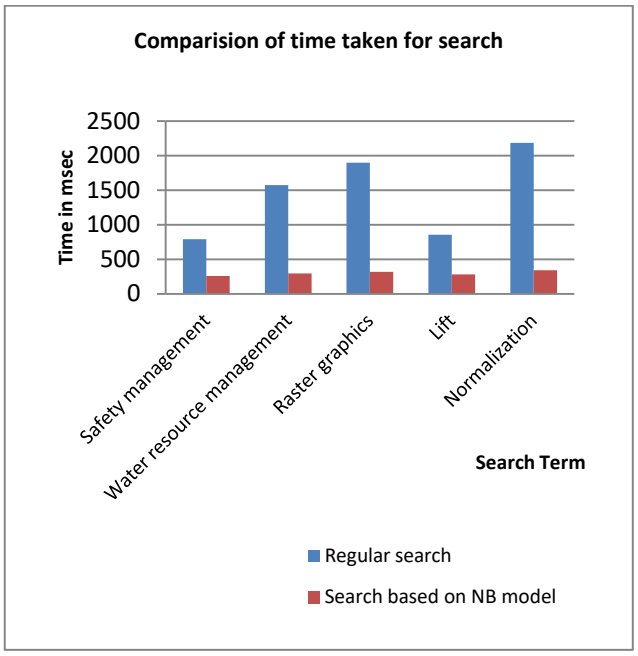

Fig. 8. Comparison of time taken for search

VI. CONCLUSION AND FUTURE WORK

In this paper, ML text classification based lecture video retrieval system is presented. The work was carried out on the video lecture corpus of 100 videos downloaded from YouTube. The audio content is extracted and subjected to resampling. The texts transcripts are extracted from the resampled audio file. Automatic summary and keywords extraction is carried out from the text transcripts. The generated summary and keywords are used for training the ML text classification models. Three techniques namely NB, SVM and LR are implemented for analyzing the technique that best performs on the dataset. Each of these techniques are used for implementing the search based on the ML model. The NB model showed high accuracy of predictions on the unseen search queries. Also the search implemented based on the ML model generated results in the highly reduced time. The results of the search are ranked in accordance with the scores calculated based on TFIDF and Cosine Similarity. The experimental analysis conclude that the proposed method is effective and accurately retrieves videos relevant to different categories. The future work may include the implementation of clustering techniques for further reducing the search time and also increasing the relevance of search. Video indexing of the lecture videos can also be included.

ACKNOWLEDGMENT

This work is supported by University Grants Commission (UGC) under Minor Research Project titled "Fast Content Based Search, Navigation and Retrieval system for E-Learning". Project Id: F.No:4-4/2015(MRP/UGC-SERO).

REFERENCES

1. Aasif Ansari; Muzammil H Mohammed, "Content based Video Retrieval Systems - Methods, Techniques, Trends and Challenges", International Journal of Computer Applications, Vol. 112 No. 7, 2015, pp. 13-22.
2. Jose M. Perea-Ortega; M. Teresa Martin-Valdivia; Arturo Montejo-Raez; L. Alfonso Urena-Lopez, "A Content-based Information Retrieval System for Video Searching", Proceedings of the IEEE Eighth International Symposium on Natural Language Processing, Bangkok, Thailand, 2009, pp. 21-25.
3. M. Ikonomakis; S. Kotsiantis; V. Tampakas, "Text Classification Using Machine Learning Techniques", WSEAS Transactions on Computers, Vol. 4 No. 8, 2005, pp. 966-974.
4. Sebastiani F, "Machine Learning in Automated Text Categorization", ACM Computing Surveys, Vol. 34 No. 1, 2002, pp. 1-47.
5. Haojin Yang; Christoph Meinel, "Content Based Lecture Video Retrieval Using Speech and Video Text Information", IEEE transactions on learning technologies, Vol. 7 No. 2, 2014, pp. 142-154.
6. S. Repp; A. Gross; C. Meinel, "Browsing within lecture videos based on the chain index of speech transcription", IEEE transactions on learning technologies, Vol 1 No. 3, 2008, pp. 145–156.
7. M.Ayesha Parveen; P. Deepan; S. Jummera Nasrin, "Video Data Retrieval Based On Text Content Detection" in ICEICT 2016, Proceedings of the Second International Conference on Electrical, Information and Communication Technology, 2016, pp. 38–43.
8. M. R. Naphade; T. Kristjansson; B. Fery; T. S. Huang, "Probabilistic Multimedia Objects Multijects: A novel Approach to Indexing and Retrieval in Multimedia Systems" in ICIP 1998, Proceedings of the IEEE International Conference on Image Processing, Chicago, USA, 1998, pp. 536-540.
9. John R. Smith; Savitha Srinivasan; Arnon Amir; Sanker Basu; Giri Iyengar; Ching-Yung Lin; Milind Naphade; Dulce Ponceleon; Belle Tseng, "Integrating Features, Models, and Semantics for TREC Video Retrieval" in TREC-10 2001, Proceedings of the 10th Text Retrieval Conference, Gaithersburg, Maryland, 2001, pp. 240-249.
10. M. Grcar; D. Mladenic; P. Kese, "Semi-automatic categorization of videos on videolectures.net, Machine Learning and Knowledge Discovery in Databases", ECML PKDD 2009, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, Vol. 5782, 2009, pp. 730-733.
11. Rajendra K R; O R Devanand; S. K. Sahay, "Web Document Clustering and Ranking using Tf-Idf based Apriori Approach", IJCA Proceedings on ICACEA, No. 2, 2014, pp. 34-37.
12. I. Mani; M. T. Maybury, "Advances in Automatic Text Summarization", MIT Press, Cambridge, MA, USA, 1999.
13. I. Rish, "An empirical study of the naive bayes classifier" in IJCAI 2001, workshop on empirical methods in artificial intelligence, 2001, pp. 41–46.
14. Thorsten Joachinms, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features" in ECML 1998, Lecture Notes in Artificial Intelligence, Springer, Berlin, Heidelberg, Vol. 1398, 1998, pp, 137-142.
15. Jiaul H. Paik, "A Novel TF-IDF Weighting Scheme for Effective Ranking" in SIGIR 2013, Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval, New York, USA, 2013, pp. 343-352.
16. Bing Zhou; Yiyu Yao, "Evaluating information retrieval system performance based on user preference", Journal Of Intelligent Information Systems, Vol. 34 No. 3, 2010, pp 227-248.
17. Zhang Harry, "The Optimality of Naive Bayes" in FLAIRS 2004, Proceedings of the 17th International FLAIRS Conference, Florida, USA 2004.

AUTHORS PROFILE

Ms. Lakshmi Haritha Medida is currently pursuing Ph D from JNTUA, Ananthapuramu, Andhra Pradesh in the Department of Computer Science and Engineering and received her M. Tech Degree in Computer Science and Engineering from JNTUK, Andhra Pradesh in the year 2016 and pursued her B. Tech in Electronics and Communication Engineering from JNTUK, Andhra Pradesh in the year 2011. Her research interests include Information Retrieval, Data Mining and Image Processing.

Dr. K Ramani is working is as a Professor and Head in the Department of Information Technology, SVEC, A. Rangampet and received her Ph.D Degree from JNTUK, Andhra Pradesh in the year 2010. She has more than 16 years of teaching experience. Her areas of interests include Image Processing, Data mining, Computer Networks and Software Engineering.