VTutor: A Platform for Improving Searchability and Interactivity of Recorded Lectures

Damsith Karunaratna

dept. of Software Engineering

Sri Lanka Institute of

Information Technology

Malabe, Sri Lanka

dammakaru@gmail.com

Sachini Epa

dept. of Software Engineering

Sri Lanka Institute of

Information Technology

Malabe, Sri Lanka

sachi.epa96@gmail.com

Isuru Hettiarachchi

dept. of Software Engineering

Sri Lanka Institute of

Information Technology

Malabe, Sri Lanka

isuru71@gmail.com

Nuwan Kodagoda

dept. of Software Engineering

Sri Lanka Institute of

Information Technology

Malabe, Sri Lanka

nuwan.k@sliit.lk

Stefinie Fernando

dept. of Software Engineering

Sri Lanka Institute of

Information Technology

Malabe, Sri Lanka

minolispencer@gmail.com

Kushnara Suriyawansa

dept. of Software Engineering

Sri Lanka Institute of

Information Technology

Malabe, Sri Lanka

kushnara.s@sliit.lk

*Abstract*—Recorded lectures have gained popularity as a method of delivering lecture content as this approach gives learners distinct advantages such as the ability to follow lectures without time or location constraints and consume the lectures at their own pace [1] . Although recorded lectures have many advantages, they tend to be lengthy and tedious to watch. They also prove cumbersome when specific information needs to be extracted from them. Another drawback is that recorded lecture videos fail to show the connection between the lecture and its supporting material such as lecture slides and questionnaires. Though many of the existing platforms allow videos to be edited to make them more interactive, the methods employed by these platforms are manual and therefore tedious and time consuming. VTutor is a web platform that aims to address these drawbacks by introducing automation into the video enhancement process and by combining the lecture material to create an enhanced user experience. Specifically, VTutor allows users to navigate through a lecture video using sub topics, its corresponding slides and code samples. Furthermore it is also equipped with the ability to automatically generate questions based on lecture content thus improving the level of engagement that a learner has with the lecture.

Keywords— *E-Learning, Recorded Lectures, OCR, Topic Based Video Segmentation, Question Generation, Text Extraction, Code matching, Slide matching*

# INTRODUCTION

Today, e-learning has become an essential component of higher education for both teachers and students. According to a study on the effectiveness of e-learning on education, it was found that students nowadays are more satisfied with web enhanced learning when compared to a traditional classroom environment [1].  Therefore, it is commonplace to see universities and higher education institutes adopting some form of e-learning to assist their students. In fact, many institutes use their own customized version of a Learning Management System (LMS) to provide online course material.

Online education is beneficial in many ways. For instance, as lecture content is always available online, the possibility of missing a lecture is low and teachers can ensure that students have access to course material irrespective of time and location [1]. Recorded lecture videos also enable students with different styles of learning and different levels of understanding to obtain a better grasp of the subject. For example, those who are familiar with the work can skip ahead to a section of interest while those that need more time to understand the concepts can pause and rewind to digest the lecture at their own pace [2, 3].

Many LMSs enable lecturers to upload course material such as tutorials, lab sheets, lecture slides and recorded lecture videos. Whilst videos are effective because they address both the visual and auditory aspects of teaching [4], many students find it tedious to watch recorded lecture videos because of its duration, which normally lasts around 1 - 3 hours and its lack of interconnectivity and relevance to other course material [5].

Although platforms such as LearnWorlds[[1]](#footnote-2), Echo360[[2]](#footnote-3) and Techsmith Relay[[3]](#footnote-4) allow users to create interactive course material, the methods used by them are time consuming as they require the lecturer to spend time editing and making the video interactive. Other applications like Camstasia[[4]](#footnote-5) allow users to create interactive learning material only if the videos are recorded using their proprietary software, which is quite impractical in a normal lecture hall.

V-Tutor is a smart web platform which is capable of automatically transforming raw lecture material into content which is more interactive while saving the time and effort of the content creators. It provides many interactive ways for leaners to navigate through lecture videos such as using topics, slides and specific lines in code samples. In addition, it is also capable of generating content-based questions from the videos which can be utilized by the learners to check their knowledge on the key points of a specific lecture video.

# Literature Review

The following section reviews existing literature in each domain relevant to the VTutor system’s functionality.

## Code Matching

Several studies have been conducted on identifying source code in video and image files. In a research conducted in 2011, Kambathula and Iyer suggest a system which would enable lecture videos to be tagged automatically so that different sections can be identified easily [6]. Their system can highlight portions of each video in response to user queries. Luca Ponzanelli et al in 2016 introduced CodeTube [7], a similar search engine which when given a query, returns self-contained fragments of the corresponding lecture videos. In a research conducted in 2018, a deep-learning approach which leverages Convolutional Neural Networks (CNNs) to classify the presence or absence of Java code in video frames is proposed [8]. Their system can achieve an average accuracy of 98%. However it is limited to identifying code in the Java language and cannot be successfully applied in a system which would analyze videos in many programming languages. Research on algorithms such as ResNet50 [9] and InceptionV3 [10] prove they are good candidates for this purpose.

Although the topic area (matching each line of code in a sample code file to its occurrence in the lecture video) has not been widely researched and would most likely have an algorithmic solution, research that has already been conducted on source code mining and text detection can be used as a basis to create a system which identifies relevant portions of a live-coding video which correspond to the source code file.

## Topic Segmentation

The most commonly used mechanism for segmenting videos is based on scene changes. A research conducted in 2000 used color histogram distance computation between successive images to detect scene changes [14]. Another research done by Zhang and Smoliar, proposed a system for detecting progressive transitions based on both motion and statistical analysis [15]. This mechanism of segmentation based on scene changes is not applicable to lecture videos, since they have very few scene changes and even these scene changes do not match the topic transitions [4].

A research done in 2013 [16], focused on segmenting lecture videos into topics by analyzing its supplementary synchronized slides. As stated in their paper, the mechanism was approximately 90% accurate, except that it always assumes that the slides are synchronized with the video streams. Since their method is based solely on matching slide content, its accuracy is limited only to certain types of videos.

Moreover, segmentation methods utilizing transcribed text or closed captions have also been researched. The main motivational factor for work in this area was the Topic Detection and Tracking (TDT) initiative conducted in 1998 [14]. Their algorithm is trained mainly using broadcast and news domain data sets. Unlike in the broadcast domain, speeches in lecture videos are often unscripted and spontaneous. Furthermore, the large training dataset used for many methods in TDT, is not available for lecture videos [4, 16, 17].

Inspired by the work on TDT, in 2004 Michael Chau, and his team members, conducted another research to identify topic changes based on multiple linguistic features [4]. Work in this area can be considered as a potentially successful solution and can be improved further as audio and the transcribed text extracted from lecture videos provide rich content information for topic change detection.

## Question Generation

Automatic generation of questions is an area that has undergone much research. Shah et al. suggest a method of generating Multiple Choice Questions (MCQs) for a text passage with the aid of a trained knowledge base developed using Wikipedia articles [18]. TEDQuiz [19] is a system that generates MCQs for *TED Talks* video clips using a graph-based algorithm. The system generates two types of questions. The first type is gist-content questions which ask about the overall theme of the content that is generated by identifying the most important sentence using LexRank [20] and creating distractors by less important sentences. The second type is detailed questions which use Heilman and Smith’s work [21, 22] to create question stem and selecting words for distractors using WordNet and similar corpus. A similar system which analyzes a text transcript of a video lecture to suggest self-assessment items at runtime by identifying discourse boundaries from the lecture and retrieving related Wikipedia text segments is seen in [23].

Apart from this, automatic question generation was carried out using ontology-based strategies. SeMCQ [25] is a Protégé plugin created for automatic ontology-driven multiple question test generation. OntoQue [26] is an automatic question generation engine based on domain ontologies which can generate MCQ, true/false questions and fill-in-the-blank questions. Papasalouros et al. suggest an approach of generating MCQs based on domain-specific ontologies that use simple natural language generation techniques [27].

## Slide Matching

Several research projects have been carried out related to synchronization of lecture slides with the video. Among them, a research has been done by the Hongkong University and City University of HongKong proposed a system which utilizes OCR [28]. In this approach, geometry-based approach which works well with images with less noise. Super Reconstruction is used to enhance the visual quality of the images with the purpose of making them appropriate for commercial OCR systems.

Zentation [29] is an online tool that can be used to synchronize lecture slides with the video. However, it is not perfect when it comes to synchronization since most of the time this works well with presentations that have a smaller number of slides.

## Denoising Audio

Several studies have been carried out in this area with the aim of reducing background noise and enhancing intelligibility [30, 31, 32, 33, 34]. Removal of high frequency noise for speech enhancement with frequency response masking (FRM) has also been implemented. A filter has been designed to have impulse responses associated with various cut off frequencies to minimize the error when comparing the original speech signal and the filtered speech signal [35].

Audacity [36] is a tool that can be used to remove background noise of an audio and it can also be used to enhance the audio quality. In addition, this tool enables the user to change the pitch without any changes in the tempo. However, Audacity is not capable of removing irregular noises like traffic sounds and sounds of an audience which can occur in lecture videos.

# METHODOLOGY

## Code matching

Live coding lectures, often seen in the field of Information Technology, usually feature a screen capture in which the instructor types code into a text editor or Integrated Development Environment (IDE) while narrating. The searchability of such a video would be greatly improved if the student is able to use the source code discussed within the lecture as an index to pinpoint the time at which each line of code is discussed in the video.

The following steps illustrate how the system carries out code matching.

*Frame extraction:* The video is sampled at a rate of 0.5 frames per second using the FFmpeg[[5]](#footnote-6) open source library to create a set of images *I.* The sampling rate is set as such so that the number of duplicate frames is reduced. The set *I* is further narrowed down by performing a standard pixel-wise comparison between consecutive frames and removing those with high similarity.

*Text extraction:* For each image , text is extracted using the Optical Character Recognition capabilities of TESSERACT-OCR[[6]](#footnote-7) and stored in a set of text files *T* which maintains the chronological ordering of each frame. The filename of each file is the timestamp of the frame from which it was extracted. This is beneficial for later stages of the algorithm.

*Source code matching:* The following algorithm demonstrates the source code matching process. The *gestalt()* function utilizes an algorithm based on Ratcliff and Obershelp’s ‘gestalt pattern matching’ algorithm [11]. This function finds the longest contiguous matching subsequence between two hashable strings and returns a ratio of similarity between the sequences.

The efficiency of the algorithm mentioned below is greatly improved by leveraging the VGG16 network [12] shown in Figure 1 to classify frames as those “*containing code*”. Thus, effectively reducing the number of iterations required to match each line of code.

Given: *T, Source code file (C)*

INITIALIZE reversed\_timestamps= reverse(*T*); result= []; i=0;

FOR *sc\_line* in *C*

max\_ratio = cur\_ratio = 0, max\_ratio\_frame = ‘’

FOR frame IN reversed\_timestamps

FOR line IN frame DO

cur\_ratio = *gestalt* (sc\_line, line);

IF cur\_ratio >= max\_ratio THEN

max\_ratio = cur\_ratio

max\_ratio\_frame = timestamp

ENDIF

ENDFOR

ENDFOR

result[i++] = {sc\_line, max\_ratio\_frame};

ENDFOR

OUTPUT result



Figure 1 - VGG16 Network

## Topic based segmentation

Usually recorded lecture videos are very long in duration and consists of several subtopics. Some students may find it difficult to gasp all main points within a lecture at one go. Therefore tagging videos with timestamps based on topic transitions within the video is considered highly favorable.

As elaborated in the previous section of this paper, people have tried different mechanisms for identifying topic transitions within lecture videos. In this research we focus on a linguistic approach based on video transcripts. Video transcripts provide rich content information and will be ideal for detecting topic transitions. Also, generally computing audio and visual features is a very time-consuming process, while computing text takes comparatively less amount of time. Our approach consists of two main parts: Topic modelling and text segmentation based on the topic model. In our system topic modelling is achieved through a state-of-art algorithm called Latent Dirichlet allocation (LDA) and the topical segmentation is done using topic tiling algorithm which is based on LDA.

LDA is a form of unsupervised learning and is one of the most popular generative, probabilistic text modelling techniques in machine learning. LDA works with the assumption that each document was generated by picking a set of topics and then for each topic picking a set of words [13]. Input to LDA is a collection of documents, which in our case is the video transcripts. LDA considers each document is in the form of a ‘bag of words’ [13]. Therefore, in order to transform raw transcripts into a bag of words structure, it need to undergo data pre-processing. Standard data pre-processing for LDA consists of the following steps:

* *Tokenization* - Split text into words, lowercase the words and remove punctuation
* *Remove stopwords* - stopwords are most commonly used words in a language, for an example word ‘the’ in English language. Stopwords are usually not relevant to any topics hence, removing these from the corpus supports LDA to extract topics more accurately.
* *Lemmatization* - third person words are changed to first person, and verbs in different tenses are converted to present tense.
* *Stemming* - words are reduced to their root form.

Once preprocessing is done the corpus is subjected to LDA. The generative process of LDA is as follows:

For each document w in a corpus D:

1. Choose N Poisson (ξ).

2. Choose Θ Dir (α).

3. For each of the N words wn:

a. Choose a topic zn Multinomial (Θ).

Choose a word wn from 𝑝(𝑤𝑛 |𝑧𝑛, 𝛽), a multinomial probability conditioned on the topic zn. [14]

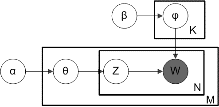


Figure 2 Plate notation for LDA with Dirichlet-distributed topic-word distributions [15]

α - per-document topic distributions,

β - per-topic word distribution,

θ - topic distribution for document m,

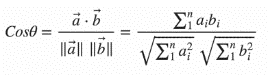
φ - word distribution for topic k,

z - topic for the n-th word in document m

w - specific word

Next the transcripts are subjected to topic tiling which is a LDA based text segmentation algorithm. Topic Tiling was first introduced by Riedl and Biemann in 2012. When compared with other segmentation methods based on LDA, topic toiling is computationally less expensive as it performs segmentation in linear time. [16]

In topic tiling, initially the document is split into units based on sentence boundaries. Each sentence is represented by a N dimensional vector where N is the number of topics represented in the topic model. Then the coherence between each sentence is computed by calculating the cosine similarity between two adjacent sentences using Equation 1 [17].



Equation 1 Cosine Similarity Formula

*Where* ***ai*** *and* ***bi*** *are the vectors of two adjacent sentences.*

Values close to one indicates a substantial connectivity of the two sentences whereas, values close to 0 indicate a maginal relatedness. The coherence are then plotted to trace the local minima, which is utilized in identifying the possible segmentation boundaries.

The final outcome of the topic tiling algorithm will be a XML file with text segments. This XML file is then read by another algorithm and matched with a timestamped transcript of the video to extract the timestamps where the segmentations occur. Outcome of the topic modelling component will be a set timestamps along with set of key words suggested from the topic model. The high-level view of the overall segmentation process is shown in Figure 3.

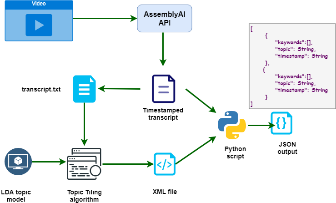


Figure 3 High level view of overall Topic based Segmentation process

## Question Generation

Automatic question generation is used as a feature to increase the interactivity of video lectures as well as allowing learners to evaluate their knowledge on learning outcomes of the lecture. The main of automating the question generation process is to increase the efficiency of lecturers by minimizing the time spent on creating questions. Question generation component has 3 sub components as mentioned below.

*Extracting Input Text:* Question generation process is based on raw text from sentences or paragraphs with factual statements and the work carried out by Michael Heilman and Noah A. Smith on Question Generation via Overgenerating Transformations and Ranking [18]. For this process factual statements need to be collected for the given lecture video. Use of transcript of lecture videos is not a feasible option as in-class lectures tend to carry out more noise and in a verbal manner that content and facts are scattered throughout the lecture. To address this issue, we identify Wikipedia articles that correlates with the main subject discussed in the lecture using web scraper implemented using Wikipedia-API[[7]](#footnote-8). To have a 100% accuracy on the main subject since an error of this will effect on the output of the process in a huge margin, we get human user input directly.

*Question Generation:* Once the Wikipedia articles identified text content is extracted using Wikipedia-API[[8]](#footnote-9) to use as the input of an implementation of Heilman’s and Smith’s work mentioned earlier which includes three stages.

*Stage 1 –* Factual statement extraction from input text using semantic entailments, removing discourse markers and adjunct modifiers, splitting conjunctions.

*Stage 2 –* Question item creation after identifying answer phrases and applying wh-movement to the sentences. Steps in of this stage is shown in figure 4.

*Stage 3 –* Statistical question ranking to eliminate invalid and unacceptable questions.

Question

Input sentence

Mark Unmovable Phrases

Generate Possible Question Phrases

Subject-Auxiliary Inversion

Remove Answer, Insert Question Phrase

Decomposition of main verb

Post Processing

Figure 4 Question item creation steps

*Question Reviewing and Presenting to the end user:* As the final process generated questions are presented to the lecturer before publishing the video where he or she can review, edit questions, answers and add distractors.

## Slide Matching

Electronic slides are often added to Learning management Systems (LMS) together with the Recorded lecture videos. In the present, the platforms that is used do not provide the users the privilege to match a certain electronic slide with the lecture video in order to find the point of time in the video where the selected slide is discussed. Therefore, having a methodology to navigate through the lecture video using the electronic slides as an index can considerably improve the learnability and the ability to search while making the learning process efficient and easier.

As a platform that is developed to address the gap between the increasing requirements of the learners and existing products in the same domain, VTutor has provide a tool for the learners to match the electronic slides with the relevant lecture video. Therefore, once the user selects a slide from the slide deck displayed, the platform will automatically take the user to the point of time in the lecture video, where the selected slide is being discussed.

Below are the steps that the platform mainly performs in order to provide the above discussed functionality.

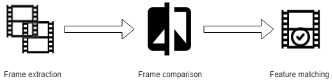


Figure 5 Overview of slide matching component

*Frame extraction* : The lecture video generates frames at a rate of one frame per every second using Ffmpeg, a free and open source library for handling audio, video and other multimedia types. Each frame generated in this step is named in such a way that it has the time of occurrence (in seconds) with the name of the frame.

*Frame Comparing* : The user-selected slide is read and compared against all the frames of the lecture video generated in the previous step and the difference between every frame and the selected slide (as a percentage) is calculated using OpenCV2, an open source library for image processing and computer vision.

*Slide Matching* : As a result of the slide comparison carried out in the previous step, the algorithm chooses the first frame having less than 1% of difference with the compared slide and recognizes it as the first occurrence of the selected slide in the lecture video.

In the final step, after identifying the first frame with a difference less than 1% through the algorithm, the platform automatically navigates the lecture video to the point of time where that particular frame was occurred with the aid of the naming convention used when extracting frames using Ffmpeg in the first step.

Below mentioned is the algorithm that is used for frame comparison in the platform.

Given: The selected slide *m*, frames generated for the lecture video *frames*

FOR frame in frames:

frame\_read = cv2.imread(frame, CV\_LOAD\_IMAGE\_GRAYSCALE);

slide\_read = cv2.imread(m, CV\_LOAD\_IMAGE\_GRAYSCALE);

res = cv2.absdiff(frame\_read, file\_read);

res = res.astype(np.unit8);

diff\_precentage = (numpy.count\_nonzero(res) \* 100) / res.size;

IF diff\_precentage < 1 THEN

OUTPUT frame;

break;

END IF

END FOR

At the current stage of the research the platform is only able to build the synchronization between screen-recorded lecture videos and electronic lecture slides.

## Audio Denoising

A raw lecture video that is recorded using a normal microphone usually contains a significant amount of background noise which affects the quality of the audio. In the VTutor platform, the lecture video is automatically de-noised once it is uploaded, before making it available for the users to access. In order to do this, the platform makes use of the Deep Affects Audio Denoising API[[9]](#footnote-10).

Once the raw audio that is extracted from the video is sent to the API through a POST request, the Deep Affects audio de-noising API removes noise from the audio signal and returns the de-noised audio clip to the web hook URL passed. The API is organized around REST and all the requests are made over SSL. All the requests and responses including errors are encoded in JSON. Among all the de-noising APIs available, Deep Affects has a large free tier which can handle 100 requests per day for a single user, and 5 requests can be made per minute to the API. It has reasonable pricing when the free tier is exceeded.

# RESEARCH FINDINGS AND RESULTS

The main objective of this research is to provide an easy to use, automated platform which provides a quick and efficient way for lecture creators to deliver video lectures which are more interactive and have increased searchability. This segment of the paper presents a discussion of the results obtained for each component of the platform implemented as a solution.

The trained VGG16 neural network could identify frames with a mean accuracy of 95.217 percent, a precision of 0.849 and a recall of 0.939. Indicating that the classification of frames as “containing code” can be carried out with a good degree of confidence. Furthermore, the code matching algorithm was able to match each line with perfect accuracy although it takes a considerable amount of time to complete (On average 0.7 times the length of the video), however the advantage that no human intervention is required for this purpose outweighs this slight drawback.

The topic-based video segmentation process was tested against manual segmentation by a human to evaluate its accuracy. Table 1 and Table 2 shows evaluation results on two sample videos compared to manual segmentation of the same video in a human perspective in two scenarios: number of segments provided, and number of segments not provided.

On average there is time difference of 2.02 minutes between the timestamps suggested by the algorithm and when the same process is done manually by a human. Thus, the above method can be considered as a feasible and an effective way of identifying topic transitions within a video.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Video Title - OOC Lecture 5** | | | | | |
| **Number of segments not provided** | | | | | |
|  | | **Topic Tiling Algorithm** | | **Manual Segmentation** | |
| **No of segments** | | 5 | | 3 | |
| **Number of segments provided** | | | | | |
|  | **Timestamps (hh:mm:ss)** | | | | |
| **Using Algorithm** | | **Manual Segmentation** | | **Time difference (minutes)** |
| **Segment 1** | 00:04:45 | | 00:02:33 | | 2.2 |
| **Segment 2** | 00:17: 54 | | 00:19:31 | | 1.62 |
| **Segment 3** | 00:42:11 | | 00:39:28 | | 2.71 |

Table 1 Sample 1 text segmentation results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Video Title - OOC Lecture 6** | | | | |
| **Number of segments not provided** | | | | |
|  | **Topic Tiling Algorithm** | | **Manual Segmentation** | |
| **No. of segments** | 8 | | 5 | |
| **Number of segments provided** | | | | |
|  | **Timestamps (hh:mm:ss)** | | | |
| **Using Algorithm** | **Manual Segmentation** | | **Time difference in minutes** |
| **Segment 1** | 00:02:10 | 00:03:37 | | 1.45 |
| **Segment 2** | 00:10:40 | 00:08:32 | | 2.14 |
| **Segment 3** | 00:16:12 | 00:13:43 | | 2.48 |
| **Segment 4** | 00:25:56 | 00:26:34 | | 0.64 |
| **Segment 5** | 00:53:39 | 00:50:22 | | 3.28 |

Table 2 Sample 2 text segmentation results

The accuracy of the above results depends on the accuracy the video transcripts and how well the LDA model is trained. Transcription process is mainly affected by 2 factors: background noise level and the accent used in the audio. In our research evaluations were done mainly using videos with Asian English accent.

The question generation component was tested on a system with an Intel i5 1.70GHz processor and 8 gigabytes of RAM. A Wikipedia article on C++ classes was used as the input source. After feeding 160 sentences extracted from the Wikipedia article to the system, 325 question items were generated in a time period of 21 minutes.

A qualitative test was carried out on a sample of 100 question items. Out of these 325 question items. A group of 3 final year BSc. in Information Technology undergraduates were given the sample questions. Table 3 shows the result of qualitative test.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rating** | **Valid** | **Vague** | **Grammar Error** | **Wrong Answer** |
| **Percentage (Average)** | 50% | 15% | 12% | 23% |

Table 3 Qualitative test results for generated questions

The performance of the slide matching component is compared with the manual results and against existing algorithms such as SIFT, SURF and ORB. Below Table 4 and Table 5 shows the performance comparison of the slide matching component against above mentioned algorithms.

|  |  |
| --- | --- |
| Video Title – OOC Lecture 5 | |
| Algorithm | Time taken to find the match (in seconds) |
| Manual Matching | 47 |
| SIFT | 6 |
| SURF | 8 |
| ORB | 11 |
| Our solution | 8 |

Table 4 Performance comparison – OOC Lecture 05

|  |  |
| --- | --- |
| Video Title – OOC Lecture 6 | |
| Algorithm | Approximate time taken to find the match (in seconds) |
| Manual Matching | 32 |
| SIFT | 6 |
| SURF | 9 |
| ORB | 11 |
| Our solution | 8 |

Table 5 Performance comparison – OOC Lecture 06

By comparing the results, it is evident that the algorithm used in the discussed platform has a considerable efficiency when compared with other algorithms. SIFT having the least time to find the match cannot be used in this context as it requires high powered devices to perform efficiently.

# CONCLUSION & FUTURE WORKS

Recorded lecture videos are a widely used method of conveying lectures in many universities. They come with a myriad of benefits. Recorded lectures are usually lengthy and lack interactivity which results in poor learner engagement. The main intention of this research is to provide a means by which content creators can increase the searchability of lecture videos by indexing them using unconventional means such as lecture slides and source code files in addition to automatically generated topics which will provide a novel and efficient way to navigate through the lecture video. The value of the system is further improved by the fact that questions are automatically generated based on the video content.

In future works the VTutor system can be improved upon by increasing the accuracy of the code identification and slide matching algorithms as well as providing an efficient means of generating question distractors. In addition, the performance can be further improved enabling the possibility of matching the electronic slides with other types of lecture videos such as speaker embedded lecture videos. Enhancing the audio quality of the lecture videos after background noise removal is also a problem that can be addressed. Another possible feature which would improve usability is the ability to change the accent of the speaker according to the user’s learning style.

##### Acknowledgment

We would like to convey our sincere appreciation to the administration of Sri Lanka Institute of Information Technology (SLIIT) for providing us with a suitable environment and prerequisites to complete this project. We also want to express our gratitude to our supervisors Dr. Nuwan Kodagoda and Miss Kushnara Suriyawansa for their support and guidance in making this project a success.

##### References

|  |  |
| --- | --- |
| [1] | "Importance and Effectiveness of E-learning," 1 December 2015. [Online]. Available: https://higheredrevolution.com/importance-and-effectiveness-of-e-learning-9513046ed46c. [Accessed 6 March 2019]. |
| [2] | M. Brandsteidl, T. Mayerhofer, M. Seidl and C. Huemer, "Replacing traditional classroom lectures with lecture videos: an experience report," in *Proceedings of the 8th edition of the Educators' Symposium*, 2012. |
| [3] | M. Maher, H. Lipford and V. Singh, "Flipped classroom strategies using online videos," UNC Charlotte, North Carolina, 2013. |
| [4] | M. Chau, M. Lin, J. F. Nunamaker and H. Chen, "Segmentation of Lecture Videos Based on Text: A Method Combining Multiple Linguistic Features," in *Proceedings of the 37th Hawaii International Conference on System Sciences*, Hawaii, 2004. |
| [5] | Z. Woolfitt, "The effective use of video in higher education," 2015. [Online]. Available: https://www.inholland.nl/media/10230/the-effective-use-of-video-in-higher-education-woolfitt-october-2015.pdf. [Accessed 6 March 2019]. |
| [6] | V. K. Kamabathula and S. Iyer, "Automated Tagging to Enable Fine-Grained Browsing of Lecture Videos," in *2011 IEEE International Conference on Technology for Education*, 2011. |
| [7] | Luca Ponzanelli et al, "CodeTube: Extracting Relevant Fragments from Software Development Tutorials," in *Proceedings of the 38th International Conference on Software Engineering Companion - ICSE ’16*, 2016. |
| [8] | J. Ott, A. Atchison, P. Harnack, A. Bergh and E. Linstead, "A deep learning approach to identifying source code in images and video," in *Proceedings of the 15th International Conference on Mining Software Repositories - MSR ’18*, 2018. |
| [9] | K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. |
| [10] | C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. |
| [11] | Xlinux.nist.gov, "Ratcliff/Obershelp pattern recognition," 2019. [Online]. Available: https://xlinux.nist.gov/dads/HTML/ratcliffObershelp.html. [Accessed 02 Aug 2019]. |
| [12] | K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR,* vol. abs/1409.1556, 2014. |
| [13] | T. Doll, "LDA Topic Modeling: An Explanation," 25 June 2018. [Online]. Available: https://towardsdatascience.com/lda-topic-modeling-an-explanation-e184c90aadcd. [Accessed 8 August 2019]. |
| [14] | A. Onan, S. Korukoglu and B. Hasan, "LDA-based Topic Modelling in Text Sentiment classification: An Empirical Analysis," *International Journal of Computational Linguistics and Applications,* vol. VII, pp. 101-119, 2016. |
| [15] | "Latent Dirichlet allocation," [Online]. Available: https://en.wikipedia.org/wiki/Latent\_Dirichlet\_allocation. [Accessed 8 August 2019]. |
| [16] | "TopicTiling," [Online]. Available: https://github.com/riedlma/topictiling. [Accessed 6 August 2019]. |
| [17] | R. Martin and B. Chris, "How Text Segmentation Algorithms Gain from Topic Models," in *2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montreal, 2012. |
| [18] | M. Heilman and N. A. Smith, "Question Generation via Overgenerating Transformations and Ranking," in *Technical report, Language Technologies Institute, Carnegie Mellon University Technical Report*, 2009. |
| [19] | M. J. Rubin, "The effectiveness of live-coding to teach introductory programming," in *Proceeding of the 44th ACM technical symposium on Computer science education - SIGCSE ’13*, Denver, Colorado, USA, 2013. |
| [20] | A. M. a. M. L. L. Ponzanelli, "StORMeD: Stack Overflow ready made data," in *Proceedings of MSR 2015 (12th Working Conference on Mining Software Repositories)*, 2015. |
| [21] | M. Lin, C. B. R. Diller, Ming, N. Forsgren, Y. Huang and J. F. Nunamaker, Jr, "Segmenting Lecture Videos by Topic: From Manual to Automated Methods," in *Eleventh Americas Conference on Information Systems*, Omaha, NE, 2005. |
| [22] | H. J. Zhang and S. W. Smoliar, "Developing power tools for video indexing and retrieval," in *SPIE'94 Storage and Retrieval for Video Databases*, San Jose, CA, USA, 1994. |
| [23] | X. Che, H. Yang and C. Meinel, "Lecture Video Segmentation by Automatically Analyzing the Synchronized Slides," in *MM’13*, Barcelona, Spain, 2013. |
| [24] | J. Allan, J. Carbonel, G. Doddington, J. Yamron and Y. Yang, "Topic detection and tracking pilot study: Final report," in *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*. |
| [25] | R. Shah, D. Shah and L. Kurup, "Automatic Question Generation for Intelligent Tutoring Systems," in *2017 2nd International Conference on Communication Systems, Computing and IT Applications (CSCITA)*, Mumbai, 2017. |
| [26] | Y.-T. Huang, Y.-M. Tseng, Y. S. Sun and M. C. Chen, "TEDQuiz: Automatic Quiz Generation for TED Talks Video Clips to Assess Listening Comprehension," in *2014 IEEE 14th International Conference on Advanced Learning Technologies*, Athens, 2014. |
| [27] | G. Erkan and D. R. Radev, "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization," in *Journal of Artificial Intelligence Research, Vol 22*, 2004. |
| [28] | M. Heilman, "Automatic Factual Question Generation from Text," Doctoral thesis, Carnegie Mellon University, Pittsburgh, 2011. |
| [29] | A. Krishna, P. Bhowmick, K. Ghosh, A. Sahu and S. Roy, "Automatic Generation and Insertion of Assessment Items in Online Video Courses," in *IUI Companion '15 Proceedings of the 20th International Conference on Intelligent User Interfaces Companion*, Atlanta, Georgia, 2015. |
| [30] | M. Heilman and N. A. Smith , "GoodQuestion! StatisticalRankingforQuestionGeneration," in *HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Log Angeles, California, 2010. |
| [31] | M. Tosic and M. Cubric, "SeMCQ – Protégé Plugin for Automatic Ontology-Driven Multiple Choice Question Tests Generation," in *Procs of the 11th International Protege Conference*, 2009. |
| [32] | M. Al-Yahya, "OntoQue: A Question Generation Engine for Educational Assesment Based on Domain Ontologies," in *2011 IEEE 11th International Conference on Advanced Learning Technologies*, Athens, 2011. |
| [33] | A. Papasalouros , K. Kanaris and K. Kotis, "Automatic Generation Of Multiple Choice Questions From Domain Ontologies," in *IADIS International Conference e-Learning 2008*, Amsterdam, 2008. |
| [34] | F. Wang, C. W. Ngo and T. C. Pong, "Synchronization of lecture videos and electronic slides by video text analysis," in *Proceedings of the eleventh ACM international conference on Multimedia - MULTIMEDIA ’03*, Hong Kong, 2003. |
| [35] | "Zentation.com - Webinar software," Zentation.com, 2019. [Online]. Available: http://www.zentation.com/. [Accessed 08 March 2019]. |
| [36] | K. Hymavathy and P. Janardhanan, "Noise Filtering in Speech Using Frequency Response Masking Technique," *International Journal of Emerging Trends in Engineering and Development,* 2013. |
| [37] | S. Muangjaroen and T. Yingthawornsuk, "A Study of Noise Reduction in Speech Signal Using FIR Filtering," in *Proceedings of the International Conference on Advances in Electrical and Electronics Engineering*, Pattaya, 2012. |
| [38] | T. L. Kumar, K. S. Rajan and ., "Noise Suppression in Speech Signals Using Adaptive Algorithms," *International Journal of Engineering Research and Applications,* vol. 2, pp. 718-721, 2012. |
| [39] | R. S. J. G. V. R. S. T. M. a. K. A. Aggarwal, "Noise Reduction of Speech Signal Using Wavelet Transform with Modified Universal Threshold," *International Journal of Computer Applications,* vol. 20, pp. 15-19, 2011. |
| [40] | E. Verteletskaya and B. Simak, "Speech Distortion Minimized Noise Reduction Algorithm," in *Proceedings of theWorld Congress on Engineering and Computer Science*, San Francisco, 2010. |
| [41] | M. Karam, H. F. Khazaal, H. Aglan and C. Cole, "Noise Removal in Speech Processing Using Spectral Subtraction," *Journal of Signal and Information Processing,* vol. 5, no. 2, pp. 32-41, 2014. |
| [42] | "Audacity - Free Open source, cross platform audio software," Audacity, 2019. [Online]. Available: https://www.audacityteam.org/. [Accessed 05 March 2019]. |
| [43] | A. Onan, S. Korukkoglu and H. Bulut, "LDA-based Topic Modelling in Text Sentiment Classification: An Empirical Analysis," *International Journal of Computational Linguistics and Applications,* vol. VII, pp. 101-119, 2016. |

1. https://www.learnworlds.com/ [↑](#footnote-ref-2)
2. https://echo360.com/ [↑](#footnote-ref-3)
3. https://www.techsmith.com/lecture-capture.html [↑](#footnote-ref-4)
4. https://www.techsmith.com/video-editor.html [↑](#footnote-ref-5)
5. https://ffmpeg.org/ [↑](#footnote-ref-6)
6. https://github.com/tesseract-ocr/ [↑](#footnote-ref-7)
7. <https://github.com/martin-majlis/Wikipedia-API/> [↑](#footnote-ref-8)
8. https://github.com/martin-majlis/Wikipedia-API/ [↑](#footnote-ref-9)
9. https://www.deepaffects.com/apis/ [↑](#footnote-ref-10)