**PLATFORM FOR IMPROVING SEARCHABILITY AND INTERACTIVITY OF RECORDED LECTURES**

Hettiarachchi H. A. I. S.

(IT15146366)

Bachelor of Science (Honors) in Information Technology

Specializing in Software Engineering

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

August 2019

**PLATFORM FOR IMPROVING SEARCHABILITY AND INTERACTIVITY OF RECORDED LECTURES**

Hettiarachchi H. A. I. S.

(IT15146366)

Dissertation submitted in partial fulfilment of the requirements for the Bachelor of Science in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

August 2019

**DECLARATION**

I declare that this my own work and this dissertation does not incorporate without acknowledgment of any material previously submitted for a Degree or Diploma in any other university or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Also, I hereby grant to Sri Lanka Institute of Information Technology, the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

H. A. I. S. Hettiarachchi

IT15146366

The above candidate has carried out research for the bachelor’s degree dissertation under my supervision.

Signature of the supervisor: Date:

Signature of the co-supervisor: Date:

**Abstract**

E-Learning has become commonplace and many leading Universities provide the facility to view pre-recorded lectures online. This approach gives learners the ability to follow lectures without time or location constraints and consume the lectures at their own pace. Despite their advantages, recorded lectures 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 the lecture videos fail to show the connection between the lecture and its supporting material such as lecture slides and questionnaires.

Several platforms exist where videos can be edited to make them more interactive, however this is a time-consuming process. We propose a system which will automatically improve the interactivity and accessibility of recorded lectures in a few clicks. The system will take in raw lecture videos along with supporting material such as lecture slides and code samples. It will then carry out noise removal and optimizing on the raw video footage before matching the slides and code samples to occurrences in the video. Some novel features we plan on introducing are automatic generation and suggestion of questions based on the content and automatic video segmentation according to topics. The main objective of the system is to create a web platform which can add interactivity and accessibility to course material thereby improving learner engagement.

**Acknowledgement**

The work described in this research paper was carried out as part of the final year research project for the subject Comprehensive Design Analysis Project. The completed final project is the result of combining all the hard work of the group members and the encouragement, support and guidance given by many others. Therefore, it is our duty to express our gratitude to all who supported us in this endeavor. We would like to thank Dr. Nuwan Kodagooda and Ms. Kushnara Suriyawansa our supervisor and co-supervisor, for the invaluable advice and guidance they provided. We are also extremely grateful to Mr. Jayantha Amararachchi, Senior Lecturer/ Head-SLIIT Centre for Research who gave and confirmed the permission to carry out this research. We also wish to thank all our colleagues and friends for their support. Finally, we would like to thank all others whose names are not listed particularly but have given their support in many ways and encouraged us to make this project a success.

**Table of Contents**

[1. INTRODUCTION 1](#_Toc16543083)

[1.1. Background and Literature Survey 1](#_Toc16543084)

[1.1.1. Literature Survey on Automatic Question Generation 2](#_Toc16543085)

[1.2. Research Gap 3](#_Toc16543086)

[1.3. Research Problem 4](#_Toc16543087)

[1.4. Research Objectives 4](#_Toc16543088)

[2. METHODOLOGY 6](#_Toc16543089)

[2.1. Automatic Question Generation 7](#_Toc16543090)

[2.1.1. Extracting Input Text 7](#_Toc16543093)

[2.1.2. Question Generation 9](#_Toc16543094)

[2.1.3. Question Reviewing and Presenting to the User 12](#_Toc16543095)

[2.2. Web Application 13](#_Toc16543096)

[2.3. Commercialization Aspect of the Product 14](#_Toc16543097)

[2.2. Testing and implementation 15](#_Toc16543098)

[3. RESULTS AND DISCUSSION 16](#_Toc16543099)

[3.1. Results 16](#_Toc16543100)

[3.2. Discussion 16](#_Toc16543101)

[4. CONCLUSION 18](#_Toc16543102)

[5. REFERENCES 19](#_Toc16543103)

**List of Tables**

[Table 1.1: Comparison of existing products 3](#_Toc16543104)

[Table 3.1: Qualitative testing results 16](#_Toc16543105)

**List of Figures**

[Figure 2.1: High-level architecture diagram 6](file:///C:\Users\USER\Documents\SLIIT\CDAP\Documents\Final_Report_IT15146366.docx#_Toc16543106)

[Figure 2.2: High-level system diagram of automatic question generation process 7](file:///C:\Users\USER\Documents\SLIIT\CDAP\Documents\Final_Report_IT15146366.docx#_Toc16543107)

[Figure 2.3: Question creation process by transforming sentences 11](file:///C:\Users\USER\Documents\SLIIT\CDAP\Documents\Final_Report_IT15146366.docx#_Toc16543108)

[Figure 2.4: User interface for question review 13](#_Toc16543109)

[Figure 2.5: Video playback user interface 14](#_Toc16543110)

**List of Abbreviations**

API Application Programming Interface  
RAM Random Access Memory  
MOOC Massive Open Online Course  
NLP Natural Language Processing  
SaaS Software as a Service  
URL Unified Resource Location  
AWS Amazon Web Services  
GHz Gigahertz  
BSc Bachelor of Science

# **INTRODUCTION**

## **Background and Literature Survey**

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 common to see universities and higher education institutes adopting some form of e-learning to assist their students. Many institutes use their own customized version of a Learning Management System (LMS) to provide online course material.

Online education is beneficial for both students and teachers 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].

Usually, many LMSs enable lecturers to upload course material such as tutorials, lab sheets, lecture slides and recorded lecture videos. Whilst videos are more effective because they address both 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 there are many video-creation and editing platforms which allow users to create interactive course material, these methods are time consuming as they require the lecturer to spend a lot of time editing and making the video interactive.

At the time of writing this proposal, to the best of our knowledge, there is no system which automatically identifies the relationship between different types of course material and enables the creation of interactive courses in a few steps. Hence, our main focus will be on improving two aspects of course material: accessibility and interactivity. Our goal is to develop an intelligent system capable of improving the interactivity and learner engagement of course material in just a few clicks.

### **Literature Survey on Automatic Question Generation**

The main goal of any lecture is that the learner should achieve knowledge on a specific subject or area. This specific knowledge gained from a lecture is known as its learning outcome. One of the primary and most effective ways to evaluate whether the learner has achieved these learning outcomes is quizzes. However, formulation of questions for quizzes can be a time-consuming task.

Research has been carried out on ways to automatically generate questions. Shah et al suggest a method of generating Multiple Choice Questions (MCQs) for an input text passage with the aid of a one-time trained knowledge base developed using Wikipedia articles [6]. During the execution of the system words in sentences are mapped to a predefined dictionary and an Inverse Document Frequency score is used to choose the word that serves as a blank. Distractors are generated by the paradigmatic discovery on a self-made corpus and dictionary.

TEDQuiz [7] 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 [8] and creating distractors by less important sentences. The second type is detailed questions which use Heilman and Smith’s work [9, 10] 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 is seen in [11]. The process is carried out by identifying discourse boundaries from the lecture and retrieving Wikipedia text segments related to identified discourse boundaries for further well-formed and formal discourse to generate question items: MCQs generated using Heilman’s work [12] and distractors generated on an ontology-based strategy using Wikipedia category taxonomy as a replacement for the ontology.

Apart from this, automatic question generation was carried out using ontology-based strategies. SeMCQ [13] is a Protégé plugin created for automatic ontology-driven multiple question test generation. OntoQue [14] 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 [15].

## **Research Gap**

Even though there are several products already available with similar objectives, they mostly focus on the use of manual processes that involve human intervention to make the content more interactive and increase the searchability. Our proposed solution aims to reduce the amount of human interaction needed for this process by introducing a platform which will analyze and augment the content automatically. Table 1.1: Comparison of existing products is a comparison of the proposed system with existing systems in the market.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Features | LearnWorlds | Echo360 | TechSmith Relay | Our Solution |
| Matching lines in code samples to occurrence in recorded lectures | ✗ | ✗ | ✗ | ✔ |
| Automated segmentation of lecture video into topic units | ✗ | ✗ | ✗ | ✔ |
| Matching slides with the lecture video | ✗ | ✔ | ✗ | ✔ |
| Automatic question generation | ✗ | ✗ | ✗ | ✔ |

Table 1.1: Comparison of existing products

## **Research Problem**

Nowadays, video lectures have become increasingly popular and many education institutes use Learning Management Systems that support video content. Whilst video lectures have benefits such as giving learners remote access to lectures, there are a few drawbacks such as poor searchability through the video and less interaction with the learner.

To overcome these drawbacks many lecture platforms have introduced tools such as web-based video editors that allow lecturers to add captions, divide the video into discussed topics, link lecture slides and embed questions into the video. However, these tools require human intervention which is time-consuming. When considering the domain of computer science, none of the available platforms provide a tool to map programming language code segments with their occurrences in the video.

Because there is a need for video lectures to be more interactive and searchable, but that a significant amount of time is taken by lecture creators to make them so, we conclude that a platform which will analyze and augment the content automatically will be useful.

## **Research Objectives**

The proposed system is a research study to improve the method of delivery for video lectures and increase the engagement of the learner. The objectives of the research are as follows

**General Objective**

To develop an automated platform which provides a quick and efficient way for lecture creators to deliver video lectures which are more interactive and have increased searchability.

**Specific Objectives**

* Automatically tag specific points in the video which correspond to a given lecture slide or line of code in a given supporting code sample.
* Automatically identify main topic transitions of a given lecture video and segment the video according to the identified positions.
* Automatically generate question items related to the lecture video and present to end user.
* Automatically remove unnecessary background noises and enhance the quality of the audio.

# **METHODOLOGY**

High level architecture diagram of the platform is shown in Figure 2.1: High-level architecture diagram Lecturer must create an account as a lecturer and sign in to the platform to use lecture uploading and managing tools. To initiate the processing of video, the lecturer should upload a lecture recorded video and other materials such as code sample files, lecture slides to the platform. When the frontend uploading process is finished lecturer will be notified the processing is began. In the backend of the platform, video file and other supporting materials will be sent to an AWS S3[[1]](#footnote-1) bucket in same time creating an instance of metadata to the uploaded lecturer in a non-relational MongoDB database hosted in cloud. Once the S3 bucket stored the files AWS Simple Notification Service[[2]](#footnote-2) will send a notification to the backend with URLs of the files to start processes of each component in the platform.

A screenshot of a cell phone

Description automatically generated

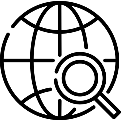
Figure 2.1: High-level architecture diagram

## **Automatic Question Generation**

As mentioned in the previous section automatic question generation for video lecture is an isolated component from other components in the platform. Question generation process can divide into 3 major sub components. These sub components are explained in section wise in this section. High level system diagram of the automatic question generation component is shown in the Figure 2.2.

Figure 2.2: High-level system diagram of automatic question generation process

Main subject of the lecture



Web scraper to search and extract web content

Question generation

Present questions to end user

Search results and web content



### **Extracting Input Text**

Question generation process is mainly carried out using a rule-based methodology using text in paragraphs or sentences that contain factual statements as the input to the system. The first and the obvious approach for selecting the input text was the transcript of the video. However, this approach had to be discarded to several reason mentioned below since these recorded video lectures are done in a traditional classroom environment which can be longer than 1 hour to 2 hours.

* During a longer lecture, lecturer could talk about non-subject related content to take the focus of the students.
* Lectures carried out in a verbal manner that facts are scattered into several sentences.
* Lectures can be noisy.

To address the above issues, source of input text for the system had to chosen from internet rather than using the transcript of videos. To extract information or text from internet a search key phrase correlated with the lecture that could identify as the main discussion subject of the lecture needed to feed as an input to the system. For this key phrase extraction, systematic way of key phrases extraction had to be discarded since an error of this step will affect the outcome of the question generation process (error in a key phrase extraction could result in generating a set of question items that does not have any relation to the lecture video). To have 100% accuracy of this step, a basic human user input was taken as the key phrase or subject discussed in lecture.

Extracting content on internet is also carried out in two approaches. First approach is using a general web search from the key phrase to obtain articles or web pages.

* Once the key phrase is obtained, question generation process will systematically carry out a web search for the given key phrase. For this step, a search engine scraper implemented using GoogleScraper[[3]](#footnote-3) that imitate a web search query for the key phrase obtained and return the URLs of first 10 search results on a given search engine. Search engine of the scraper is set as Bing[[4]](#footnote-4) since it is the least problematic search engine that can be used for scraping. These extracted URLs are sent to another implementation of general web scraper implemented using Scrapy[[5]](#footnote-5) which will crawl through given URLs and extract text elements from web pages. However, using a general web search results to extract content has a possibility of copyright issues.
* To avoid this problem second approach is used where only the Wikipedia[[6]](#footnote-6) articles are used as a source to extract content. This method is also carried out in a similar manner explained in above section. Main difference in this approach is search engine query is specified to search using Wikipedia articles and content extraction process is carried out using Wikipedia-API[[7]](#footnote-7) instead of Scrapy implementation.

### **Question Generation**

Question generation process for the platform is heavily based on the work of Michael Heilman’s Automatic Factual Question Generation from Text [9] which is considered by many researchers as one of best works regarding the automatic question generation using text and rule-based approach. In the system introduced by Heilman, question generation is carried out in 3 stages.

#### **Stage 1 – NLP Transformation on Source Sentences**

In this stage extracted text from the web undergoes series of processes to obtain simplified factual statements from complex sentences that can be used to create question items.

For the simplified factual statement extraction, algorithm introduced in Michael Heilman and Noah A. Smith’s Extraction Simplified Statements for Factual Question Generation [16] is used. The extraction is based on semantic entailment (by removing discourse markers and adjunct modifiers and splitting conjunctions) and extraction by presupposition as explained in the paper. A high-level pseudocode of the algorithm is mentioned below which works on Penn Treebank style [17] phrase structure trees. To obtain the tree structure Stanford Parser [18][[8]](#footnote-8) is used.

**Algorithm 1 extractSimplifiedSentneces(t)**

*Tresults*← ∅

*Textracted*← {t} ∪ **extract** new sentences trees from *t* following: non-restrictive appositives; non-restrictive relative clauses; subordinate clauses with a subject and finite verb; and participial phrases that modify noun phrases, verb phrases, or clauses.

**for all** *t ∈ Textracted* **do**

*Textracted ←* *Tresults* ∪ **extractHelper(***t***)**

**end for**

**return** *Tresults*

**Algorithm 2 extractHelper(t)**

*Tresults* = ∅

**move** any leading prepositional phrases and questions in *t* to be the last children of the main verb phrase.

**remove** the following from *t:* noun modifiers offset by commas (non-restrictive appositives, non-restrictive relative clauses, parenthetical phrases, participial phrases), verb modifiers offset by commas (subordinate clauses, participial phrases, prepositional phrases), leading modifiers of the main clause (nodes that precede the subject).

**if** *t* has S, SBAR, or VP nodes conjoined with a conjunction c ∉ { or, nor } **then**

*Tconjuncts ←* **extract** new sentence trees for each conjunct in the leftmost, topmost set of conjoined S, SBAR, or VP nodes in *t.*

**for all** *tconjunct* ∈ *Tconjuncts* **do**

*Tresults ← Tresults* ∪ **extractHelper(***tconjunct***)**

**end for**

**else if** *t* has a subject and finite main verb **then**

*Tresults ← Tresults* ∪ {t}

**end if**

**return** *Tresults*

#### **Stage 2 – Question Creation**

In this stage, the factual statements extracted using above algorithm is used as input to create possible question items. Creation of question is based on rules of wh-movement comes in linguistics. In Heilman’s work of question creation, he has introduced 6 steps that have to applied to create a question from an extracted factual statement which could contain noun phrase, prepositional phrase or subordinate clause as the answer phrase. Figure 2.3 illustrate the process of transforming a sentence to create a question using these 6 steps.

These 6 steps are explained briefly under this section. A deep explanation of these steps can be found in the thesis publication of Automatic Factual Question Generation from Text.

Figure 2.3: Question creation process by transforming sentences

Decomposition of main verb

Input sentence

Mark Unmovable Phrases

Generate Possible Question Phrases

Subject-Auxiliary Inversion

Remove Answer, Insert Question Phrase

Post Processing

Question

**Marking Unmovable Phrases**

Mark the phrases of input tree as unmovable on the constraints of wh-movement. Tregex[[9]](#footnote-9), which is a Java program for identifying patterns in trees is used in this step.

**Generate Possible Question Phrases**

Input sentences are annotated with an entity types that each word token is labeled (such as PERSON, ORGANIZATION, LOCATION, etc.) using a Java implementation of Supersense Tagger described by Massimiliano Ciaramita and Yasemin Altun [19].

**Decomposition of the Main Verb**

Convert question phrases that does not contain an auxiliary verb into a form of *do* and base form of the main verb. This step is skipped if an auxiliary verb is already available in the sentence.

**Subject-Auxiliary Inversion**

Move auxiliary verb in sentence to a position in front the subject of the main-clause.

**Removing Answer Phrase and Inserting Question Phrase**

Removes the selected answer phrase from the sentence and insert question phrase generated using the selected answer phrase.

**Post-processing**

Additional post-processing to properly format question such as adding question marks to the end of question phrases, etc.

#### **Stage 3 – Question Ranking**

In addition to question creation process Heilman’s paper introduce a question ranking component to select the most suitable questions. This stage is skipped in this implementation of platform.

### **Question Reviewing and Presenting to the User**

Question items generated as described in the previous section will be saved to the metadata instance created initially by uploading the lecture video and other supporting materials. After the completion of all processes related the video lecture, lecturer will be notified as *video lecture processing is finished*. These processed videos will be shown to the lecturer in a separate section in the platform as *videos up for publish* where he or she can review the automatically generated components such as questions.

For the question generation component, questions will be shown with answers to the lecturer as a list. From this list questions can be removed or edited if they are invalid or does not match with the content of lecture. Answer of the question also can be edited and add distractors according to the lecturer’s will. A sample user interface of this process is shown in Figure 2.4.

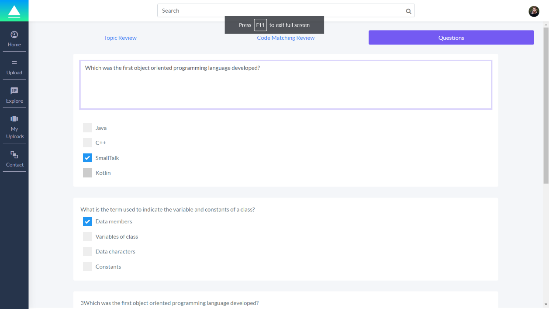


Figure 2.4: User interface for question review

## **Web Application**

The frontend client of the platform is implemented as a web application developed using Angular 7 framework. Figure 2.5 represent the user interface of video player page which is the final representation of output of the platform to the user.

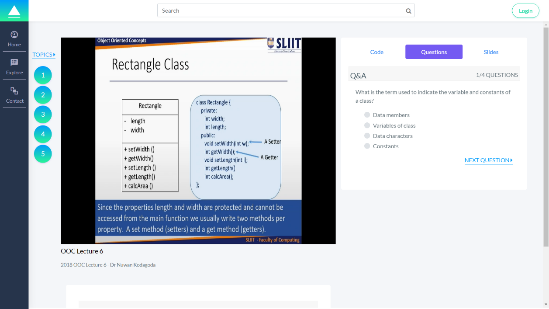


Figure 2.5: Video playback user interface

## **Commercialization Aspect of the Product**

As shown in the literature survey and research gap, none of the available commercial products provide automated unique features introduced in our platform. Considering the unique features provided in a web-based nature, our platform has a unique business potential in eLearning field. Few of the business product approaches that is available for the platform is mentioned below.

* Standalone platform for video lectures provided as SaaS product, where customers can get license and use the features.
* Free platform for video lectures with payable content that viewers must pay purchase or subscribe to view premium content.
* Integration to existing MOOCs and LMSs as a plugin.

## **Testing and implementation**

Question generation component is implemented as a separate web service using RESTful architectural pattern to increase the performance on the overall system and question generation component itself. Web scraping and text extraction processes are implemented using Python 3 and question generation process is implemented as a Java spring boot API.

# **RESULTS AND DISCUSSION**

## **Results**

The results and the evaluation of the question generation component is discussed in this section. A performance testing for the system was carried out using a personal computer with Intel Core i5 1.7GHz mobile processor and 8GB of RAM which use Windows as the operating system. Wikipedia article on C++ classes[[10]](#footnote-10) was used as the input text source for the system. According the Count Wordsworth[[11]](#footnote-11) website, Wikipedia article contains 186 sentences and 3851 words.

Question generation as an isolated system took ***21 minutes*** to parse, extract and generate question items from the given Wikipedia article. From the given 186 sentences system was able to create ***325 questions*** which doubled the question per sentence count.

Qualitative testing for the generated question items was carried out by providing a sample of 100 questions generated to 3 students studies in final year BSc. in Information Technology. The results of the qualitative testing are shown in Table 3.1.

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

Table 3.1: Qualitative testing results

## **Discussion**

The performance results of the question generation component in can be vary due to several reasons as mentioned below.

* Complexity of the input text source
  + Providing complex input text source to the system can increase the time taken to simplify and extract factual statements from it which could results a delay in overall question generation process.
  + However, selecting a less complex input text source is not under control of the system in the current implementation of the platform.
* Performance of the computer
  + Some components of the system are high memory intensive (such as parsing the input text source). Increasing the performance of the computer can result in an increase of the system.
  + In a hosted environment system could work faster. However, server costs can be high.

Quality of the question items generated mainly based on the quality of input sentences. Using web scraped content on different subjects can vary this quality in an uncontrollable manner. To increase the quality of questions in future works, providing the input text source as supporting materials to the platform (reference textbooks, etc.) can increase the quality of the question items. Implementing a ranking system as mentioned in Heilman’s work could also increase the quality of output.

MCQ distractors are not generated from the question generation component. In future works, an implementation of automatic distractor generation will be an advantage for the lecture creators.

As discussed in section 2.1.1 usage scraped web content to generate questions can result in a violation of terms of use in some websites. Hence the decision to use only Wikipedia articles for scraping process is made.

# **CONCLUSION**

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 and interactivity of lecture videos. Automatic question generation feature can consider as a value-added feature implemented to increase the interactivity of lecture videos on top of other core features in platform that increase searchability and interactivity.

In future works, question generation can be further improved as discussed in section 3.2. with features like automatic distractor generation and improving the accuracy and quality of question items by adopting different approaches for the question generation mentioned in the literature.

# **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] | 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. |
| [7] | 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. |
| [8] | 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. |
| [9] | M. Heilman, "Automatic Factual Question Generation from Text," Doctoral thesis, Carnegie Mellon University, Pittsburgh, 2011. |
| [10] | 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. |
| [11] | 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. |
| [12] | 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. |
| [13] | 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. |
| [14] | 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. |
| [15] | 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. |
| [16] | M. Heilman and N. A. Smith, "Extracting Simplified Statements for Factual Question Generation," 2010. |
| [17] | M. . P. Marcus, M. A. Marcinkiewicz and B. Santorini, "Building a large annotated corpus of English: the penn treebank," *Computational Linguistics - Special issue on using large corpora: II,* vol. 19, no. 2, pp. 313-330, June 1993. |
| [18] | D. Klein and C. D. Manning, "Fast Exact Inference with a Factored Model for Natural Language Parsing," in *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. |
| [19] | M. Ciaramita and Y. Altun, "Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger," in *EMNLP '06 Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, Sydney, 2006. |

1. <https://aws.amazon.com/s3/> [↑](#footnote-ref-1)
2. <https://aws.amazon.com/sns/> [↑](#footnote-ref-2)
3. <https://github.com/NikolaiT/GoogleScraper> [↑](#footnote-ref-3)
4. Bing.com [↑](#footnote-ref-4)
5. <https://scrapy.org/> [↑](#footnote-ref-5)
6. <https://www.wikipedia.org/> [↑](#footnote-ref-6)
7. <https://github.com/martin-majlis/Wikipedia-API/> [↑](#footnote-ref-7)
8. <https://nlp.stanford.edu/software/lex-parser.shtml> [↑](#footnote-ref-8)
9. <https://nlp.stanford.edu/software/tregex.shtml> [↑](#footnote-ref-9)
10. <https://en.wikipedia.org/wiki/C%2B%2B_classes> [↑](#footnote-ref-10)
11. <http://countwordsworth.com/> [↑](#footnote-ref-11)