Automatic Tutorial Generation from Input Text File using Natural Language Processing

N. S. Kumar
Computer Science
PES University
Bangalore, India, 560085
nskumar@pes.edu

Shubha Manohar
Computer Science
PES University
Bangalore, India, 560085
shubhamhegde6@gmail.com

Shrutiya M
Computer Science
PES University
Bangalore, India, 560085
shrutiya03@gmail.com

Tejaswini Amaresh
Computer Science
PES University
Bangalore, India, 560085
amaresh.tejaswini@gmail.com

Kritika Kapoor
Computer Science
PES University
Bangalore, India, 560085
kapoorkritika66@gmail.com

***Abstract*—In the present scenario, it has become a major necessity to create easy-to-access, easily-understandable and less time-consuming resources/tutorials. Our application proposes to create a full-fledged tutorial, provided an input in the form of a text file (txt/pdf). We have adopted a generator-subscriber model with role specific duties. The generator is provided with the option to generate tutorials and the subscriber can view/access the tutorials. The input provided by the generator is analysed to identify headings, subheadings and paragraphs as a hierarchy. Additionally, the uploaded material is summarised and a concise presentation with audio voiceover, to enhance comprehension of the user, is presented for quick learning. Furthermore, to ensure user interaction, assessments in the form of multiple-choice questions are dynamically created from the uploaded material. The scores of assessments are instantly provided and the subscriber's performance is recorded to show progress. The application is further extended to handle multimedia input in the form of images. In order to increase the usability of the application and make it accessible to a wider population , language agnosticism has been attempted. The features of language agnosticism are currently concentrated on Indian languages (Kannada and Hindi). A complete user-friendly web interface with a catalogue of automatically generated tutorials is provided for easy use of all the features and hassle-free learning.**

***Index Terms*—Tutorial Generation, NLP, Text Summarization, Q/A generation, Multimedia Handler, PDF Parser**

I. Introduction

A tutorial system is essential for a wide variety of user groups majorly involving teachers-students, mentors-trainees and many others. The documents and information available to the users is in huge volume, thus making it impossible to create a handwritten summary or an easy to interpret tutorial within a short time period.

The application built enables automatic generation of tutorials given an input text file, overcoming the problem of information overload and manual work that needs to be done otherwise. This helps the end user to easily understand the basic concepts of the topic clearly. End users can actively interact with the application, navigating from one topic to another, through a well-built web interface unlike the traditional trying to listen and grasp concepts in an otherwise time-consuming verbal comprehension.

The web application supports two roles for the end users. The 'generators' upload the documents onto the application and are responsible for tutorial generation. The 'subscribers' access the tutorials created and attempt the assessments based on topics read. The tutorial created can be accessed on the website with easy navigation tools or downloaded as a concise presentation with a voice over. Attempts have been made to create a concise presentation from a text file which concentrates on summarisation of the text through NLP techniques. This was followed by a subtopic generation phase to create a well-structured tutorial. In order to generate the assessments for every subtopic of the tutorial, QuestGen.ai module was used to provide most appropriate objective type questions. The application was also scaled to handle multimedia content to make the tutorial more presentable. Language agnosticism was attempted by extending the application to handle kannada and hindi text files. The application built proves to be a user-friendly platform for generators to make their content reach a large number of users and the subscribers to master new skills by referring to a well-structured easy to understand tutorial.

II. Literature Survey

Our project involves several components such as Text summarisation, PPT generation, Video tutorial generation, inherent Q/A system and progress tracker. There have been separate attempts at building some of the components in various manners. Several papers have been published pertaining to this.

A. Tutorial generation from related materials

This section details the survey of tutorial generation from related materials such as diagrams and learning materials.

1) Automatic Generation of Tutorial Systems from Development Specification [1]: In this paper, automatic tutorial generation was targeted at software as a step-by-step demo on how to use a particular software. The input to this system was use case diagrams, sequence diagrams and test data provided by developers and the generated tutorial was directly woven into the software. The system architecture consisted of five main steps - Adding extra description to sequence diagrams, Extracting function name from use case diagrams, Extracting operation information from sequence diagrams, Generating tutorial system automatically and Weaving tutorial system into target software automatically. It had an interactive GUI where features like showing Software Functions and Operations, showing examples of Software Usage, demonstrating Input Texts etc were highlighted for easy navigation of the software for the users.

This was an intuitive method at tutorial generation, but it has a scope limiting only to software. Furthermore, the sequence diagrams have to be annotated by the developers which limits the degree of automation of tutorial generation.

2) Automatic Generation of Contents Models for Digital Learning Materials [2]: In this paper, a method to automatically generate a contents model by analysing learning materials with the aim of supporting the construction of knowledge structures is proposed. Learning materials are text-mined and analysed to generate a co-occurrence graph. The co-occurrence graph is then modelled to generate an organized content graph. To reduce the complexity of the graph to the relevant key points, an optimal spanning tree that only selects the strongest relations between nodes is searched. The relations between nodes are expressed through weighted edges. The limitations include that the generation of the graph is dependent on the input text and therefore may create off topic nodes from examples. The detailed level of the output contents-model requires supervision from the domain expert.

B. Presentation Generation from Text files

This section details the survey of PPT generation from a text file which also include attempts at text summarisation.

1) Effective Classroom Presentation Generation Using Text Summarization [3]: In this paper, classroom presentation generation was aimed at using text summarization. Single text file taken as input and pre-processed. Sentences were scored using Cue-Phrase Method, Word frequencies, Title Method and Location Methods and the highest rated sentences formed the summary. Summary was presented as a PPT with 3 sentences per slide. More Importance was given to summarising the entire text rather than the PPT creation and the drawbacks being that it does not correctly divide on sub-topics as such.

2) A Semantic Machine Learning Approach to Automatic PPT Generation [4]: In this paper, Machine learning was employed to automate PPT generation. Input of multiple documents of type pdf, text and MS word was taken and a PPT was generated. First, Input was concatenated into a single string and pre-processed. Then, Feature Extraction (Title Feature, Sentence Length, proper Noun, Numerical Data, Term Weight) was done and values were stored. The tree containing the sentences was traversed in preorder and the sentences were classified using Random Forest. The result generated was a concise presentation of slides containing indices like Abstract, Motivation, Problem Statement, Objectives, introduction, Summary and Key words only. The resulting presentation contains pre-decided indices which fixes the structure of the PPT. Contiguous sub topic wise PPT was not formed, which is the main aim of a tutorial.

3) Enhancing Automatic PPT Generation Technique through NLP for Textual Data [5]: In this paper, NLP was the main idea behind PPT generation. Input of any format (PDF/DOC) file was given and output was a PPT. Input was concatenated into single string and preprocessed. Feature Extraction (Title Feature, Sentence Length, proper Noun, Numerical Data, Term Weight) was done and values stored. Data was then passed to the fuzzifier, which grouped the sentence into categories of importance according to the numerical score calculated. Fuzzy IF THEN Rules were then performed, on fluffy sets portrayed by proper enrolment capacities. It then utilized distinctive elements to make PPT on the basis of the analysis performed. MRR was used as a measuring strategy. All the text isn't very conveniently segregated into headings. Key-expressions and sentence mapping is not performed per slide.

4) SlidesGen: Automatic Generation of Presentation Slides for a Technical Paper Using Summarization [6]: In this paper, Input was documents in LATEX form. These documents were converted to XML format which was parsed and information was extracted. Sections were categorized and then key phrases were extracted for some sections of the paper following which details were saved in a file (configuration file). Extraction of key phrases was used in summarization of the sections. To generate slides QueSTS (query specific extractive Summarizer) was used. Input was text which is in the form of an Integrated Graph (IG) i.e. nodes representing sentences, linked to each other via weighted edges if the sentences were similar. Based on a particular query, nodes were generated as output, which bear greatest relevance to the query, calculated using the node weight which was further used to generate a coherent and readable summary. The XML file and the configuration file were given as input to the Slides Generator component and it generated slides for each section and subsection. Natural language processing techniques are not used in this paper to compress the extracted sentences and to identify appropriate indentation structure for them. PPT produced as output is less appealing, containing only 4 slides only. The domain is restricted as it is research-paper specific.

C. Question Answer Generation

This section details the survey of the current state-of-the-art methods for question answer generation.

1) Enhanced Automatic Question Creator – EAQC: Concept, Development and Evaluation of an Automatic Test Item Creation Tool to Foster Modern e-Education [7]: In this paper, most important concepts out of textual learning content were extracted and single choice, multiple-choice, completion exercises and open-ended questions on the basis of these concepts were created. The system had three main modules:

a) The Pre-processing module: Files of several formats were text cleaned, language detected and was transformed involving internal XML.

b) The Concept Extraction module: Ran term weighting extraction, statistical, semantic and structural analysis of the most suitable phrases was achieved.

c) The Assessment Creation module: The suitable sentence for each phrase, also identified distractors and antonyms, created question items and reference answers were determined. This approach is a Question Answer generator, and does not embody tutorial generation as such. This could be used to guarantee interaction from the user, as an extra feature of our project.

D. Multimedia Handling

This section details the survey of multimedia generation from text.

1) Video Generation from Text [7]: In this paper, Video was generated using a hybrid framework employing a VAE and a GAN. The static features, called "gist" are used to sketch text-conditioned background color and object layout structure (Universal features). Dynamic features are considered by transforming input text into an image filter (Details). However, the video generated is generated for short sentences. The video generated is short with not much descriptive differences between the frames.

2) TiVGAN: Text to Image to Video Generation With Step-by-Step Evolutionary Generator [7]: This paper uses Generative Adversarial Network. Rather than first finding a mapping function between the text and all video frames, the network is trained with respect to one image and is gradually extended to longer frames. It has two stages: Text-to-Image Generation and Evolutionary Generation. In each step m, 2m images can be obtained by iterative operations of recurrent unit, R and generator, G learned in the first step of test-to-image generation. At each step m, mth step-discriminator DSm is newly added to discriminate the sequence of 2m frames along with the image discriminator D1. Several experiments were conducted on three diverse datasets: KTH Action, MUG, and Kinetics. The video generated is generated for short sentences. The video generated is short with not much descriptive differences between the frames.

E. Language Agnosticism

This section details the survey of how language agnosticism was achieved.

1) A Study on Abstractive Summarization Techniques in Indian Languages [7]: This paper provides an overview of abstractive text summarization for Indian languages. Abstractive summarization is predominantly classified into two types namely the structure based and semantic based approaches. The structure based approach involves summary generation in a predefined structure without losing any meaning. The semantic based approach involves a document input, a semantic representation and a natural language generation phase to arrive at the output. One of the approaches for abstractive summarization involves getting the extractive summary which is further used to get the important concepts from the source text. The preprocessing step consists of data chunking, part of speech tagging, word sense disambiguation, etc. The 4 approaches described in this paper are:

- Summarizer extracts key phrases and post processing through refinement and rephrasing results in the abstractive summary.
- Consists of a preprocessing module ,categorization module and a final post processing module to extract the key phrases and further mapping, resulting in the summary.
- Rich semantic graph (RSG) is constructed and topics are ranked based on relevance.
- The linguistic triples were obtained in the document and an RSG is used to obtain the abstractive summary.

III. Product Survey

There are some already existing products that are similar to some components of our project.

A. Text to Video Products

This section details the survey of the current products for creating videos from text.

1) Raw Shorts [8]: Created Animated Videos from Text using AI within seconds. It is more of an editor rather than a completely automated video generator. Presented a storyboard to populate. Used AI to recommend short images/videos for each storyboard. The recommendations were based on the keyword identified which wasn't very accurate. Provided audio for the text with the availability of several languages. However, it is more of an editor and no Indian languages are available.

2) Viomatic [9]: Created Video from content. It is more of an editor. It summarised the article and populated the slides and also gave an option to add images. Provided audio for the text with the availability of several languages. However, no Indian regional languages are available. Also, it was not instant (upwards of 5 minutes depending on the content). Maximum length of video possible was 10 minutes (paid). Free version allows a maximum of 3 minutes with a maximum of 12 slides.

3) Lumen5 [10]: This tool was similar to Raw ShortsRaw Shorts: Online Video Maker — Video Creation Software. Here, the background image was generated based on relevance. However, it is more of an editor and no Indian languages are available.

B. Text to PPT Products

This section details the survey of the current products for creating presentations from text.

1) Google Slides [11]: This tool populated text based on hierarchy into the different slides of a PPT. However, it offers just direct conversion of text to PPT with no summarisation or Semantic analysis done. Also, addition of images, audio, etc are not provided automatically.

2) Convertio [12]: It took a .txt file input and provided a .ppt file output. However, no summarisation or Semantic analysis is done. Automatic Multimedia inclusion is not supported.

C. Assessment Generation Products

This section details the survey of the current products for generating questions and assessments.

1) Typeform [13]: This tool created online tests. It provides a UI to copy paste questions into a template on sharing, the student could answer the tests. A timer was provided. Post finishing the test, the results of the test were instantly generated to the user.

2) Smart Question Generation using Natural language processing [14]: Input can be provided in the form of direct text, text file or a test link. The best/important sentences which could be converted into questions were selected. These sentences were analysed and the most sensible question from them were provided.

3) Questgen AI [15]: It was an open source NLP library focused on developing easy to use Question generation algorithms. It used AI leveraging state-of-the-art transformer models like T5, BERT and OpenAI GPT-2 etc. It was able to create MCQs, Boolean Questions (Yes/No) and General FAQs.

D. Language Agnosticism Products

This section details the survey of the current products for Language agnosticism.

1) INLTK [13]: It is a Natural language toolkit for around 15 Indian languages. Compatible tasks include tokenization, getting embedded vectors, predict next n words, language identification, removing foreign language sentences and so on. Can be used for transfer learning and paraphrasing.

2) Indic NLP Library [13]: It is used for advanced text processing tasks for Indian languages. Functionalities provided by this include Text Normalization, Script Information, Tokenization, Word Segmentation, Script Conversion, Romanization Indicization, Transliteration and Translation

3) Hindi Text Summarisation [13]: Summarizes the text document using numerical methods.

E. Multimedia Handling

1) PyMuPDF [13]: Parses images from the input PDF file successfully.

2) Text2img API [13]: Generates an image from given text.

IV. Methodology

Our project involves creating a Navigable tutorial based on the input text file (PDF/txt) along with automatic generation of several components such as Summary Presentation, inherent Q/A system, Multimedia handling, Language Agnosticism and progress tracker.

A. Input

The generator is authorized to input a text file of PDF/.txt format. He/she will provide the name and category of the tutorial to be generated. On submitting the text file, the entire navigable tutorial with voiceover along with Q/A system and summarised presentation is created. A labelling picture for the tutorial is also automatically generated using web scraping. In addition to this, a quick preview of the text and important sentences containing summary is provided for the Generator.

B. Topic-Subtopic Modelling

The input text is analysed and the headings, sub-topics and paragraphs are extracted. The most used font can be considered the paragraph. The font size generated for text in the input document varied a lot due to size variation. This was then pre-processed to bucket the surrounding font sizes in pre-assigned categories, hence tagging the text with appropriate annotation to indicate post processed font sizes. PyMuPDF is used to identify the paragraphs, different levels of headers (larger than paragraph) and different levels of subscripts (smaller than paragraph). This information is used to create a hierarchy of the headings, sub-topics and content in a recursive tree-based approach. This topic-subtopic hierarchy is later used to populate the navigable tutorial and sidebars in the correct order.

C. Navigable Tutorial

A navigable tutorial interface is provided using React components. All the important navigation tabs are placed on the top making them prominent for easy user. The user can sequentially access the tutorial in a hassle-free manner. Options to navigate to a specific sub-topic is provided too with the help of SideBar Menu. The SideBar Menu component is dynamically populated in a recursive manner using the tree-based Topic-Subtopic Hierarchy. The assessments can be easily accessed and taken by the 'subscriber' with automatic navigation to the next question, once he answers the current question. Advertisement sections are provided for online advertising which the users can navigate to. Subscribers can view the percentage of correct and wrong answers for all the subscribed tutorials in a visually engaging way.

D. Audio Voiceover

For each page in the navigable tutorial, the user can either read the content or listen to the content being read out with an audio voiceover which is provided. The context of the input text is mapped using the Topic-Subtopic hierarchy to provide audio explanation of the content. Implementation of the voiceover was done through the python gTTS module.

E. Assessment Generation

Automatic generation of assessments/MCQs is provided. This question generation module was implemented through an open source module named Questgen.ai. This module uses natural language processing techniques to parse and analyse what kind of questions to be generated on the key sentences. It uses WordNet to efficiently generate hypernyms which act as effective distractors/options for the MCQs generated. The generator can view the generated questions. The subscriber is provided the option to take assessments through the generated questions. These assessments are integrated with the tutorial to ensure user interactivity. After every two subtopics, the corresponding questions are generated from the text and mapped which is rendered as part of the tutorial. The scores/results of the assessments are instantly provided.

F. Presentation Generation

A presentation is also provided for the user that can be downloaded for a quick revision. The first step towards creating a presentation is summarization. To do this, the input is firstly pre-processed to remove stop words. The text is then lemmatized and is passed to the NLTK summarizer. The NLTK summarizer is an extractive summarizer which efficiently fetches the important components of the input text. The summarized text is provided as an input to the PPT module. Automatic generation of concise PPTs is implemented through Python PPTX module. The slides are made effective by presenting the summarised text as concise bullet points not exceeding three per page. The presentation generated preserves the Topic-Subtopic hierarchy providing one slide from the main titles and maintaining the size of the subtitles for better user readability. Audio voiceover is also provided for every slide for better comprehensibility.

G. User Profile

Both the generator and the subscriber have user profiles which serve different purposes. The generator's user profile is dynamically populated with the tutorials created by him/her. The subscriber's profile is automatically populated with his/her progress. It shows the tutorials that the subscriber has finished along with the percentages of right answers and wrong answers in the assessment component of that particular tutorial.

H. Database

SQLAlchemy database was used. The database consists of 5 tables - User, Tutorial, Assessment, UserProgress, Feedback.

- User table is used to store the profile information and the respective roles (Generator/Subscriber) of the user.
- Tutorial table consists of details about the tutorials stored in the database and is used to retrieve created tutorials.
- Assessment table consists of the questions generated for the tutorial, the corresponding answers and the distractors.
- UserProgress table is used to store the scores/results of the user and to keep track of the assessments which are taken up by the user.
- Feedback table is used to get a detailed feedback about the application and to help enhance the user experience of the application.

I. Web Interface

A complete user-friendly web interface with a catalogue of automatically generated tutorials is provided for easy use of all the features and hassle-free learning.

- Frontend : Javascript based ReactJs is used for frontend. Different Components are written for different concerns using Single Responsibility Principle. Styling is done with the help of CSS.
- Backend : Python based Flask is used for backend. Different endpoints are written for different concerns using Single Responsibility Principle which is accessed by the specific frontend components.

J. Multimedia handling

The application can successfully handle multimedia in the input text file. The input PDF is parsed and the images in each page are identified and displayed on the tutorial. To achieve this, all the images are first extracted from the pdf. Then analyzing the topic-subtopic hierarchy, the appropriate image is rendered on the tutorial, under the correct heading to enhance user comprehensibility, thus making the tutorial more interactive rather than just informative.

K. Language agnosticism

Language agnosticism has been attempted by extending the application to handle Indian languages. Currently ,the application supports Kannada and Hindi. NLP techniques and libraries have been utilized to clean , parse, summarize, display a navigable tutorial and generate assessments. The application can further be extended to handle other languages using a similar methodology.

V. Results and Evaluation

The user interface developed was evaluated based on various metrics and are specified in the below figures. The website opens up with the home screen to login/sign-up. On signing in/ logging in as a generator, option to upload input file is provided. Post input upload, the generator can view the navigable tutorial. Automatically generated assessment can be viewed by the generator. The subscriber can view list of existing tutorials and can access any of them. The subscriber is provided periodic assessments to enhance user interaction.

A. User Statistics

To test the application, the application was deployed on a PaaS platform (Heroku), so that it could be freely accessible to anyone with the generated link. Feedback metrics at the end of the tutorial were added to be rated by the users after completion of the tutorial. The website link was sent to an audience of approximately around 20 people, who were asked to take up a tutorial of their choice from a set of six automatically generated tutorials. The average time taken to create a tutorial from the text file was 103 seconds or 1 minute 33 seconds. This was averaged over 6 PDFs of varying sizes with maximum size being 19 pages and minimum size being 2 pages. Feedback scores for UI, MCQ, User friendliness, PPT quality and Tutorial quality were recorded and averaged over to rate the entire application on a whole as depicted in Fig. 1

The average mcq scores of every tutorial were also recorded to quantify the comprehensibility and to record the level of difficulty of the pre-existing tutorials as depicted in Fig. 2.

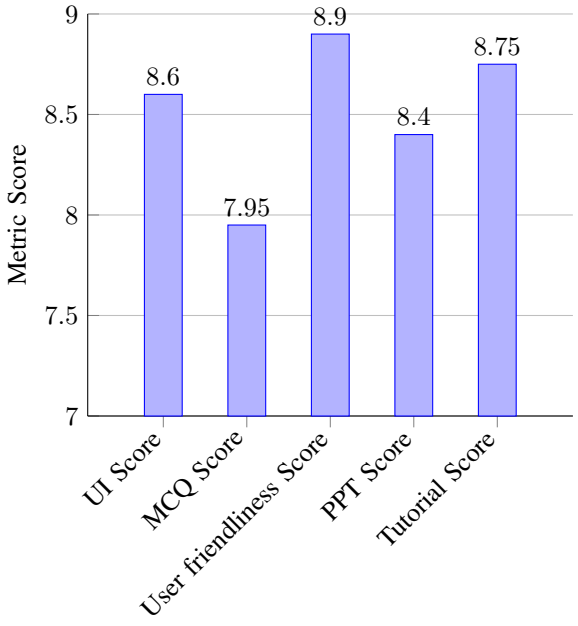

Fig. 1. Average Metric Scores

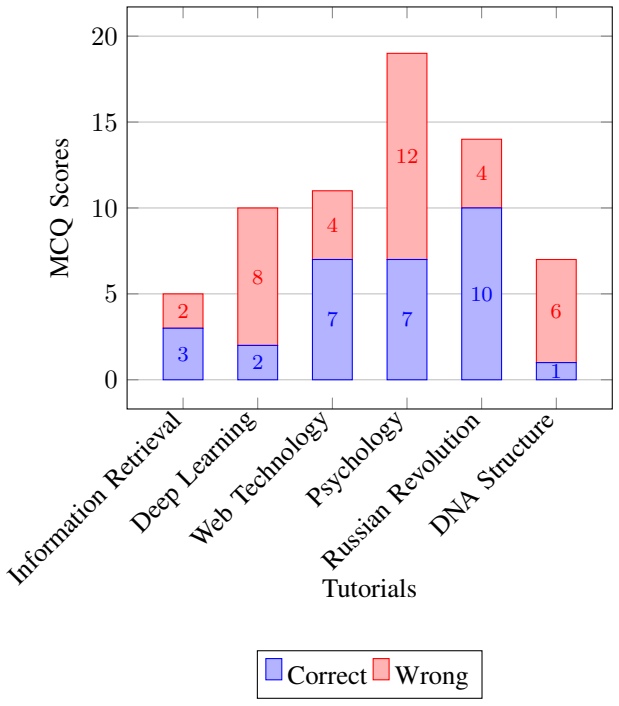

Fig. 2. Average MCQ Scores of Pre-Generated Tutorials

Along with these metrics, a section was provided to the users where they could enter their valuable feedback which could be used to increase the efficiency of the tutorials generated. They contained constructive criticism and some appreciation. A few selected verbatim feedbacks are shown in Table. I.

VI. Conclusion

Successfully built an end-to-end application where the Generator can provide an input text file and a full-fledged tutorial with assessments and progress tracking options is created and an end user, a subscriber can take advantage of the created tutorial to learn the topic.

TABLE I
FEEDBACK COMMENTS

Sl.no	Feedback
1.	Quality of questions can be improved.
2.	It's difficult to see the correct answer for the last question. Please improve the quality of the question and the options.
3.	No one would read the PPT without any visual content in it. UI seems good.
4	A more detailed navigation bar, that is, with more tabs, like progress, tutorial, assessments etc would be better.
5	The generated ppt covers the topic well. The quality of assessments can be better. In particular, the options for the questions need to be improved.

VII. FUTURE WORK

This project will be further extended to include features involving user recommendations. Handling of code segments, tables, mathematical formulae will also be attempted later.

ACKNOWLEDGMENT

We would like to express our gratitude to Prof. N S Kumar, Department of Computer Science and Engineering, PES University, for his continuous guidance, assistance, and encouragement throughout the development of this project.

REFERENCES

[1] Iwata, H., Shirogane, J. and Fukazawa, Y., 2006, March. Automatic generation of tutorial systems from development specification. In International Conference on Fundamental Approaches to Software Engineering (pp. 79-92). Springer, Berlin, Heidelberg.

[2] Flanagan, B., Akcapinar, G.Ö.K.H.A.N., Majumdar, R. and Ogata, H., 2018, November. Automatic generation of contents models for digital learning materials. In 26th International Conference on Computers in Education Main Conference Proceedings (pp. 804-806). Asia-Pacific Society for Computers in Education (APSCE).

[3] Sariki, T.P., Kumar, B. and Ragala, R., 2014. Effective Classroom Presentation Generation Using Text Summarization. IJCTA, ISSN, pp.2229-6093.

[4] Phage, K.R., Rawade, S.S., Thorat, K.S. and Agawane, R.V., A SEMANTIC MACHINE LEARNING APPROACH TO AUTOMATIC PPT GENERATION.

[5] Belote, P., Bidwai, S., Jadhav, S., Kapadnis, P. and Sharma, N., Enhancing Automatic PPT Generation Technique through NLP for Textual Data.

[6] Sravanthi, M., Chowdary, C.R. and Kumar, P.S., 2009, March. Slidesgen: Automatic generation of presentation slides for a technical paper using summarization.

[7] Gutl, C., Lankmayr, K., Weinhofer, J. and Hofler, M., 2011. Enhanced Automatic Question Creator–EAQC: Concept, Development and Evaluation of an Automatic Test Item Creation Tool to Foster Modern e-Education. Electronic Journal of e-Learning, 9(1), pp.23-38.

[8] Raw Shorts [Computer software]. (2009). Retrieved from https://www.rawshorts.com/

[9] Viomatic [Computer software]. (2019). Retrieved from https://www.viomatic.com/en/home/

[10] Lumen5 [Computer software]. (2017). Retrieved from https://lumen5.com/

[11] Google Slides [Computer software]. (2006). Retrieved from https://www.google.com/slides/

[12] Convertio [Computer software]. (2014). Retrieved from https://convertio.co/

[13] Typeform [Computer software]. (2012). Retrieved from https://www.typeform.com/

[14] Aditya, Sarvaiya. (2018). 'Smart Question Generation', *DynamicHub*, 5 April. Available at: http://dynamichub.in/aditya/sqg/ (Accessed: 8 December 2020).

[15] Ramsri Goutham Golla, Questgen.ai, (2019), GitHub repository, https://github.com/ramsrigouthamg/Questgen.ai

[16] Details of Indic NLP libraries.Retrieved from https://www.analyticsvidhya.com/blog/2020/01/3-important-nlp-libraries-indian-languages-python/

[17] Datta, A., Ramabhadran, B., Emond, J., Kannan, A. and Roark, B., 2020, May. Language-agnostic multilingual modeling. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 8239-8243). IEEE.

[18] Sunitha, C., Jaya, A. and Ganesh, A., 2016. A study on abstractive summarization techniques in Indian languages. Procedia Computer Science, 87, pp.25-31.

[19] Li, Y., Min, M., Shen, D., Carlson, D. and Carin, L., 2018, April. Video generation from text. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

[20] Kim, D., Joo, D. and Kim, J., 2020. TiVGAN: Text to Image to Video Generation With Step-by-Step Evolutionary Generator. IEEE Access, 8, pp.153113-153122.

[21] Obeidat, R., Duwairi, R. and Al-Aiad, A., 2019, August. A collaborative recommendation system for online courses recommendations. In 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML) (pp. 49-54). IEEE.

[22] Gulzar, Z., Leema, A.A. and Deepak, G., 2018. Pcrs: Personalized course recommender system based on hybrid approach. Procedia Computer Science, 125, pp.518-524.