Semester Long Internship Report

On

Web app for auto-generation of mind maps

Submitted by

Ved Patwardhan

Under the guidance of

Prof. Kannan Moudgalya

PI of Spoken Tutorial Project

September 7, 2022

Acknowledgement

I, the FOSSEE intern of the Spoken-Tutorial Mind Map Generation Module, is overwhelmed in all humbleness and gratefulness to acknowledge my deep gratitude to all those who have helped me put my ideas to perfection and have assigned tasks, well above the level of simplicity and into something concrete and unique.

I, wholeheartedly thank **Ms. Nancy Varkey, Senior Project Manager, Spoken-Tutorial.org** for having faith in me, selecting me to be a part of this valuable project and for constantly motivating the entire team to do better.

I am very thankful to my mentors **Mr. Saurabh Adhikary** and **Ms. Kirti Ambre** for their valuable suggestions. They were and are always there to show the right track when needed help. With help of their brilliant guidance and encouragement, I was able to complete my tasks properly and was up to the mark in all the tasks assigned. During the process, I got a chance to see the stronger side of my technical and non-technical aspects and also strengthen my concepts.

With Regards.

Ved Patwardhan
(Pune Institute of Computer Technology, Pune)

Contents

1 Abstract

The Spoken Tutorial project is the initiative of the Talk to a teacher activity launched by the Ministry of Human Resources and Development, Government of India. It is being developed by IIT Bombay and provides spoken tutorials on FOSS available in several Indian languages, for the learner to be able to learn in any language he/she is comfortable in.

A mind map is a graphical representation of various concepts related to a topic and the relationships between them. This representation finds its value in human understanding and also in gaining a global view of the document in general. Various scientific studies have affirmed the fact that a mind map provides a faster way of learning concepts.

There is a large number of tutorials available on the website in multiple forms like video and text. In order to learn through a spoken tutorial or a script, learners need to go through them from the start to the end of the content. Further, there is a lack of an option to understand the importance of various keywords in that script. This could be time-consuming for the learner, particularly in cases where the objective is to run-through multiple scripts in a short period of time. A mind map is useful for such purposes.

This project provides the functionality of generating a mind map from a given spoken tutorial script or for multiple scripts simultaneously, to provide learners the opportunity to learn and revise a large number of concepts in a relatively short period of time. A mind map is also a much more attractive way than a script.

2 Web app for auto-generation of mind maps

The web application developed allows the use of links to the Timed Scripts of spoken tutorials to generate a mind map for representing the content in one or more tutorials using natural language processing. The process of generating the mind map involves multiple stages. The first stage involves scraping data from the Timed Script page of a tutorial to filter out unnecessary content. This is followed by cleaning of the text which involves converting the text to lower case, removing stop words and converting the text into a list of sentences.

The list of sentences generated from one or more scripts is then passed to a lemmatiser model that converts all the words into their root forms. The resultant sequences of words are then passed to a BERT-based keyword extraction model, that understands patterns from the text to extract keywords from the text along with their importance with respect to the overall information conveyed through the text.

The word sequences generated after lemmatization are also used to train a Word2Vec model to generate word embeddings for the keywords generated previously. These word embeddings act as numeric representations of the keywords which convey their semantic meaning. Such word embeddings are then used to calculate the degree of similarity between the keywords to establish the edges between them. Finally, a force-directed graph drawing algorithm is used to display the mind map for the given spoken tutorials or scripts.

3 Problems encountered

3.1 Problem 1

The first problem was scraping and cleaning the text from the Timed Script of the given spoken tutorial. This is crucial as the pipeline that follows is sensitive to noise such as case of the input, contributor names, stop words, special characters, digits, etc.

To address this, the Timed Script page was scraped using BeautifulSoup, which is a library that simplifies the process. The text scraped was then filtered out to remove stopwords, digits, special characters and converted to lower case. The pipeline is very sensitive to the contributor names of the scripts, so removing them is crucial. The contributor names are limited, as a result of which they were noted down and filtered out of the text accordingly.

3.2 Problem 2

The second problem was efficient lemmatization of data followed by keyword extraction from it. This is an important problem as there is a tradeoff between the processing time and quality of output generated. Finding the combination of the right balance is important given the number of lemmatization and keyword extraction options available in the community.

The StanfordCoreNLP lemmatizer followed by a BERT-based keyword extractor provided the right balance to achieve acceptable results in a decent amount of time. Although, one of the downsides of this process is that it is heavily dependent on the cleaning part of the pipeline for handling noise.

3.3 Problem 3

The third and problem to be addressed was to extract relationships between the keywords. While doing so, the header of a script was also to be handled separately as the header of a script is supposed to be related to every other keyword in that script. After getting the weights of the relationships, a threshold was needed to create a meaningful graph.

In order to address this problem, the input text consisting of sequences of lemmatized words is used to train a Word2Vec model. Then, using the word embeddings of the keywords as returned by the model are used to get a numeric estimate of the relationship between those keywords using cosine similarity. Finally, a 92 percentile threshold was used to filter out the stronger relationships between keywords to create the graph.

3.4 Problem 4

The final problem to be solved was graph drawing. The graph generated in the previous stages was then to be drawn based on the strength of the relationships. This raised the need of using force-directed graph drawing of the mind map. Along with this, the mind map should also be interactive such that users can click on a particular keyword and the mind map would be redrawn emphasizing the relationships of that keyword with other keywords.

This problem was addressed by using D3.js, a library which provided a force simulation API that was used to draw the mind map considering the strengths of relationships of various keywords. Event listeners were added for all keyword nodes in the mind map. On clicking a particular keyword node, the strength of its relation-

ship with other keywords was multiplied by a suitable factor of 5, and then the edges were again thresholded based on the 92nd percentile. Finally, the mind map was redrawn, highlighting the relationships of that keyword with other keywords.

4 Design Considerations

There are a number of design considerations. Two different web applications are provided which demonstrate the use of the pipeline either as a standalone application that accepts links to spoken tutorial timed scripts or an integration into the spoken tutorial website through a Generate Mind Map button. The pipeline is particularly designed to work with the Timed Script of a spoken tutorial and the scraping process works only with the Timed Script. Contributor names are hard-coded to not be present in the mind map. The mind map is generated with an appropriate value of the repulsive strength of keyword nodes in the force simulation to accommodate the mind map in the given amount of space available. The mind map is designed to generate 15 keywords per script for the same purpose. Along with the lemmatizer and keyword extractor used, other lemmatizers and keyword extractors are also supported which could be enabled easily.

5 Installation and updation guide

5.1 Installation

Creating a virtual environment

```
$ python/python3 -m venv mind_map_venv
$ source mind_map_venv/bin/activate
```

Installation of requirements

```
$ pip install -r requirements.txt -q
```

5.2 Updation

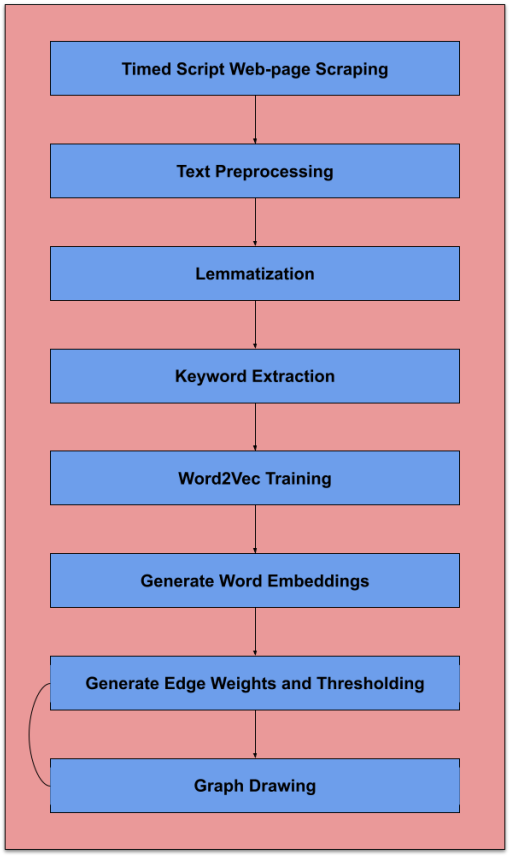
Any update to the mind map generation pipeline should be made to the main_code folder or the views.py file in the django application. To update the lemmatizers or keyword extractors to be used out of the ones available, the views.py file should be updated with the corresponding function call from the lemmatizers.py or the extractors.py file. The number of keywords generated per script can also be updated in the views.py file by changing the value of the top_n argument passed to the keyword extraction function. The percentile criteria of the threshold for edge strength can be updated through the graph.py file. Various factors related to the force simulation such as charge, center, etc. can be updated through app.js.

6 Code and Working

6.1 Code

The code is available at this GitHub repository. It is currently present on two branches for two different web applications. The final_integration branch contains code for the web application that can be used as an integration into the spoken tutorial website for generating a mind map for a script using the "Generate Mind Map" button at the bottom of every page.

6.2 Working



7 Results

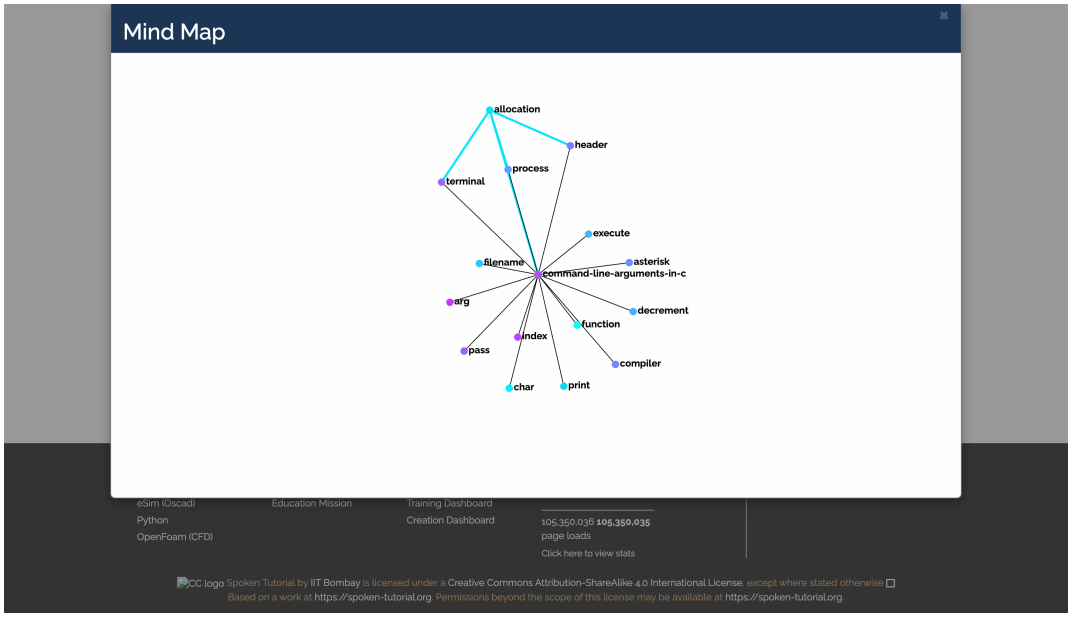


Figure 1: Integrated App (final_integration branch

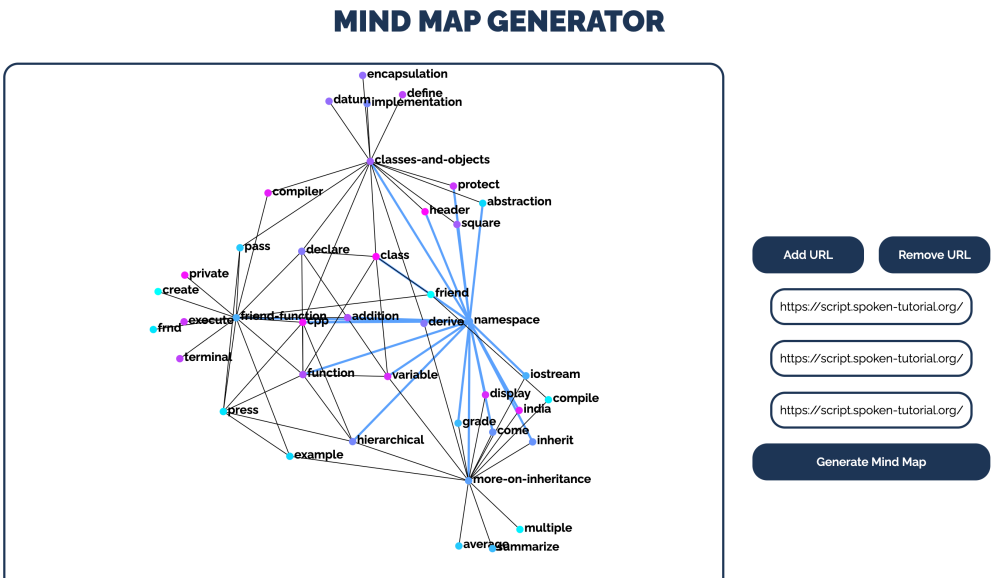


Figure 2: Standalone App (master branch)

References

[1] Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013a. Efficient estimation of word representations in vector space. ICLR. https://arxiv.org/abs/1301.3781.Google Scholar.

[2] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. Inform. Process. Lett., 31:7–15, 1989.

[3] Devlin, Jacob et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." ArXiv abs/1810.04805 (2019): n. pag.