BlOOMQUEST: A PERSONALIZED LEARNING PLATFORM

Weerasekara N.N.

(IT20133504)

BSc (Hons) in Information Technology
Specializing in Data Science

Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

February 2023

BlOOMQUEST: A PERSONALIZED LEARNING PLATFORM

Weerasekara N.N.

(IT20133504)

Dissertation submitted in partial fulfillment of the requirements for the Bachelor of Science in Information Technology Specializing in Data Science

Department of Information Technology

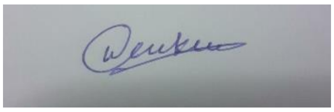
Sri Lanka Institute of Information Technology
Sri Lanka

February 2023

Declaration

“I declare that this is my own work and this dissertation1 does not incorporate without acknowledgement 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 acknowledgement 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)”.

Signature:  Date:2023-09-09

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

Signature of the supervisor: Date

Abstract

Creating well-balanced and high-quality exams that cater to the various cognitive levels of undergraduates is crucial. To achieve this, many lecturers rely on Bloom's Taxonomy cognitive domain, which is a popular framework used to assess students' intellectual abilities and skills. However, despite its widespread use, many students are not aware of Bloom's Taxonomy and its impact on their learning experience. This lack of understanding can result in students missing vital opportunities for growth and development in their academic pursuits. To address this issue, a personalized self-learning system has been developed based on Bloom's Taxonomy, which aims to help students understand and apply it in their own learning process. The system is personalized according to the student's learning material and level of performance. It extracts named entities and relations between them from the student's material and generates a knowledge graph. This approach provides a unique opportunity for students to learn from the type and level of content that they are best suited to, ultimately leading to better performance and more effective learning. The output of this project is a desktop application that provides a fully-fledged personalized self-learning system based on Bloom's Taxonomy. The proposal focuses on implementing a personalized dynamic study plan and recommending external study resources based on the user's performance level and Bloom's Taxonomy. The system's ability to find relevant external resources for the user is critical in ensuring that they have access to a variety of study materials that are tailored to their specific learning needs. The system may also create a thorough mind-map for a given study material, which will improve the user's learning process. In summary, this project aims to address the lack of understanding of Bloom's Taxonomy among students and provides them with a personalized self-learning system to help them understand and apply it to their own learning process. This system can ultimately lead to better academic performance and improved learning outcomes for undergraduates.

Acknowledgement

As I began my research adventure, I discovered myself surrounded by a vast reservoir of assistance, knowledge, and inspiration. I express my sincere thanks to all who were instrumental in helping to shape my thesis.

I would like to start by expressing my sincere gratitude to my supervisor, Mr. Prasanna Sumathipala, whose unwavering leadership, tolerance, and knowledge guided me through the challenges of my study. The basis of my trip was your astute criticism. Thanks for all the wise counsel, Mr. Sathira Hettiarachchi, my co-supervisor.

I want to express my sincere appreciation to everyone in my research team. The convoluted research process was made more interesting and delightful by your friendship, conversations, brainstorming sessions, and collaborative attitude.

I owe the Research Panel a huge debt of gratitude for their time, knowledge, and critical criticism. Your astute insights and recommendations have really helped me to improve my study.

Table of Contents

List of Figures

1. Introduction

Mind Mapping is a widely recognized method for taking notes, which has proven to be highly effective in facilitating learning and self-study. This technique allows one to create visual representations of concepts and ideas in a non-linear, free-form manner that is easy to understand and remember.

Mind mapping can play a crucial role in the self-learning process for students. By allowing the mind to draw correlations and connections between many pieces of knowledge, this method promotes creativity and aids in the development of critical thinking skills. Mind mapping can help students to visualize the relationships between different concepts and ideas. Mind maps provide a graphical representation of the relationships between different topics, making it easier to understand complex information and identify connections between different concepts. Before we conducted the research, we first invited respondents to complete a survey in which we inquired about their ability to generate Mind maps. And this was their response.

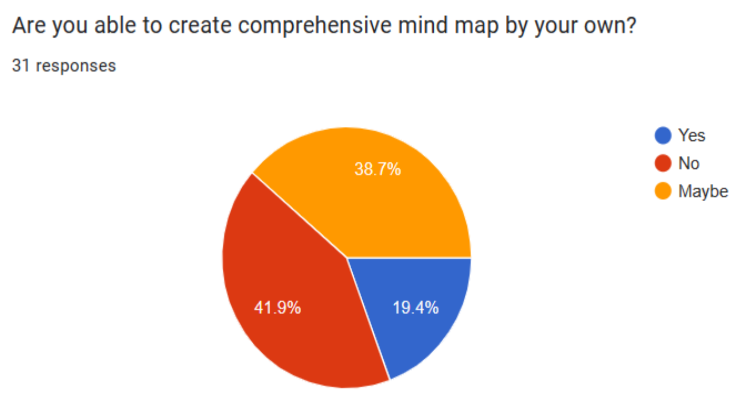


Figure 1 : Ability to Generate mind-maps.

The thing is many students don't use mind maps because creating mind maps requires highly increased mental effort and complete focus. Only a few are focused enough to draw good mind maps and the rest don't have that complete focus to draw mind maps. My point is that if every student gets a chance to summarize their work using a mind map, they could improve their memory retention significantly. This paper provides a tool that builds a mind map according to the given study material by the student. This will be helpful for the student who doesn't know how to create or make a mind map. Overall, mind mapping is a flexible and useful tool that may be used in a variety of contexts, including education, business, and personal growth, by people of all ages and professions.

In conclusion, Mind mapping can be a powerful way to help students understand and apply it in their own learning process. This technique can help to simplify complex information and provide a visual representation of the relationships between different concepts, making it easier for students to understand and recall the material. By using this tool, students can improve their self-studying process, which can lead to improved academic performance and success in their academic pursuits.

1.1 Background literature

Researchers and developers have made outstanding progress in using the strength of NLP and ML approaches to transform different fields into the constantly developing field of artificial intelligence and machine learning. These projects include a wide range of applications, such as improving natural language comprehension and creating in-depth knowledge graphs and mind maps. Interdisciplinary teams have been at the vanguard of these initiatives for many years, cooperating to realize the full potential of these technologies. In this context, we explore the dynamic world of NLP and ML applications, emphasizing their profound influence on knowledge representation and information structuring.

In 2021, Yeoil, Taejin and Namgyu Kim who were Graduates in Kookmin University in faculty of Business IT have done research about Deep Learning-Based Knowledge Graph Generation for COVID-19 [1]. In order to create a knowledge network particular to COVID-19, this study suggests an Open Information Extraction (OpenIE) method based on unsupervised learning. The system uses a COVID-19 entity dictionary and a fine-tuned BERT language model to extract connecting words between entities and outperforms original BERT in terms of accuracy and score. It highlights the two methods of natural language processing - statistical and neural network - and emphasizes the benefits of using neural networks, such as the ability to efficiently expand the model.

Researchers from the University of Mexico, Ivan Lopez-Arevalo, Jose L. Martinez-Rodriguez, and Ana B. Rios-Alvarado conducted research in 2018 on an OpenIE-based method for building knowledge graphs from text. [2] This method involves the use of Natural Language Processing (NLP) and Information Extraction (IE) techniques to convert the input text into a machine-readable format consisting of RDF triples. RDF graphs are used to represent entities and relationships between them, where entities can be anything that has a unique identifier, such as a person, place, concept, or event. Relationships between entities are represented as edges in the graph, with properties that describe the nature of the relationship. By constructing a graph from these triples, Knowledge Graphs can be used to represent knowledge from multiple sources and domains and to support various applications, such as information retrieval and natural language processing.

A research study for creating mind maps from articles using machine learning was completed in 2019 by M.F. Kuroki, L.S. Riza, and Rasim at the department of computer science education at Universitas Pendidikan Indonesia. [3] For the data collection they have used articles. The topic sentence of a paragraph is chosen using the information retrieval approach, pre-processing, core NLP, and feature extraction approaches in model creation. Application development uses a stage-by-stage, linear process model. In experiments, experts choose the criteria for articles, and the application and the experts choose the topic phrases. The results are articles with topic sentences in each paragraph. With a title, subtitle, and topic sentence for each paragraph, this system creates teaching materials in the form of a mind map. The system's output is contrasted with the average values produced by two human experts, whose accuracy rate averages 53.55%. This indicates a moderate level of system correctness.

In 2013, Ayu Purwarianthi, Athia Saelan, Irfan Afif, Filman Ferdian, and Alfan Farizki conducted research on developing an autonomous mind map generator in Indonesian language at the school of electrical engineering and informatics institution in Indonesia. [4] Indonesian Mind Map Generator utilizes Indonesian natural language understanding tools such as a POS tagger, syntactic parser, and semantic analyzer to facilitate easy creation of Mind Map objects. The tools' accuracy rates for the POS tagger, syntactic parser, and semantic analyzer are 96.5%, 47.22%, and 62.5%, respectively. They were created with the aim of addressing the dearth of Indonesian language resources. The Mind Map generator also employs radial drawing visualization and an editor for modifications. In evaluation, the Mind Map object was easily understood for simple sentences by 5 respondents.

In 2020 Danilo Dessi, Franceso Osborne, Diego Recupero, Davide Buscaldi and Enrico Motta have done research on knowledge graphs creation from NLP. [5] The abundance of scientific literature makes analysis difficult, necessitating technical infrastructures for effective browsing, analysis, and research forecasting. Although knowledge graphs, which are extensive networks of entities and relationships, are useful tools, they do not explicitly represent the knowledge found in research papers. This work introduces a unique architecture that uses NLP and ML methods to extract entities and relationships from research publications and add them to a knowledge graph. A scientific knowledge network was created using the hybrid approach's accurate extraction of 109,105 triples from 26,827 abstracts. The method is universal and adaptable to any domain.

In 2011 Robert, Mirko and Mladen from university of Zagreb, Croatia has implemented a mind map generating software model by using text mining algorithm. [6] This software is compatible with desktop, laptop, PDA, and mobile devices. A web service based on SOA is advised because PDAs and mobile phones may execute slowly. The algorithms will be run by the web service, which will then produce a mind map that will be saved on a database server. All mind maps can be searched for and downloaded by users, and the database can be used to integrate mind maps and conduct additional research. This software has accuracy related to other systems.

1.2 Research Gap

It is clear from reading the corpus of published research articles that text mining methods [6] and machine learning techniques [3] are the main foundations for the present automated systems for producing mind maps from text input. Although some methods have shown potential, they frequently cannot completely incorporate domain-specific information. As a result, the mind maps they produce can be insufficient or fall short of correctly capturing the text's subtle purpose.

These systems' inability to handle large text inputs successfully stands out as one significant flaw. The system suggested in this study seeks to deal with these restrictions, nevertheless. The goal of this innovative method is to develop a knowledge-based system that makes use of domain-specific knowledge to produce mind maps that are not only much more accurate but also significantly more relevant in their representation of the underlying text. This method improves its capacity to identify key concepts and links, leading to more precise and insightful mind maps by adding specialist information pertinent to the topic matter.

Comparison Criteria	Paper [3]	Paper [4]	Paper [6]	Proposed System
knowledge-based concepts	✗	✗	✗	✓
Handle long text	✓	✗	✗	✓
Accuracy	✓	✗	✗	✓
Speed	✗	✓	✓	✓
User needs focus	✗	✗	✗	✓

1.3 Research Problem

"Can a knowledge-based system improve the usefulness of mind maps generated from long text data compared to existing text mining algorithms and Machine learning approaches?"

An ongoing problem with automated mind map development is accuracy, as shown by a review of the aforementioned research publications. While many of these methods are novel and show promise, they have had trouble consistently producing mind maps that are extremely accurate [1][2][3][4]. It is critical that these representations are accurate since errors may result in erroneous or incomplete interpretations of the underlying data.

Additionally, a noteworthy flaw in the current systems is their inability to handle lengthy text inputs well. Their usefulness is severely limited by this flaw, especially when working with in-depth study materials or long texts. The ability of a system to process lengthier inputs is crucial since it directly affects the system's potential to produce detailed and insightful mind maps.

Mind map's usefulness is closely related to the veracity of the information it depicts. The effectiveness of the mind map as a tool for learning or organizing knowledge is significantly reduced if errors are present. Therefore, ensuring that these technologies serve their intended purpose efficiently and meaningfully requires the pursuit of better accuracy and the capacity to accept lengthier texts in mind map development. In the area of knowledge visualization and representation, addressing these issues and improving the accuracy of produced mind maps remains an essential goal.

Before conducting our research, we conducted a survey to gather respondents' feedback on their satisfaction with current automated mind-map generators. Here are their responses.

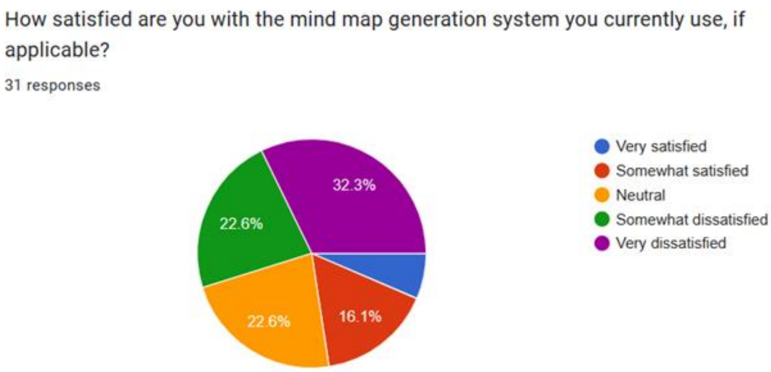


Figure 2: Satisfaction about current mind maps

1.4 Research Objectives

The primary goal of this component is to design and implement a knowledge-based system that can generate more precise and significant mind maps from textual data. By developing a knowledge-based system for mind map generation, this project aims to improve upon the limitations of existing text mining algorithms and machine learning approaches, which may not always capture the full semantic meaning of the input text. The proposed system seeks to address this issue by incorporating domain-specific knowledge and rules into the analysis process, which can enhance the accuracy and relevance of the generated mind maps. Additionally, the project's overarching objective is to develop a knowledge-based system that can overcome the limitations of current text mining and machine learning approaches and generate more precise and meaningful mind maps from text data.

1.5 Research Sub Objectives

To design a mind map that supports self-study and helps students to better understand and retain key concepts in a particular subject: - The mind map will be designed to provide a visual representation of the connections between the various concepts, making it easier for students to understand and remember them. The mind map will be structured in a way that enables students to navigate through the various concepts and explore the relationships between them.

To review existing literature on automated systems for creating mind maps from text data and identify the limitations of current approaches: - This review will identify the strengths and limitations of current approaches and provide insights into how the proposed system can be improved. The review will also explore the different techniques used for creating mind maps, including machine learning, natural language processing, and data visualization.

To conduct user studies to gather feedback on the usability and usefulness of the proposed system: - The user studies will involve recruiting participants who are studying a particular subject and asking them to use the system to create a mind map based on the material they have learned. The feedback gathered from the participants will be used to improve the system's design and functionality. The user studies will also evaluate the effectiveness of the mind map in facilitating learning and retention of key concepts.

2. Methodology

2.1 Overall Methodology for Generating a Mind map.

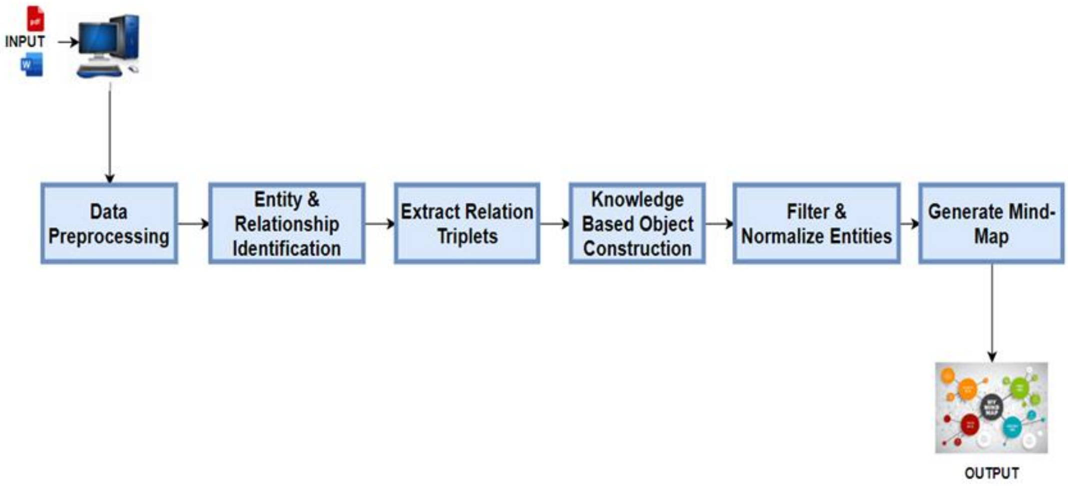


Figure 3: Overall Component diagram for Mind-Map

Our thorough step-by-step process offers a disciplined framework for producing in-depth mind maps using study materials. This strategy effortlessly incorporates many steps, each of which plays a crucial part in the process. We guarantee that the research material is ready for analysis by starting with thorough data pretreatment and cleaning and organizing it. This stage ensures that the succeeding processes have access to accurate and consistent data by removing duplication and dealing with different document formats.

The entity and relation extraction stage begins after data preparation. throughout this crucial stage. This makes it possible for us to efficiently detect and categorize things and their connections within the research material. Our Mind maps are constructed using these things, and the relationships between them show how intricately connected they are.

Our approach moves on to the extraction of relation triplets after successfully extracting entities and relations. The fundamental framework of our knowledge base is composed of these triplets, which include a subject entity, a predicate (the relation), and an object entity. They help students better understand complicated interdependencies by encapsulating the underlying relationships that underpin the concepts in the study material.

The next step is to build these things methodically now that we have knowledge-based objects at our disposal. This guarantees that the learned information is arranged logically and saved in a structured fashion. This stage not only increases the effectiveness of our mind map creation but also provides students looking for a deeper comprehension of the subject with an invaluable resource.

We use entity filtering and normalization to further improve the coherence and clarity of our mind maps. Issues with synonyms, abbreviations, and variations in entity names are resolved in this stage. We improve the final mind maps' readability and efficacy by standardizing entity representations.

In the end, our process results in the creation of comprehensive mind maps. We create visually perceptible representations of the structured data and knowledge-based items by utilizing graph theory and visualization approaches. The outcome is a dynamic mind map that accurately replicates the study material's underlying hierarchical structure and complex linkages. Entities become nodes; relationships become edges.

Let me explain how these steps are taken, using relevant tools and technologies.

2.1.1 Preprocess the Paragraph

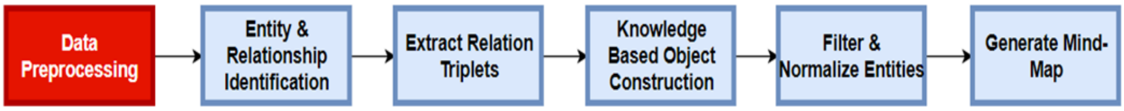


Figure 4: Preprocessing Step

In this system, text preprocessing is a crucial first step that entails cleaning and converting unstructured text input into a more organized and manageable format. This procedure often entails steps like deleting superfluous letters, punctuation, and special symbols, changing the case of the text, and tokenizing the text to separate it into words or tokens. Text preprocessing delivers more accurate and useful results in many text-based applications. Text preprocessing helps enhance the quality of text data by making it more consistent and acceptable for later analysis.

2.1.2 Extract entities and relationships

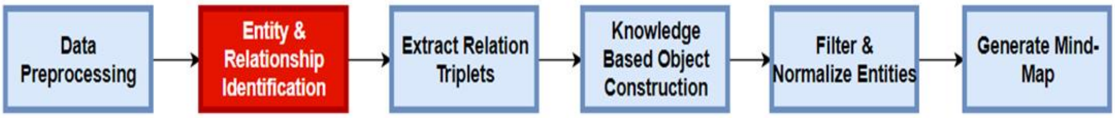


Figure 5: Extract Entities and Relationships Step

The process of locating and classifying certain things or objects inside a text is known as entity extraction, often referred to as named entity recognition (NER). These entities might be actual things like people, companies, places, or dates, or they could be more domain-specific terminology like diseases, substances, or financial instruments. Entity extraction's main objective is to discover and correctly classify these entities inside the text, frequently by labeling them with standard terms like "person," "organization," or "location," or with bespoke terms made specifically for the intended application.

Finding and extracting significant links or linkages between entities stated in a text is referred to as relationship extraction or relation extraction. Recognizing how things relate to one another and if they play certain roles or have connections that may be captured by predetermined relationships is necessary to achieve this. For instance, connection extraction in a news item would entail noting that "Apple Inc." (entity) purchased "Tesla" (entity) and designating their relationship as a "acquisition." To properly identify the nature of the relationship, relationship extraction often requires knowledge about the context of the entities and the surrounding language. Information retrieval, knowledge graph construction, and the creation of organized databases from unstructured text all benefit from its use.

In tasks involving natural language processing, named entity recognition (NER) and relationship classification (RC) are often used algorithms for entity and relationship extraction. The conventional pipeline, which employs NER before moving on to RC, might, however, generate mistakes that spread throughout the procedure. This sequential method can have its limitations, particularly when tackling connections in text data that are intricate or subtle. A preset set of relation types is another restriction placed on RC, which may not fully account for all possible links between entities.

Recent developments in natural language processing have proposed creative end-to-end techniques that try to handle both problems concurrently in order to solve the difficulties presented by the sequential application of Named Entity Recognition (NER) followed by Relation Classification (RC). Relation Extraction (RE) is the term used most frequently to describe this integrated process. In the framework of this post, we will go into the use of the amazing REBEL end-to-end model, which was created by BabelScape. Researchers and practitioners can accelerate the entity and connection extraction process, reducing the possibility of error propagation while increasing the ability to capture a greater variety of relation types, by implementing REBEL and other cutting-edge models.

In order to translate a phrase with entities and implicit relations into a sequence of triplets that explicitly refer to those relations, BabelScape trained the text-to-text model REBEL by fine-tuning BART. More than 200 distinct relation kinds were used to train it. Using entities and relations discovered in Wikipedia abstracts and Wikidata, the authors constructed a bespoke dataset for REBEL pre-training and filtered it using a RoBERTa Natural Language Inference model (similar to this model). To learn more about how the dataset was created, check the paper. On a variety of benchmarks for Relation Extraction and Relation Classification, the model performs well. And for this step I'm using this pre-trained model to extract entities and relations in an accurate way.

2.1.3 Extract relation triplets from the text that has been processed by REBEL

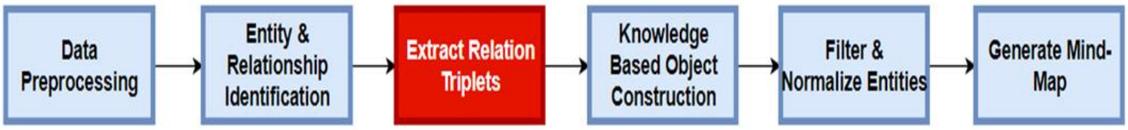


Figure 6: Extract relation triplets from the text that has been processed by REBEL

The creation of a custom function that can parse the structured strings produced by the REBEL model and convert them into relation triplets is a vital next step in the procedure. The subject, relation type, and object of each extracted piece of knowledge are all contained inside these triplets, which operate as the basic units of extracted knowledge. The function, which facilitates this translation, accounts for the addition of additional tokens during the model training phase, such as placeholders like "<triplet>," "<subj>," and "<obj> ". The borders and functions of entities and relationships inside the produced strings are clearly defined by these tokens.

Since each connection is represented as a dictionary, the function analyzes these strings to create a list of relations. These dictionaries have three fundamental words: "head" to indicate the subject (for example, "Fabio"), "type" to indicate the sort of relation (for example, "lives in"), and "tail" to indicate the object (for example, "Italy"). By using this function, we fill the gap between the model's output and a structured, useable representation of the knowledge that was extracted, making it more available and usable for later applications like knowledge graph generation and information retrieval.

2.1.4 Pass extracted triplets into a knowledge-based object.

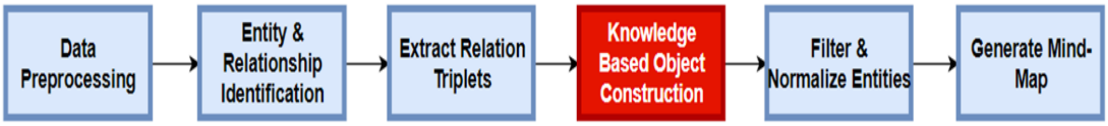


Figure 7: Pass extracted triplets into a knowledge-based object.

Our knowledge base creation approach includes a filtering stage as well, to improve accuracy and coherence. Entity linking, which includes comparing the retrieved entities with Wikipedia pages, is a successful filtering strategy. This stage involves checking to see if terms like "Cristiano Ronaldo" and "Cristiano" have a similar Wikipedia page. In the event that such a relationship is found, the entities are normalized to the page's title, combining them into a single representation. It's vital to note that this approach is predicated on the notion that Wikipedia consistently has accurate information on these entities as a result of user contributions. A more accurate and focused depiction of entities is made possible by temporarily excluding from the knowledge base any entities without associated Wikipedia entries.

A significant element of our knowledge base design is the method "are_relations_equal," which provides a way to determine whether two relations are equivalent based on the terms "head," "type," and "tail." This technique is crucial for knowledge base management because it enables us to establish whether two relations belong to the same information or to different facets of the same notion. We construct a related equality criteria by comparing these qualities, improving the accuracy and effectiveness of operations inside the knowledge base. When working with huge datasets or updating the knowledge base, this functionality is especially useful for ensuring that redundant or repeated relations are properly detected and maintained. Overall, "are_relations_equal" is a crucial technique that improves precision and coherence.

2.1.5 Filter and Normalize the Entities

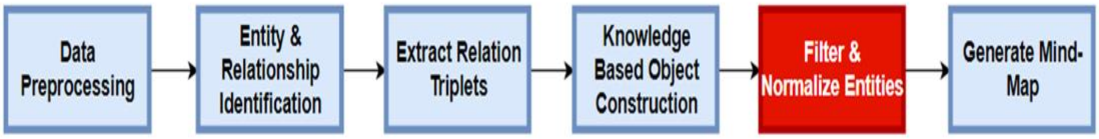


Figure 8: Filter and Normalize the entities.

Our knowledge base creation approach includes a filtering stage as well, to improve accuracy and coherence. Entity linking, which includes comparing the retrieved entities with Wikipedia pages, is a successful filtering strategy. This stage involves checking to see if terms like "Cristiano Ronaldo" and "Cristiano" have a similar Wikipedia page. In the event that such a relationship is found, the entities are normalized to the page's title, combining them into a single representation. It's vital to note that this approach is predicated on the notion that Wikipedia consistently has accurate information on these entities as a result of user contributions. A more accurate and focused depiction of entities is made possible by temporarily excluding from the knowledge base any entities without associated Wikipedia entries.

A significant element of our knowledge base design is the method "are_relations_equal," which provides a way to determine whether two relations are equivalent based on the terms "head," "type," and "tail." This technique is crucial for knowledge base management because it enables us to establish whether two relations belong to the same information or to different facets of the same notion. We construct a related equality criteria by comparing these qualities, improving the accuracy and effectiveness of operations inside the knowledge base. When working with huge datasets or updating the knowledge base, this functionality is especially useful for ensuring that redundant or repeated relations are properly detected and maintained. Overall, "are_relations_equal" is a crucial technique that improves precision and coherence.

2.1.6 Visualize the Mind-Map

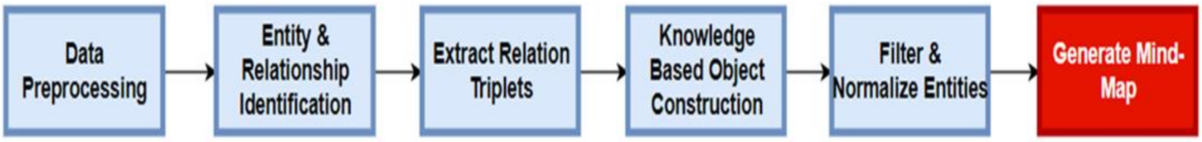


Figure 9: Visualize the Mind-map.

A crucial step in turning abstract data into a clear and visual representation is the creation of a graph visualization using the relations held in a knowledge base (KB). The Pyvis package, a potent tool for creating interactive network graphs, is used in this visualization stage. Using Pyvis, we can convert the intricate network of relations into an understandable and aesthetically pleasing graph structure. A visual representation of the underlying knowledge structure is created by turning each relation into a node and representing the connections between them as edges.

Once the graph has been created, it allows users to explore the links and interconnections within the knowledge base dynamically while also being visually instructive and engaging. The generated graph is stored as an HTML file for easy distribution and additional analysis. This ensures accessibility and shareability. This visual representation not only makes the knowledge base easier to grasp, but it also provides researchers, analysts, and learners with a thorough understanding of the links between the many entities in the data.

2.2 Functional and non-functional requirements

The characteristics and functionalities that this system must have to satisfy the needs of its users are known as functional requirements. Below main points specify what must be done by this system and how it must function in response to particular inputs or activities.

- Ability to extract and process text data from various file formats (e.g., PDF, Word).
- Ability to identify key concepts and relationships within the input text data.
- Ability to generate a visual mind map that represents the key concepts and relationships.
- Ability to allow users to modify and edit the generated mind map.
- Ability to export the generated mind map in various file formats (e.g., PNG, SVG).

Nonfunctional criteria are used to determine the performance and quality attributes of this system. These standards give overall performance a higher priority than system functionality.

- Accuracy: The system should accurately represent the key concepts and relationships.
- Usability: The system should be user-friendly and easy to navigate.
- Security: The system should ensure the confidentiality and integrity of the text data and generated mind maps.
- Performance: The system process text and generates mind maps in a reasonable time frame. It explains how in the below,

Splitting into Spans:

Working with lengthy textual content creates a few difficulties, and the practice of breaking it up into smaller, more manageable pieces, known as "spans," offers a workable answer. Processing the full document at once might be computationally and memory expensive when working with large materials. In order to improve efficiency and accuracy, it is crucial to design a function that breaks the text into spans.

Each span represents a distinct section of the original text, and in order to guarantee thorough coverage, these spans are made to slightly overlap one another. Because of this overlap, there is less chance of overlooking important information that could otherwise be scattered across span borders. This method allows the software to manage and analyze large amounts of textual data efficiently, making it more suitable for various NLP tasks like entity extraction, relationship detection, or summarization. Splitting into spans improves computing speed and encourages a more organized and methodical approach to managing vast amounts of text input, which enhances the quality of the information and insights that are finally gained.

2.3 Implementation

I used a combination of potent Python packages to apply the process and extract insightful information from textual input. The first stage in natural language processing (NLP) jobs is to load a pre-trained sequence-to-sequence model and tokenizer using the Hugging Face Transformers library. The code goes on to show how to quickly handle long text documents by breaking them up into manageable spans and creating meaningful relations using the NLP model. It also makes use of PyTorch for tensor operations, Pyvis Network for displaying the extracted knowledge graph, and the Wikipedia API to add more entity-specific data to the graph. This complete technique is a great resource for many NLP-driven applications since it provides a holistic approach to processing, analyzing, and graphically portraying textual data.

Here are the python libraries that I used to implement the component,

- **transformers:**

  Purpose: This library is used for natural language processing (NLP) tasks, particularly for working with pre-trained language models.

  Why: The Hugging Face Model Hub is used to load a pre-trained sequence-to-sequence model (AutoModelForSeq2SeqLM) and its accompanying tokenizer (AutoTokenizer). In this instance, these models are used to produce textual outputs for the purpose of extracting relationships from the input text.

- **math (Python Standard Library):**

  Purpose: The math library provides mathematical functions.

  Why: It is used in the code to do mathematical calculations, particularly to figure out how many spans are required to analyze large input text.

- **torch (PyTorch):**

Purpose: PyTorch is an open-source machine learning library for deep learning and artificial intelligence.

Why: The code employs it for a variety of machine learning tasks. Tensors from PyTorch are used to manage model input and output data.

- **Pyvis (Pyvis Network):**

Purpose: Pyvis Network is a Python library for visualizing network graphs.

Why: It is utilized to provide a visual representation of the knowledge graph that was extracted from the input text. This enables the representation of entities and relations graphically.

- **Wikipedia :**

Purpose: The wikipedia library provides an API for querying and retrieving information from Wikipedia.

Why: It is employed to retrieve further information on the entities referenced in the input text. For entities mentioned in the text, it specifically retrieves Wikipedia page names, URLs, and summaries.

React for the front end and Flask for the back end are combined in the implementation to create a strong and dynamic web application. The cornerstone for the frontend is React, a well-known JavaScript package that makes it possible to build responsive and dynamic user interfaces. It makes it easier to create reusable UI components, which greatly increases the application's modularity and maintainability. This guarantees effective updates and enhanced efficiency for a smooth user experience, together with its virtual DOM.

The approach combines React for the front end with Flask for the back end to produce a robust and dynamic web application. React, a well-known JavaScript library that enables the creation of responsive and dynamic user interfaces, serves as the foundation for the frontend. It facilitates the creation of reusable UI components, considerably enhancing the modularity and maintainability of the program. Together with its virtual DOM, this ensures efficient updates and increased efficiency for a seamless user experience.

2.4 Testing

The main goal of this painstakingly created test plan is to carefully assess and certify the quality and authenticity of the mind map produced by using the knowledge graph. In essence, it aims to make sure that the finished mind map serves as an accurate and thorough representation of the information included in the given text. This multidimensional review procedure is intended to make sure that the mind map properly captures every aspect, subtlety, and aspect of the document's content. By doing this, it not only demonstrates the durability of the knowledge graph extraction process but also highlights how well it can organize and present complicated textual information. In the end, this test strategy acts as a crucial quality control mechanism, ensuring that the mind map accurately replicates the key ideas and relationships in the text and equipping users with a potent tool for data exploration and understanding.

In the framework of this extensive test scenario, the careful assessment of the system's performance in producing a mind map from an input paper is the main point of attention. The first stage in this crucial testing scenario is the input of a document into the system, followed by a sequence of painstakingly planned processes. The created mind map is then thoroughly examined, and its contents are contrasted with those of the original material. The main goal is to verify with a high degree of confidence that the mind map generated by the system represents the complex relationships, ideas, and details of the text in an authentic and correct manner.

The system's capacity to successfully condense complicated textual material into an organized and readable manner will be put to the test in this situation. It also looks for any potential differences or inconsistencies between the mind map and the original material, revealing important information about the precision and dependability of the system. Stakeholders may be certain that the system will enable knowledge extraction and understanding, promoting informed decision-making and facilitating information exploration, by carefully implementing this scenario.

Experiment 1: Input a simple document with a single concept and verify that the generated mind map accurately reflects the concept.

1. Preparation: Choose a straightforward paper with a single, distinct topic, such as "The Earth's Rotation."

2. Execution: Input the document into the system for mind map generation.

3. Validation Steps:

   - Check the mind map that was developed to make sure the core node is "The Earth's Rotation," a single notion.

   - Make sure the mind map doesn't contain any more unconnected connections or thoughts.

4. Verification: The generated mind map precisely and completely captures the single notion from the paper.

Experiment 2: Input a document with multiple concepts and verify that the generated mind map accurately reflects all the concepts.

1. Preparation: Pick a paper containing a variety of clear themes, such "Renewable Energy Sources."

2. Execution: Input the document into the system for mind map generation.

3. Validation Steps:

- Check the mind map that was created to make sure it has all the different concepts, the right nodes, and the right connections.

- Verify that the document's important ideas are not omitted from the mind map.

4. Verification: All of the concepts from the input paper are represented precisely in the generated mind map.

Experiment 3: Input a complex document with multiple sub-concepts and verify that the generated mind map accurately reflects all the sub-concepts.

1. Preparation: Create a complicated text with several levels of supporting ideas, such as "Artificial Intelligence and Its Applications."

2.Execution: Send the complicated document to the mind map generating system.

3. Validation Steps:

- Make sure the created mind map properly captures all the sub-concepts and establishes a hierarchical structure by carefully going through it.

- Verify that the mind map accurately depicts the connections between the primary concepts and their supporting notions.

4. Verification: The complicated document's convoluted structure of concepts is successfully represented by the mind map that was generated.

Experiment 4: Input a document with ambiguous concepts and verify that the generated mind map accurately represents the most relevant concepts.

1. Preparation: Pick a document using terms like "bank" that might mean different things depending on the context.

2. Execution: Input the document into the system for mind map generation.

3. Validation Steps:

- Make sure the created mind map emphasizes the most contextually appropriate interpretation of confusing topics by analyzing it.

- Make sure there are no erroneous or irrelevant notions in the mind map.

4. Verification: The generated mind map appropriately depicts the concepts that are applicable in the given context for the unclear phrases in the paper.

Experiment 5: Input a document with no clear concepts and verify that the generated mind map does not produce any misleading information.

1. Preparation: Choose a piece of writing that lacks clear concepts or a clear framework, such a collection of phrases at random.

2. Execution: Provide the document to the system for mind map generation.

3. Validation Steps:

- Check the created mind map carefully to make sure it doesn’t include any fictitious concepts or links in the absence of a defined content structure.

- Make sure there are no new details added to the mind map that are not necessary.

4. Verification: When presented with a material missing clear concepts, make sure the created mind map suitably abstains from supplying misleading or unnecessary information.

2.5 Results and Discussion

This section contains the findings and in-depth analyses of the painstakingly constructed test cases used to assess the effectiveness of our knowledge graph-based mind map creation method. These test cases were carefully designed to evaluate the system's capacity to represent material from a variety of documents, ranging from simple to complicated, accurately and to manage ambiguous or unstructured data. Through these experiments, we aimed to confirm the system's ability to visualize and extract knowledge, eventually hoping to offer insightful information on its dependability and applicability for diverse document processing jobs. The findings and discussions that follow offer a thorough examination of the system's performance in each test situation, highlighting its advantages and disadvantages in regard to encoding and displaying textual data in a visual and organized manner.

Experiment 1: Input a simple document with a single concept and verify that the generated mind map accurately reflects the concept.

Results: The input paper was successfully converted into a mind map, with a single node designating the notion "The Earth's Rotation." The mental map was free of unnecessary ideas or connections.

Discussion: This experiment exhibits the system's ability to faithfully extract and represent simple, discrete ideas from texts. It worked as anticipated, creating a clear and concentrated mind map.

Experiment 2: Input a document with multiple concepts and verify that the generated mind map accurately reflects all the concepts.

Results: With the help of the input material, the system successfully created a mind map that contained all the various ideas, including "Solar Energy," "Wind Power," and "Hydropower." These concepts' connections with one another were accurately portrayed.

Discussion: The system's capacity to manage documents with several, unique ideas are validated by this experiment. The ability of the created mind map to extract knowledge was demonstrated by how well it captured the variety of concepts contained in the paper.

Experiment 3: Input a complex document with multiple sub-concepts and verify that the generated mind map accurately reflects all the sub-concepts.

Results: The system successfully generated a mind map that showed the input document's hierarchical structure, including the primary concepts and their related supporting concepts. The connections between these ideas were well portrayed.

Discussion: This experiment illustrates how the system can manage papers with layered sub-concepts that are complicated. The created mind map successfully portrayed the complicated hierarchy, demonstrating its value in representing intricate knowledge hierarchies.

Experiment 4: Input a document with ambiguous concepts and verify that the generated mind map accurately represents the most relevant concepts.

Results: In order to discern between the meanings of ambiguous terms in the paper, such as "Bank" in the context of finance and "Bank" as a riverbank, the system created a mind map that emphasized the most contextually pertinent interpretations of such terms. There were no false or irrelevant ideas present.

Discussion: This test scenario demonstrates the system's capacity to understand context and choose the proper meanings for ambiguous phrases. By concentrating on the most important principles, it successfully prevents the introduction of confusion.

Experiment 5: Input a document with no clear concepts and verify that the generated mind map does not produce any misleading information.

Results: The system's ability to comprehend context and select the appropriate interpretations for ambiguous sentences is demonstrated by this test case. It avoids the introduction of confusion by focusing on the most crucial concepts.

Discussion: This experiment demonstrates the system's capability to safely handle unstructured material. It successfully steers clear of producing inaccurate or unnecessary information, demonstrating its ability to retain clarity in the absence of distinct notions.

In conclusion, the test cases show that the system successfully creates mind maps that faithfully represent the content of various publications. It displays its resilience in knowledge extraction and visualization by handling basic and complicated texts, confusing phrases, and unstructured information successfully. The system's dependability and applicability for a range of document processing activities are confirmed by these results.

2.6 Commercialization

The adoption of our ground-breaking approach by industry signifies a sea change in how people engage with and understand learning materials. Our state-of-the-art technology, which can produce thorough mind maps from study materials, has enormous promise in a number of industries. This approach may be used by educational institutions to provide students with powerful visual aids that break down complicated concepts, improve comprehension, and encourage knowledge retention. Our approach can speed up the process of turning long texts into understandable visual representations for use in the corporate environment, supporting effective training, onboarding, and knowledge exchange.

Furthermore, the versatility and scalability of our technology make it appropriate for a variety of applications. Our technology may be included by content producers and e-learning platforms to improve the quality and accessibility of instructional resources for students of all ages. The system's adaptability, which meets a range of educational demands, is ensured by its capacity to handle both straightforward and complex study materials.

From a business standpoint, our approach creates fresh possibilities for licensing and collaboration. In partnership with educational publishers, we can integrate our technology into their online learning environments to give students a more engaging and dynamic learning environment. Businesses looking for cutting-edge knowledge management and documentation solutions can also license our system.

As part of our commercialization plan, we also make specific efforts to sell to corporations, educational institutions, and content producers. Through case studies and demonstrations, we intend to demonstrate the system's efficacy and emphasize its potential to turn conventional study materials into interesting and aesthetically pleasing learning aids. Additionally, we will provide adaptable price structures to meet the various demands and spending limits of our potential customers.

In conclusion, our system's commercialization marks a significant turning point in the fields of education and knowledge management. We hope to provide educators, students, and organizations with a tool that not only clarifies complicated material but also improves the whole experience of learning and knowledge exchange by using the power of visual representation and adaptation. As we set out on this adventure, we're determined to be providing a product that not only lives up to but also beyond the expectations of our customers, eventually changing how information is accessible and comprehended in the digital era.

3. Conclusion

In conclusion, this research project represents a sizable step forward in knowledge structuring and extraction from unstructured textual material. We have exhibited a strategic way for converting unstructured material into organized and aesthetically pleasing representations by methodically detailing a thorough process. We are able to efficiently extract significant connections and entities from textual content by combining state-of-the-art natural language processing techniques, such as named entity identification and relation extraction, with end-to-end models like REBEL.

To sum up, this research endeavor marks a significant advancement in the structuring and extraction of knowledge from unstructured textual data. By carefully outlining a detailed approach, we have shown how to use strategy to turn unstructured content into arranged and visually acceptable representations. By integrating cutting-edge NLP methods, such as named entity identification and relation extraction, with end-to-end models like REBEL, we are able to effectively extract meaningful relationships and entities from textual information.

The implementation of span-based text splitting also recognizes the difficulties presented by lengthy text documents and improves computational effectiveness, guaranteeing that no important information is missed. Collectively, these methodological developments provide a complete framework for developing extensive knowledge bases, laying the groundwork for knowledge graph development, information retrieval, and data-driven decision-making.

This research gives us the skills we need to harness the power of data in an era marked by the flood of textual information, allowing us to gain new insights, forge connections, and improve our comprehension of the world. These techniques will continue to be crucial in bridging the unstructured text and structured information divide as we move forward, encouraging innovation and insight across several areas.

4. References

[1]T. Kim and N. Kim, "Deep Learning-Based Knowledge Graph Generation for COVID-19," Graduate School of Business IT, Kookmin University, Seoul, Korea, 2021.

[2]Jose L. Martinez-Rodriguez, Ivan Lopez-Arevalo, Ana B. Rios-Alvarado,

OpenIE-based approach for Knowledge Graph construction from text, Expert Systems With Applications (2018), doi: 10.1016/j.eswa.2018.07.017

[3] M. F. Nurrokhim, L. S. Riza, and Rasim, "Generating mind map from an article using machine learning," J. Phys.: Conf. Ser., vol. 1280, p. 032023, 2019. DOI: 10.1088/1742-6596/1280/3/032023.

[4] A. Saelan and A. Purwarianti, "Generating Mind Map from Indonesian Text using Natural Language Processing Tools," Procedia Technology, vol. 11, pp. 1163-1169, 2013.

[5] D. Dessì, F. Osborne, D. Reforgiato Recupero, D. Buscaldi, and E. Motta, "Generating Knowledge Graphs by Employing Natural Language Processing and Machine Learning Techniques within the Scholarly Domain," Knowledge-Based Systems, vol. 154, pp. 1-14, 2018.

[6] R. Kudelić, M. Konecki, and M. Maleković, "Mind Map Generator Software Model with Text Mining Algorithm," in Proceedings of the ITI 2011 33rd Int. Conf. on Information Technology Interfaces, Cavtat, Croatia, 2011

5. Appendices

```python
# Load model and tokenizer
tokenizer = AutoTokenizer.from_pretrained("Babelscape/rebel-large")
model = AutoModelForSeq2SeqLM.from_pretrained("Babelscape/rebel-large")
```

Figure 10: Used Python Libraries

```python
# Load model and tokenizer
tokenizer = AutoTokenizer.from_pretrained("Babelscape/rebel-large")
model = AutoModelForSeq2SeqLM.from_pretrained("Babelscape/rebel-large")
```

Figure 11: Code to load the model and Tokenizer.

```
lib
  bindings
    JS utils.js
  tom-select
    JS tom-select.complete.min.js
    # tom-select.css
  vis-9.1.2
    # vis-network.css
    JS vis-network.min.js
```

Figure 12: Folder Structure for the Libraries

```
def extract_relations_from_model_output(text):
    relations = []
    relation, subject, relation, object_ = '', '', '', ''
    text = text.strip()
    current = 'x'
    text_replaced = text.replace("<s>", "").replace("<pad>", "").replace("</s>", "")
    for token in text_replaced.split():
        if token == "<triplet>":
            current = 't'
            if relation != '':
                relations.append({
                    'head': subject.strip(),
                    'type': relation.strip(),
                    'tail': object_.strip()
                })
                relation = ''
            subject = ''
        elif token == "<subj>":
            current = 's'
            if relation != '':
                relations.append({
                    'head': subject.strip(),
                    'type': relation.strip(),
                    'tail': object_.strip()
                })
            object_ = ''
        elif token == "<obj>":
            current = 'o'
            relation = ''
```

Figure 13: Extracting Triplets

```
            object_ = ''
        elif token == "<obj>":
            current = 'o'
            relation = ''
        else:
            if current == 't':
                subject += ' ' + token
            elif current == 's':
                object_ += ' ' + token
            elif current == 'o':
                relation += ' ' + token
    if subject != '' and relation != '' and object_ != '':
        relations.append({
            'head': subject.strip(),
            'type': relation.strip(),
            'tail': object_.strip()
        })
    return relations
```

Figure 14: Return Relations

```python
class KB():
    def __init__(self):
        self.relations = []

    def are_relations_equal(self, r1, r2):
        return all(r1[attr] == r2[attr] for attr in ["head", "type", "tail"])

    def exists_relation(self, r1):
        return any(self.are_relations_equal(r1, r2) for r2 in self.relations)

    def add_relation(self, r):
        if not self.exists_relation(r):
            self.relations.append(r)

    def print(self):
        print("Relations:")
        for r in self.relations:
            print(f"  {r}")
```

Figure 15: Knowledge Base Object Creation

```python
def from_text_to_kb(text, span_length=128, verbose=False):
    # tokenize the whole text
    inputs = tokenizer([text], return_tensors="pt")

    # compute span boundaries
    num_tokens = len(inputs["input_ids"][0])
    if verbose:
        print(f"Input has {num_tokens} tokens")
    num_spans = math.ceil(num_tokens / span_length)
    if verbose:
        print(f"Input has {num_spans} spans")
    overlap = math.ceil((num_spans * span_length - num_tokens) / max(num_spans - 1, 1))
    spans_boundaries = []
    start = 0
    for i in range(num_spans):
        spans_boundaries.append([start + span_length * i, start + span_length * (i + 1)])
        start -= overlap
    if verbose:
        print(f"Span boundaries are {spans_boundaries}")

    # transform input with spans
    tensor_ids = [inputs["input_ids"][0][boundary[0]:boundary[1]] for boundary in spans_boundaries]
    tensor_masks = [inputs["attention_mask"][0][boundary[0]:boundary[1]] for boundary in spans_boundaries]
    inputs = {
        "input_ids": torch.stack(tensor_ids),
        "attention_mask": torch.stack(tensor_masks)
    }
```

Figure 16: Spanning process

```python
# generate relations
num_return_sequences = 3
gen_kwargs = {
    "max_length": 256,
    "length_penalty": 0,
    "num_beams": 3,
    "num_return_sequences": num_return_sequences
}
generated_tokens = model.generate(
    **inputs,
    **gen_kwargs,
)

# decode relations
decoded_preds = tokenizer.batch_decode(generated_tokens, skip_special_tokens=False)

# create kb
kb = KB()
i = 0
for sentence_pred in decoded_preds:
    current_span_index = i // num_return_sequences
    relations = extract_relations_from_model_output(sentence_pred)
    for relation in relations:
        relation["meta"] = {
            "spans": [spans_boundaries[current_span_index]]
        }
        kb.add_relation(relation)
    i += 1

return kb
```

Figure 17: Knowledge based class.

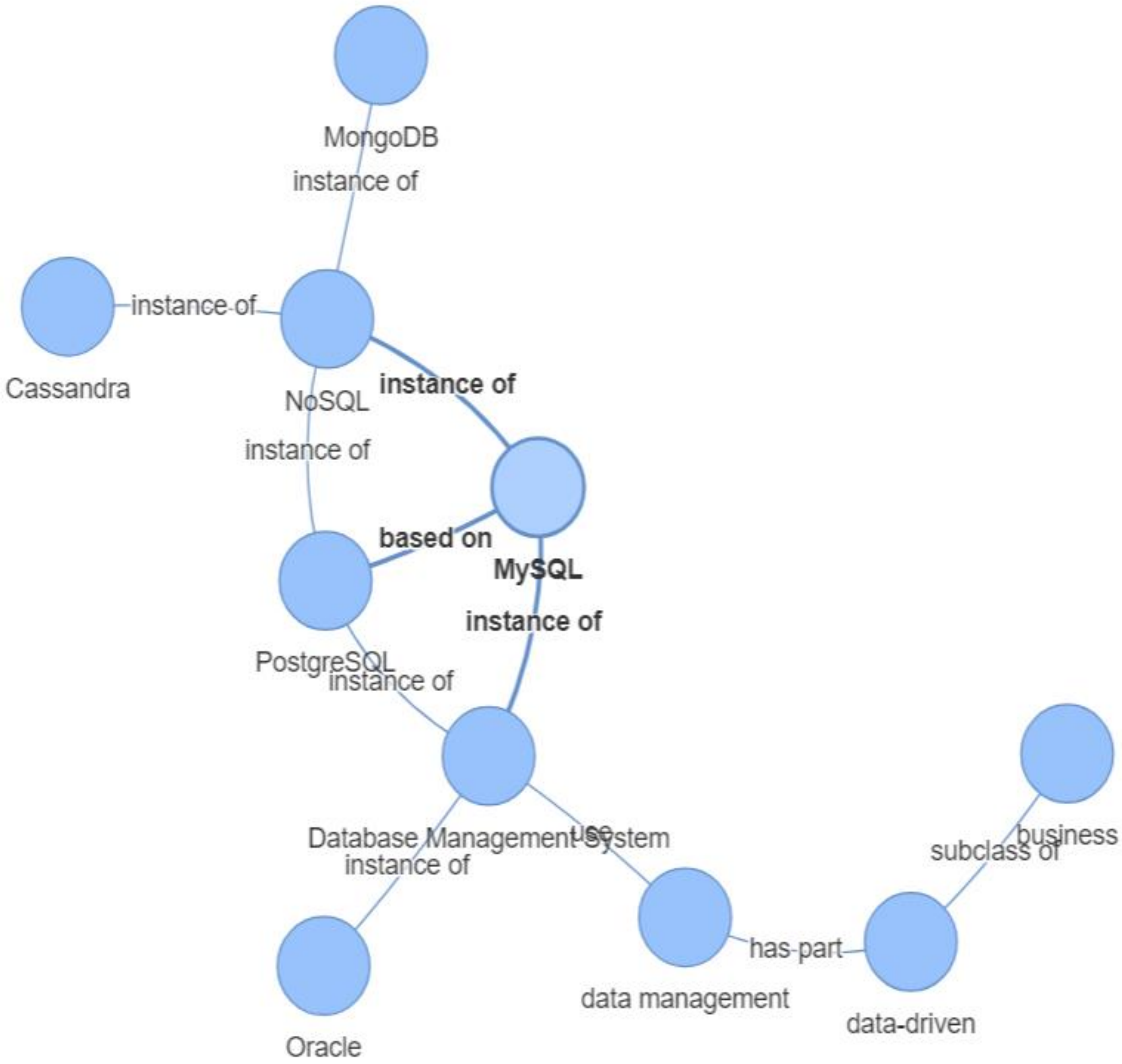


Figure 18: Mind-map Example

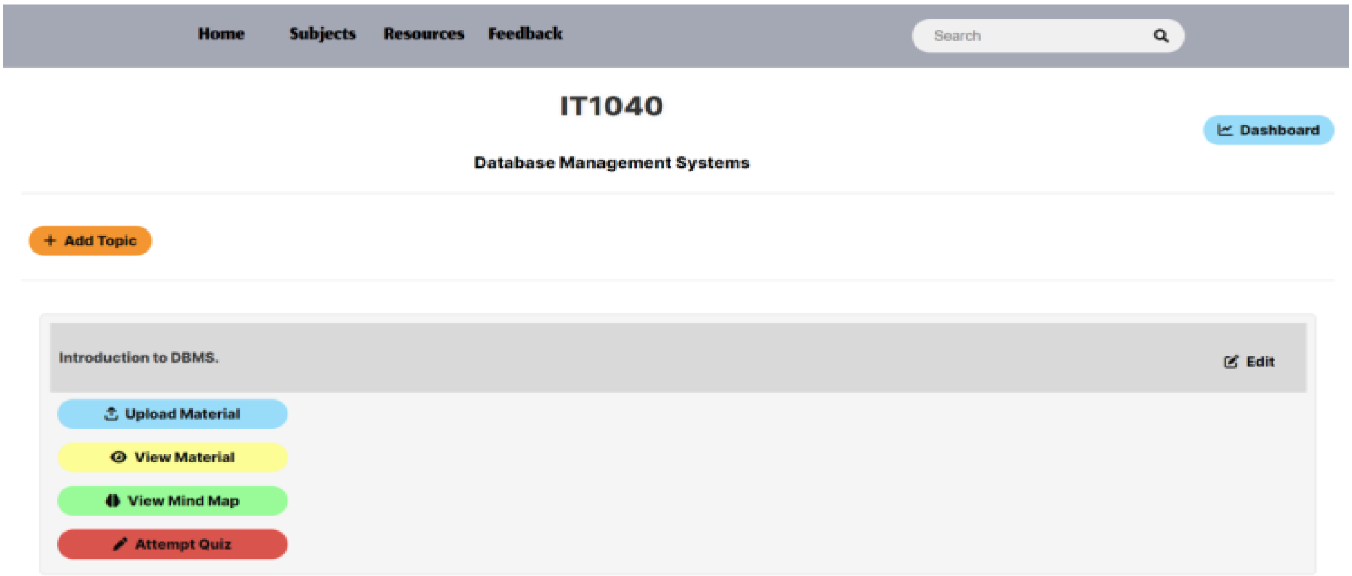


Figure 19: UI of the System 1

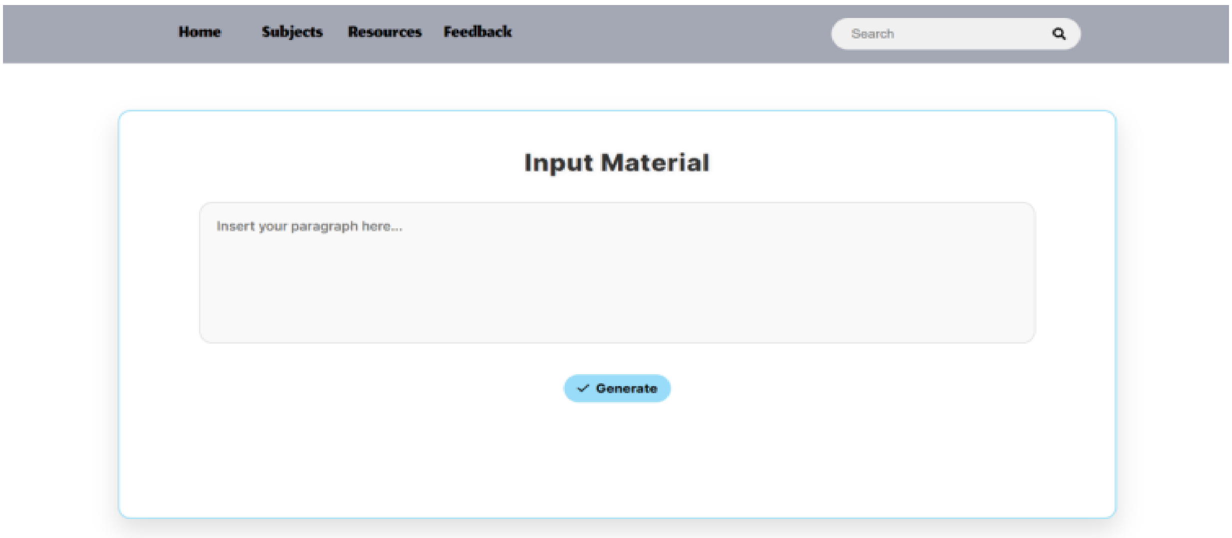


Figure 20: UI of the System 2

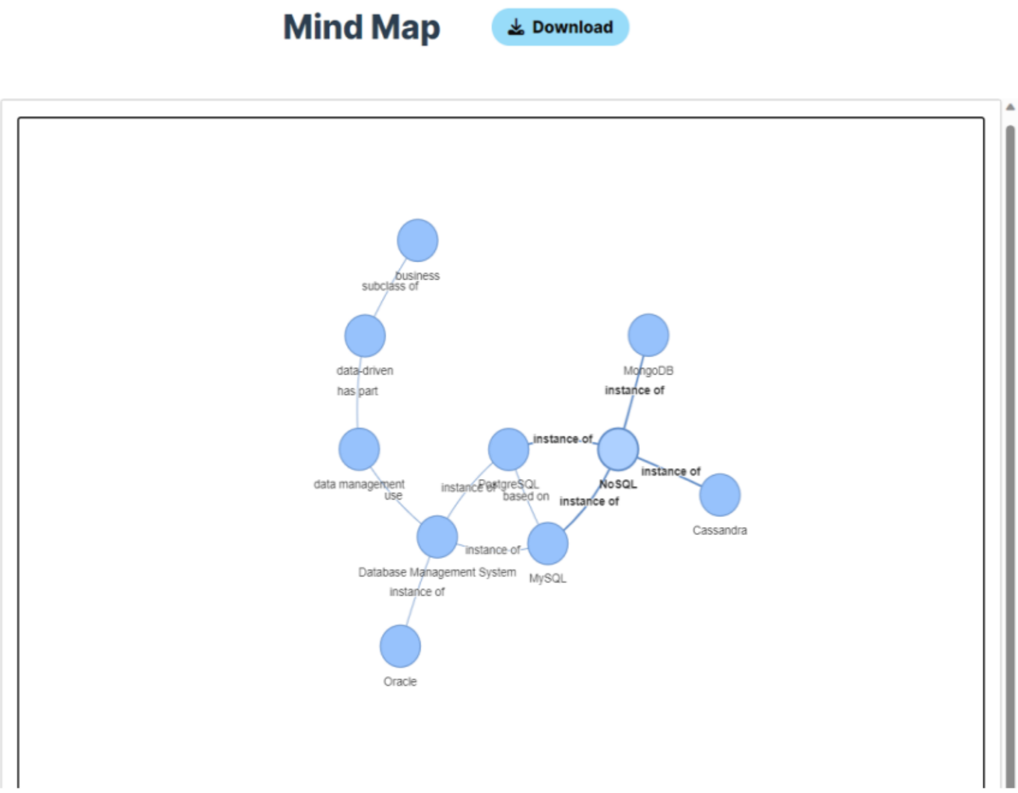


Figure 21: Mind-map UI