Mind Map Generator

UML 501 Machine Learning Project Report

Submitted by:

Prisha Aggarwal (102303758)

Gurmandeep Kaur (102303761)

BE Third Year, COE

Group No:

3C53

Submitted To:

Ms. Manisha Malik

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering Department TIET, Patiala

TABLE OF CONTENTS

S.No.	Sections	Page No.
1.	Introduction	1
2.	Need of the project	2
3.	Problem Statement	3
4.	Objectives and existing approaches	4-5
5.	Dataset Description	6
6.	Methodology / Model Description	7-9
7.	Implementation Details	10-11
8.	Results & Evaluation	12-13
9.	Conclusion	14
10.	Future Work & Limitations	15-16

Introduction

In today’s digital age, vast amounts of information are consumed in the form of long paragraphs, documents, and articles. However, understanding and retaining this information can be challenging when presented in linear text form. Mind maps provide a powerful visual representation of ideas by breaking complex content into structured, interconnected concepts. They improve comprehension, memory, and organization of thoughts.

This project focuses on developing a machine learning–based **Mind-Map Generator** that converts textual paragraphs into meaningful, hierarchical mind maps. By applying natural language processing (NLP) techniques, the model identifies key concepts, extracts relationships, and visually organizes the information in a structured map. This automation simplifies the learning process and enhances the user’s ability to grasp and recall information effectively.

Need of Project

1. **Simplifying Complex Information:**
   Students and professionals often deal with lengthy paragraphs or dense textual data. Converting this manually into mind maps is time-consuming and requires cognitive effort. An automated tool reduces this burden.

2. **Enhancing Learning and Retention:**
   Mind maps are proven to improve understanding and long-term memory. Automatically generating them helps users learn faster and retain key ideas more efficiently.

3. **Saving Time and Effort:**
   Manually creating mind maps can take considerable time. An ML-based generator quickly summarizes and visualizes the core content, increasing productivity.

4. **Supporting Diverse Users:**
   This tool is useful for students, researchers, educators, and professionals who need to organize or revise content quickly.

5. **Bridging the Gap Between Text and Visual Learning:**
   Different individuals learn better through visual formats. This system supports visual learners by transforming written text into intuitive diagrams.

Problem Statement

Despite the availability of various digital reading tools, most systems still present information in plain text form, making it difficult for users to quickly identify important points and understand how different ideas are related. Creating mind maps manually requires careful reading, extraction of core concepts, and organizing them into a logical structure — a process that is slow, repetitive, and prone to human error.

Users often struggle to quickly understand and organize long textual content because existing tools either summarize text in plain form or rely heavily on manual input for mind-map creation. There is no system that can take a raw paragraph and automatically identify the main idea, extract supporting concepts, and organize them into a meaningful structure.

Thus, the problem is to **design and develop a machine learning–based solution that can interpret natural language, extract key concepts, detect their relationships, and generate a structured mind map automatically**, reducing human effort and improving the efficiency of learning and information processing.

Objectives

1. **To extract meaningful ideas from raw text**
   Develop an NLP-based system capable of analysing any paragraph and identifying key concepts, themes, and supporting points.

2. **To identify the most important (root) concept**
   Implement algorithms that detect the central topic around which the entire mind map will be structured.

3. **To generate logical connections between ideas**
   Determine relationships among extracted concepts and organize them hierarchically to form a coherent mind map structure.

4. **To automate mind-map creation using Machine Learning**
   Convert unstructured text into a structured mind map without any manual intervention.

5. **To provide fast, accurate, and readable output**
   Ensure that the generated mind map is concise, meaningful, and produced efficiently using optimized NLP models.

6. **To visually display the mind map on a user-friendly frontend UI**
   Integrate the backend ML model with an interactive interface where users can view, explore, and download the generated mind map.

7. **To enhance understanding and retention**
   Present information in a visual format that helps users better grasp and recall the given text.

Existing approaches

1. **Reading and summarizing manually**
   Users read the entire paragraph, underline or note down important concepts, and then manually decide the hierarchy of information.

2. **Using mind-mapping software (manual input needed)**
   Tools like XMind, MindMeister, and Coggle help create mind maps but still require the user to input topics and subtopics manually. They only assist in drawing, not generating content.

3. **Traditional summarization tools**
   Some text-summarization tools condense paragraphs but do not organize information into visual or hierarchical structures.

4. **Dependence on Paid APIs**
   Most existing solutions rely on commercial APIs or subscription-based models, making them inaccessible to students or developers who want a free and open system. Their core logic is closed-source and cannot be modified or optimized.

Dataset Description

Unlike traditional machine learning projects that rely on fixed, pre-collected datasets, this system does **not use any external dataset**. Instead, it is designed to work dynamically with **user-provided text**, which serves as real-time input data.

Whenever a user enters a paragraph, the model treats it as fresh data and performs concept extraction, semantic understanding, and relationship detection on the spot.

To understand the meaning and context of the text, the system uses **pretrained Sentence-BERT (SBERT) embeddings**, which provide rich semantic representations without requiring custom training data. These embeddings allow the model to accurately measure similarity between sentences, identify core ideas, and cluster supporting concepts.

Thus, the project relies entirely on:

- **User-generated input text (real-time dataset)**
- **Pretrained Sentence-BERT embeddings for semantic understanding**

This approach ensures high flexibility, eliminates dataset collection overhead, and allows the system to generalize to *any* type of text provided by the user.

Model Description

This project follows a structured NLP pipeline that converts raw text into a visual mind map. The system combines transformer-based semantic understanding with custom logic for concept extraction and hierarchical organization.

7.1 ML Model Used

SentenceTransformer: *all-MiniLM-L6-v2*

The system uses the **all-MiniLM-L6-v2** model from SentenceTransformers, a lightweight and efficient transformer architecture designed specifically for generating high-quality semantic embeddings.

Key advantages of this model include:

- **Low computational cost** – suitable for real-time inference.
- **Strong semantic understanding** – captures contextual meaning between sentences.
- **Compact architecture** – only ~22M parameters, making it fast and deployable on CPU.
- **Pretrained on large textual corpora** – allows the model to generalize to any user-provided text.

The model outputs a dense vector embedding for each sentence, enabling similarity calculations and concept clustering.

7.2 Pipeline

The transformation from raw text to a structured mind map follows these steps:

1. Input Text Acquisition
The user enters any paragraph or textual content through the frontend interface. This text serves as real-time input to the backend ML pipeline.

2. Sentence Splitting
The input paragraph is broken down into **individual sentences or meaningful text segments** using Python-based NLP preprocessing.

This step ensures each sentence becomes a potential node in the mind map.

3. Transformer Embedding Generation
Each sentence is passed through the Sentence-BERT model to generate **semantic embeddings**:

- These embeddings represent meaning in a numerical form.
- They allow the system to measure similarity between sentences.

Higher similarity indicates closer conceptual relationships.

4. Root Sentence Identification
To determine the **main topic**:

- A semantic centrality score is computed for each sentence.
- The sentence that is most semantically connected to all others is selected as the **root node** (main idea).

This ensures the mind map centers around the most important concept.

5. Child Sentence Extraction
All remaining sentences are compared with the root embedding:

- Sentences closely related to the root become **direct child nodes**.
- Additional clustering or similarity-based grouping is performed to maintain logical structure.

This forms the hierarchical backbone of the mind map.

6. Mind Map Structure Creation
Using custom Python logic:

- Nodes are created for each extracted sentence.
- Edges (connections) are generated based on semantic similarity scores.
- A final structured tree-like JSON format is produced for frontend visualization.

This step translates the ML insights into a graph data structure.

7. Visualization in React using ReactFlow
The structured JSON graph is sent to the frontend, where:

- **ReactFlow** renders the nodes and edges dynamically.
- A clean, interactive, draggable mind map is displayed.
- Users can expand, move, or analyze concepts visually.

This ensures an intuitive and user-friendly visualization layer.

7.3 Key Technologies

Backend

- **Python + Flask**
  Used to build the API that processes text, runs the ML model, and constructs the mind map structure.
- **Sentence-BERT (all-MiniLM-L6-v2)**
  Provides semantic embeddings for understanding the meaning of sentences.
- **Flask-CORS**
  Enables secure communication between the backend API and the React frontend.

Frontend

- **React.js**
  Creates the user interface for input and interaction.
- **ReactFlow**
  A powerful library used for node-based graph visualization, enabling interactive mind maps.

Implementation Details

The system is implemented as a full-stack application consisting of a Python-based backend for text processing and a React-based frontend for mind map visualization. Each component plays a crucial role in ensuring smooth conversion of text into an interactive graphical structure.

8.1 Backend

The backend is implemented using **Flask**, a lightweight Python web framework. Its responsibilities include:

- Receiving the input text sent from the frontend.
- Applying the NLP pipeline and transformer model.
- Constructing a structured mind map representation in JSON format.
- Returning the processed data through REST API endpoints.

Flask's simplicity and flexibility make it ideal for deploying ML-powered APIs.

8.2 Model Logic

At the core of the backend lies the Machine Learning logic powered by **Sentence-BERT (all-MiniLM-L6-v2)**. The model:

- Converts each sentence into a semantic embedding.
- Computes similarity scores to determine semantic closeness.
- Calculates centrality scores to identify the root (most important) sentence.
- Groups related sentences to form child nodes in the mind map.

Custom Python logic is layered on top of the embeddings to:

- Rank sentences,
- Build hierarchical relationships,
- And generate the final nodes and edges.

This combination of pretrained embeddings and custom algorithms enables accurate extraction of ideas without needing a pre-collected dataset.

8.3 Frontend

The frontend is built using **React**, providing an interactive and user-friendly interface. Its main tasks include:

- Accepting the user's input text.

- Sending it to the backend API for processing.
- Rendering the returned mind map using **ReactFlow**, a library designed for dynamic node-based visualizations.

ReactFlow enables features like:
- Draggable nodes
- Zooming and panning
- Editable graph layouts
- Smooth edge connections

This makes the mind map easy to explore and visually intuitive.

8.4 Communication Layer

Interaction between the React frontend and the Flask backend occurs through a **JSON-based REST API**. Key characteristics include:

- The frontend sends a POST request with user input in JSON format.
- The backend processes the request and responds with a structured JSON representation of nodes and edges.
- Cross-origin communication is enabled using **Flask-CORS**, ensuring seamless and secure data transfer.

This decoupled communication model makes the system scalable and modular, allowing independent updates to frontend or backend components.

Results and Evaluation

The developed system demonstrates strong performance in converting raw text into meaningful and visually structured mind maps. The combination of transformer-based embeddings and custom logic enables accurate identification of key ideas and their relationships.

9.1 Functionality Achievements

Accurate Extraction of Main Idea and Sub-Ideas
The system consistently identifies the **central theme** of the paragraph and organizes the remaining sentences into **related subtopics**, forming a clear hierarchical structure.
This ensures that the mind map captures the essence of the text effectively.

Works Across Text of Varying Lengths
Whether the input is a **short definition**, a **medium-length explanation**, or a **long paragraph**, the model adapts well and produces coherent mind maps. There is no dependency on paragraph size, making the system versatile.

Handles Diverse Types of Academic and Informational Text
The system performs reliably across multiple text formats, including:

- academic explanations
- lecture notes
- definitions
- conceptual descriptions
- informational paragraphs
- topic summaries

This makes the tool suitable for students, educators, and researchers.

9.2 Output example

Input Paragraph:
"Photosynthesis is the process by which green plants and some other organisms use sunlight to synthesize foods with the help of chlorophyll. It involves the conversion of carbon dioxide and water into glucose and oxygen. Photosynthesis is essential for life on Earth as it provides oxygen for respiration and organic matter for food chains. Factors affecting photosynthesis include light intensity, carbon dioxide concentration, temperature, and water availability."

Generated Mind Map (Root–Child Structure):

```
Root Node: Photosynthesis
   ├── Definition: Process by which green plants and some organisms
synthesize food using sunlight and chlorophyll.
   ├── Process: Conversion of carbon dioxide and water into glucose and
oxygen.
   ├── Importance: Provides oxygen for respiration and organic matter for
food chains.
   └── Factors Affecting Photosynthesis:
         ├── Light intensity
         ├── Carbon dioxide concentration
         ├── Temperature
         └── Water availability
```

Here is the MindMap:

Conclusion

The project successfully demonstrates the use of **Machine Learning**, particularly **transformer-based NLP models**, to convert unstructured text into structured, visual mind maps. By automatically identifying the main idea, extracting related concepts, and establishing their hierarchical relationships, the system eliminates the need for manual mind-map creation.

Key outcomes of the project include:

- **Reduction in manual effort:** Users no longer need to read, summarize, and organize text themselves.
- **Improved comprehension:** Hierarchical visualization helps users quickly grasp the central idea and supporting points.
- **Enhanced learning and revision:** The generated mind maps serve as effective study aids, allowing students to retain information more efficiently.
- **Flexibility:** The system works with diverse types of text, from definitions and notes to academic explanations and paragraphs of varying length.

Overall, this project highlights the potential of integrating **transformer-based semantic understanding** with **interactive visualization**, providing a practical tool that bridges the gap between textual content and visual learning.

Limitations & Future Work

10.1 Limitations

While the system successfully generates structured mind maps from user-provided text, certain limitations exist:

1. **Single-Level Hierarchy**
   Currently, the mind map is limited to a **one-level hierarchy** (root → children). Deeper or nested relationships between concepts are not fully represented.

2. **Complex Relationships**
   The model may not always capture intricate or abstract relationships between ideas, particularly when the text is highly technical or metaphorical.

3. **Handling Long Paragraphs**
   For very long paragraphs, semantic clustering may need improvement to avoid loosely connected or redundant nodes.

10.2 Future Work

Several enhancements can be implemented to improve the system's capability, scalability, and usability:

1. **Multi-Level Hierarchical Mind Maps**
   Extend the system to support **multi-level structures**, allowing sub-children and deeper connections for more detailed visualization.

2. **Support for Diverse Input Types**
   Add the ability to process **PDF documents**, **audio-to-text inputs**, and **image-based text** using OCR, expanding usability beyond plain text.

3. **Advanced Clustering of Similar Ideas**
   Use algorithms like **KMeans** or other clustering methods to group similar sentences, improving organization and reducing redundancy.

4. **Relation Extraction Using Dependency Parsing**
   Implement **dependency parsing** and advanced NLP techniques to capture more complex semantic relationships between ideas.

5. **Domain-Specific Fine-Tuning**
   Allow **local fine-tuning of transformer models** for specific domains (e.g., medical, legal, technical) to enhance accuracy and relevance of the generated mind maps.

By addressing these limitations and implementing the proposed future enhancements, the system can evolve into a **more robust, flexible, and intelligent mind-map generation tool**, suitable for diverse educational, professional, and research applications.