

# **SUBJECTIVE ANSWER EVALUATION**

*A Project report submitted  
In partial fulfillment of requirements  
for the award of degree of*

## **BACHELOR OF TECHNOLOGY**

**IN**

## **INFORMATION TECHNOLOGY**

**By**

<b>G. VARSHA</b>	<b>(Reg. No:20131A1218)</b>
<b>K.VENKATA SAI KRISHNA</b>	<b>(Reg. No:20131A1225)</b>
<b>M. S. VAIBHAV VENKAT</b>	<b>(Reg. No:20131A1230)</b>
<b>P. LAXMI PRANEELA</b>	<b>(Reg. No:20131A1237)</b>



Under the esteemed guidance of

**Mr. P. PRAVEEN KUMAR**

**(Assistant Professor)**

Department of Information Technology  
**GAYATRI VIDYA PARISHAD COLLEGE OF  
ENGINEERING  
(AUTONOMOUS)**  
(Affiliated to JNTU-K, Kakinada)  
**VISAKHAPATNAM**  
**2023-2024**

# **GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING**

**(Autonomous)**

**Visakhapatnam**



## **CERTIFICATE**

This report on “**SUBJECTIVE ANSWER EVALUATION**”

is a Bonafide record of the main project work submitted by

**G. VARSHA** (Reg.No:20131A1218)

**K.VENKATA SAI KRISHNA** (Reg.No:20131A1225)

**M. SAI VAIBHAV VENKAT** (Reg.No:20131A1230)

**P. LAXMI PRANEELA** (Reg.No:20131A1237)

in their VIII semester in fulfilment of the requirements for the Award of the degree of

**Bachelor of Technology**

**In**

**Information Technology**

during the academic year 2023-2024

**Mr. P. Praveen Kumar**

Assistant Professor

Dept of I.T

Project Guide

**Dr. R. V. V. Murali Krishna**

Associate Professor &

Head of Department

Dept of I.T

**EXTERNAL EXAMINER**

## **DECLARATION**

We, the under signed ,hereby duly declare that this main project entitled “**SUBJECTIVE ANSWER EVALUATION**” is a bonafide work done by us and submitted to the **Department of Information Technology, Gayatri Vidya Parishad College of Engineering (A), Visakhapatnam**, in partial fulfilment for the Award of the Degree of the B. Tech is of our own and has not been submitted to any other university or published any time before.

**Place: Visakhapatnam**

**G. Varsha**

**(20131A1218)**

**Date:**

**K. Venkata Sai Krishna**

**(20131A1225)**

**M. S. Vaibhav Venkat**

**(20131A1230)**

**P. Laxmi Praneela**

**(20131A1237)**

## ACKNOWLEDGEMENT

We would like to express our deep sense of gratitude to our esteemed institute **Gayatri Vidya Parishad College of Engineering (Autonomous)**, which has provided us an opportunity to fulfil our cherished desire.

We express our deep sense of gratitude to **Dr. R.V.V. MURALI KRISHNA ASSOCIATE PROFESSOR and Head of the Department of Information Technology, Gayatri Vidya Parishad College of Engineering(A)** for giving us an opportunity to do the project in college.

We express our sincere thanks to our Principal **Prof. A. B. KOTESWARA RAO** for his encouragement to us during the course of this project, giving us a chance to explore and learn new technologies in the form of the Main Project.

We express our profound gratitude and our deep indebtedness to our guide **Mr. P. PRAVEEN KUMAR, ASSISTANT PROFESSOR** whose valuable suggestions, guidance, along with comprehensive assistance helped us a lot in realizing my present project.

We also thank our Coordinator **Mr. B. SRINU** for the kind suggestions and guidance for the successful completion of our project work.

Finally, we would also like to thank all the members of the teaching and non-teaching staff of the Information Technology Department for all their support in the completion of our project.

<b>G. VARSHA</b>	<b>(Reg No:20131A1218)</b>
<b>K. VENKATA SAI KRISHNA</b>	<b>(Reg No:20131A1225)</b>
<b>M. SAI VAIBHAV VENKAT</b>	<b>(Reg No:20131A1230)</b>
<b>P. LAXMI PRANEELA</b>	<b>(Reg No:20131A1237)</b>

## **ABSTRACT**

Assessing subjective papers manually is both challenging and tedious, primarily due to the complexities involved in comprehending and interpreting the data. Previous attempts to automate the scoring of students' answers using computer science have predominantly relied on basic word counting methods. This project proposes an innovative approach that harnesses advanced techniques from machine learning and natural language processing, including Word Mover's Distance (WMD) and Term Frequency-Inverse Document Frequency (TF-IDF), to autonomously evaluate descriptive answers. The evaluation process involves the utilization of solution statements and keywords, and a machine learning model is trained to predict answer grades. Notably, the study incorporates additional measures such as Jaccard Similarity, Smooth Inverse Frequency (SIF) embedding similarity, METEOR score, and BLEU score to enrich the assessment process. Results indicate that WMD outperforms other metrics, highlighting the effectiveness of incorporating diverse measures. With sufficient training, the machine learning model has the potential to operate as a standalone solution, streamlining the assessment of subjective papers and reducing the burden of manual grading.

# INDEX

<b>1</b>	<b>INTRODUCTION.....</b>	<b>9</b>
1.1	OBJECTIVE.....	10
1.2	ABOUT THE PROJECT .....	11
1.3	SCOPE .....	11
1.4	PURPOSE .....	12
<b>2</b>	<b>REQUIREMENT SPECIFICATION .....</b>	<b>12</b>
2.1	FUNCTIONAL REQUIREMENTS .....	12
2.2	NON-FUNCTIONAL REQUIREMENTS .....	13
2.3	SOFTWARE REQUIREMENTS .....	13
2.3.1	OPERATING SYSTEM.....	14
2.3.2	SDK .....	14
2.4	HARDWARE REQUIREMENTS:.....	14
<b>3</b>	<b>ANALYSIS.....</b>	<b>15</b>
3.1	EXISTING SYSTEM .....	15
3.2	PROPOSED SYSTEM .....	15
<b>4</b>	<b>SOFTWARE DESCRIPTION .....</b>	<b>16</b>
4.1	TRANSFORMERS .....	16
4.2	GENSIM .....	18
4.3	WORD2VEC.....	19

4.4	TEXTGEARS.....	20
4.5	FAISS.....	21
4.6	NLTK.....	21
4.7	DJANGO.....	22
<b>5</b>	<b>PROJECT DESCRIPTION.....</b>	<b>22</b>
5.1	PROJECT DEFINITION .....	22
5.1.1	PREPROCESSING AND EMBEDDING .....	22
5.1.2	RETRIEVAL OF RELEVANT ANSWER.....	23
5.1.3	STUDENT RESPONSE EVALUATION .....	23
5.1.4	PENALIZATION FOR GRAMMATICAL ERRORS.....	23
5.1.5	FINAL SCORE CALCULATION:.....	23
5.1.6	OUTPUT.....	23
5.2	METRIC DESCRIPTION.....	24
5.2.1	JACCARD SIMILARITY .....	24
5.2.2	WMD SIMILARITY .....	24
5.2.3	BLUE SCORE.....	24
5.2.4	METEOR SCORE.....	25
5.2.5	SIF EMBEDDING SIMILARITY .....	26
5.2.6	SOFT COSINE SIMILARITY.....	27
<b>6</b>	<b>SYSTEM DESIGN .....</b>	<b>28</b>
6.1	BUILDING BLOCKS OF UML.....	28

6.2	UML DIAGRAMS .....	30
6.2.1	CLASS DIAGRAM.....	30
6.2.2	SEQUENCE DIAGRAM.....	31
6.2.3	ACTIVITY DIAGRAM.....	32
<b>7</b>	<b>DEVELOPMENT .....</b>	<b>33</b>
7.1	SAMPLE CODE .....	33
7.2	OUTPUT.....	37
7.2.1	HOME PAGE .....	37
7.2.2	About Section.....	37
7.2.3	Technology Section .....	37
7.2.4	Input 1 .....	38
7.2.5	Output 1 .....	38
7.2.6	Input 2.....	38
7.2.7	Output 2 .....	39
7.3	Testing and Validation.....	39
<b>8</b>	<b>CONCLUSION .....</b>	<b>40</b>
<b>9</b>	<b>FUTURE SCOPE.....</b>	<b>41</b>
<b>10</b>	<b>REFERENCES.....</b>	<b>42</b>



# 1 INTRODUCTION

Assessing subjective papers manually remains a formidable challenge, characterized by the intricate process of comprehending and interpreting complex data. Traditional evaluation methods, burdened by labor-intensity and susceptibility to subjective biases, have led to a persistent need for automated solutions. Previous attempts in this regard have often relied on rudimentary word counting methodologies, falling short of providing nuanced evaluations. This project addresses the limitations of existing approaches by recognizing the inherent difficulties in understanding and embracing the complexity of subjective responses. While manual assessment is established, it lacks scalability and objectivity. Automated systems, though attempting to mitigate the burden, often employ simplistic methods. Our innovative approach aims to transcend these limitations. By leveraging advanced techniques from machine learning and natural language processing, such as NLTK, POS Tagging, Tokenization, and Term Frequency-Inverse Document Frequency (TF-IDF), this project introduces a sophisticated framework for autonomously evaluating descriptive answers. Incorporating measures like Word Mover's Distance (WMD), Jaccard Similarity, Smooth Inverse Frequency (SIF) embedding similarity, METEOR score, and BLEU score, our methodology enriches the assessment process. Notably, our results showcase the effectiveness of diverse metrics, particularly highlighting the superiority of WMD over others. With sufficient training, our machine learning model holds the potential to operate as a standalone solution, streamlining the assessment of subjective papers and reducing the burden of manual grading. This project underscores the necessity for smarter systems in evaluating subjective papers, striving towards better and more accurate assessment methods. Certainly, let's delve deeper into the innovative aspects and significance of this project.

- 1. Sophisticated Evaluation Techniques:** One of the key contributions of this project lies in its utilization of advanced techniques from machine learning and natural language processing (NLP). Rather than relying solely on basic word counting methods, the project incorporates a diverse set of methodologies such as Word Mover's Distance (WMD), Term Frequency-Inverse Document Frequency (TF-IDF), Jaccard Similarity, Smooth Inverse Frequency (SIF) embedding similarity, METEOR score, and BLEU score. This multi-faceted approach allows for a more comprehensive evaluation of subjective papers, capturing nuances that traditional methods might overlook.

2. **Autonomous Evaluation Framework:** By harnessing these advanced techniques, the project aims to develop an autonomous evaluation framework for descriptive answers. This framework leverages machine learning models trained on solution statements and keywords to predict answer grades. This not only reduces the burden of manual grading but also ensures a more objective and consistent assessment process.
3. **Comparison and Validation of Metrics:** Through rigorous experimentation and analysis, the project compares the performance of various evaluation metrics. Notably, the results highlight the superiority of Word Mover's Distance (WMD) over other metrics such as cosine similarity. This empirical validation adds credibility to the proposed framework and guides future research directions.
4. **Scalability and Generalization:** Another important aspect of this project is its potential for scalability and generalization. With sufficient training and refinement, the machine learning model could evolve into a standalone solution applicable across diverse domains and educational settings. This scalability is crucial for addressing the growing demand for efficient and objective evaluation methods in education.
5. **Implications for Education and Beyond :** Beyond its immediate application in educational assessment, the project's findings have broader implications for the fields of machine learning, natural language processing, and cognitive science. By advancing our understanding of how to effectively evaluate subjective responses, the project contributes to the development of smarter systems capable of handling complex data in various contexts.

In summary, this project represents a significant step forward in the automation of subjective paper evaluation. By integrating cutting-edge techniques from machine learning and NLP, it offers a more sophisticated and accurate alternative to traditional methods, with far-reaching implications for education and beyond.

## 1.1 OBJECTIVE

The objective of the above project is to develop an innovative system for autonomously evaluating subjective papers using advanced techniques from machine learning and natural language processing. The goal is to overcome the challenges and limitations of manual assessment by leveraging methods such as Word Mover's Distance (WMD), Term Frequency-Inverse Document Frequency (TF-IDF), and other similarity

metrics to assess descriptive answers. By incorporating these techniques, the project aims to provide a more accurate, efficient, and scalable solution for grading subjective responses. Additionally, the project seeks to explore the potential of machine learning models to predict answer grades autonomously, ultimately reducing the burden of manual grading and improving the overall evaluation process.

## 1.2 ABOUT THE PROJECT

**1. Overcoming Limitations:** Addressing the limitations of existing manual assessment methods and simplistic automated approaches, the project aims to develop a framework capable of comprehensively evaluating subjective responses.

**2. Utilizing Advanced Techniques:** Leveraging advanced techniques from machine learning and natural language processing, such as Word Mover's Distance (WMD), Soft-Cosine Similarity, SIM Embedding similarity, BLEU Score, Meteor Score and various similarity measures, to enhance the accuracy and sophistication of the evaluation process.

**3. Automation and Objectivity:** Developing a machine learning model trained on solution statements and keywords to autonomously predict answer grades, thereby reducing the burden of manual grading and ensuring a more objective assessment process.

**4. Comparison and Validation:** Comparing and validating different evaluation metrics to identify the most effective approaches for assessing subjective papers. This involves empirical analysis to assess the performance of metrics like WMD, Jaccard Similarity, METEOR score, and BLEU score.

**5. Scalability and Generalization:** Designing the framework to be scalable and generalizable across various educational domains and settings, with the potential to evolve into a standalone solution applicable to diverse contexts.

**6. Contributing to Education and Research:** Contributing to the advancement of educational assessment practices and the fields of machine learning, natural language processing, and cognitive science by developing smarter systems capable of handling complex data and improving evaluation methods.

## 1.3 SCOPE

The scope of the project encompasses the development and implementation of an advanced framework for the automated evaluation of subjective papers, particularly

descriptive answers. This involves integrating sophisticated techniques from machine learning and natural language processing to pre-process text, extract relevant features, and train predictive models. The framework will incorporate multiple evaluation metrics, including Word Mover's Distance, Jaccard Similarity, METEOR score, and BLEU score, to ensure comprehensive assessment. Software implementation will enable educators and researchers to easily utilize the framework, while validation and testing will assess its accuracy and reliability against human evaluations. The project will culminate in a detailed report, documenting methodologies, experimental results, and recommendations for future enhancements and applications.

## **1.4 PURPOSE**

The purpose of the project is to revolutionize the evaluation of subjective papers, specifically descriptive answers, by introducing a sophisticated automated framework. This project is essential due to the inherent complexities of subjective content, which often eludes traditional grading methods and leads to subjective biases. The necessity for such an endeavor is underscored by the limitations of manual grading, including its labor-intensive nature, susceptibility to bias, and inconsistency among graders. By harnessing advanced techniques from machine learning and natural language processing, the project aims to streamline the assessment process, ensuring scalability, objectivity, and accuracy. Furthermore, the project seeks to enhance the educational experience by providing timely and constructive feedback to students, thereby contributing to their overall learning and growth. Ultimately, the project endeavors to bridge the gap between traditional assessment methods and the demands of modern education, paving the way for more efficient and effective evaluation practices.

# **2 REQUIREMENT SPECIFICATION**

## **2.1 FUNCTIONAL REQUIREMENTS**

- 1. Document upload and preprocessing module:** The Document Upload and Preprocessing Module permits users to upload subjective papers, extracting paragraphs for analysis. Employing sentence transformers, it converts these paragraphs into vector embeddings, capturing semantic meanings for numerical

representation. By structuring the input data into vectors, it facilitates nuanced evaluation within the automated framework, enhancing the accuracy and efficiency of subjective paper assessment.

2. **Data Storage:** The Data Storage Module securely stores uploaded documents, extracted paragraphs, and vector embeddings generated by sentence transformers in vector databases using faiss library. It ensures data integrity and accessibility, facilitating efficient retrieval and manipulation during the evaluation process.
3. **Question Submission:** Users are required to submit questions for evaluation.
4. **Student Answer Submission:** A student answer paragraph has to be submitted.
5. **Evaluation using various metrics:** The evaluation process utilizes various metrics to assess subjective papers comprehensively. These metrics include Word Mover's Distance (WMD), Jaccard Similarity, Smooth Inverse Frequency (SIF) embedding similarity, METEOR score, and BLEU score.
6. **Scoring Mechanism:** The weighted averages of all the metrics are combined to calculate the score from which deduction is done for grammatical errors and spelling mistakes calculated using textgears API.
7. **Web Application:** Enable functionalities for user interaction, result display, and feedback.

## 2.2 NON-FUNCTIONAL REQUIREMENTS

Non-Functional Requirements are the constraints or the requirements imposed on the system. They specify the quality attribute of the software. Non-Functional Requirements deal with issues like **scalability, maintainability, performance, portability, security, reliability**, and many more. Non-Functional Requirements address vital issues of quality for software system. It includes below things: Capacity, Availability and Performance etc.

## 2.3 SOFTWARE REQUIREMENTS

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view.

### **2.3.1 OPERATING SYSTEM**

Windows is a graphical operating system developed by Microsoft. It allows users to view and store files, run the software, play games, watch videos, and provides a way to connect to the internet. It was released for both home computing and professional works.

### **2.3.2 SDK**

SDK stands for software development kit or devkit for short. It's a set of software tools and programs used by developers to create applications for specific platforms. SDK tools will include a range of things, including libraries, documentation, code samples, processes, and guides those developers can use and integrate into their own apps. SDKs are designed to be used for specific platforms or programming languages. Some of SDK's used in this project are stated below:

1. NLTK
2. Django
3. Transformers
4. Scikit-Learn
5. Gensim
6. Textgears API
7. Python

## **2.4 HARDWARE REQUIREMENTS:**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements. The Hardware Interfaces Required are:

1. Ram: Minimum 4GB or higher
2. GPU or CSU: 4GB
3. SSD: 64GB
4. Processor: Intel i5 10th Gen or Ryzen 5 with Octa core.

## **3 ANALYSIS**

### **3.1 EXISTING SYSTEM**

Current assessment methods for subjective papers predominantly involve manual grading or basic automated approaches. Manual grading is time-consuming, labor-intensive, and susceptible to subjective biases, leading to inconsistencies in scoring. On the automated side, many systems use simplistic word counting methods that struggle to capture the complexities of subjective answers, resulting in a simplified evaluation process. These approaches often lack the sophistication needed to understand the nuances of language, making it challenging to provide meaningful feedback. Automated systems relying on basic algorithms may struggle with context, creativity, and a deep understanding of well-crafted responses. In essence, existing systems face challenges in terms of consistency, subjectivity, and the ability to comprehensively evaluate the intricate nature of subjective answers. Recognizing these limitations emphasizes the need for a more intelligent and nuanced system, which is the focal point of our innovative approach.

### **3.2 PROPOSED SYSTEM**

The innovative automated assessment system, blending advanced machine learning and natural language processing, revolutionizes education. Utilizing techniques like Word Mover's Distance and cosine similarity, it streamlines subjective paper grading, providing timely and insightful feedback. This dynamic tool enhances language learning, refines research paper reviews, and transforms the online learning experience for educators and students alike.

Our automated assessment system integrates machine learning and natural language processing. Our engineering solution streamlines subjective paper grading, provides timely feedback, and enriches the learning experience, contributing to educational advancement by Leveraging Word Mover's Distance, cosine similarity, Multinomial Naive Bayes, and TF-IDF.

## 4 SOFTWARE DESCRIPTION

### 4.1 TRANSFORMERS

In the context of the aforementioned project, the utilization of transformers, particularly those designed for sentence embeddings, plays a pivotal role in the evaluation of subjective papers. Transformers, a type of deep learning architecture, have revolutionized natural language processing tasks due to their ability to capture complex contextual relationships within text. Specifically, in the project, sentence transformers are employed to convert extracted paragraphs from subjective papers into dense vector representations. These vector embeddings encode semantic information, enabling a more nuanced understanding of textual content. By leveraging transformer-based models, such as BERT (Bidirectional Encoder Representations from Transformers) the evaluation framework can effectively capture the subtle nuances and semantic similarities between different paragraphs, thereby enhancing the accuracy and sophistication of the automated evaluation process. Moreover, transformers offer scalability and adaptability, allowing the framework to handle a wide range of subjective papers and evaluation tasks with improved efficiency and effectiveness. Overall, the integration of transformers represents a key advancement in the project, enabling more robust and context-aware evaluation of subjective answers.

#### 1. SENTENCE-TRANSFORMERS:

. Specifically, in the project, sentence transformers are employed to convert extracted paragraphs from subjective papers into dense vector representations. These vector embeddings encode semantic information, enabling a more nuanced understanding of textual content. By leveraging transformer-based models, such as BERT (Bidirectional Encoder Representations from Transformers) or RoBERTa (Robustly Optimized BERT Approach), the evaluation framework can effectively capture the subtle nuances and semantic similarities between different paragraphs, thereby enhancing the accuracy and sophistication of the automated evaluation process. Moreover, transformers offer scalability and adaptability, allowing the framework to handle a wide range of subjective papers and evaluation tasks with improved efficiency and effectiveness. Overall, the integration of transformers represents a key advancement in the project, enabling more robust and context-aware evaluation of subjective answers



## **2. BERT MODEL:**

BERT, which stands for Bidirectional Encoder Representations from Transformers, is a ground breaking model in natural language processing (NLP) developed by Google. It belongs to the family of transformer-based architectures and has significantly advanced various NLP tasks, including text classification, named entity recognition, and question answering. In the project context, BERT serves a critical role within the sentence transformers framework. Initially, the input text undergoes tokenization, breaking it down into individual tokens. These tokens are then processed by the BERT architecture, which leverages bidirectional context during pre-training to capture deep contextual relationships between words in both directions of a sentence. This bidirectional approach enables BERT to understand the context of a word based on its surrounding words, resulting in more accurate representations of word meanings and sentence semantics. Furthermore, BERT is pre-trained on large corpora of text data using unsupervised learning objectives, such as masked language modeling and next sentence prediction, enabling it to learn rich contextual representations of words and sentences. Fine-tuning BERT on specific downstream tasks with task-specific datasets further enhances its performance, making it a versatile and powerful tool for various NLP applications, including the evaluation of subjective papers within the project framework. Overall, BERT has revolutionized the field of NLP by providing state-of-the-art performance on a wide range of tasks and has become a cornerstone in many modern NLP systems and applications.

## **3. SENTENCE-TRANSFORMERS/MULTI-QA-DISTILBERT-COS-V1:**

This is a sentence-transformers model: It maps sentences & paragraphs to a 768dimensional dense vector space and was designed for semantic search. It has been trained on 215M (question, answer) pairs from diverse sources.

Technical Details

In the following some technical details how this model must be used:

Setting	Value
Dimensions	768
Produces normalized embeddings	Yes
Pooling-Method	Mean pooling
Suitable score functions	dot-product ( <code>util.dot_score</code> ), cosine-similarity ( <code>util.cos_sim</code> ), or euclidean distance

Note: When loaded with sentence-transformers, this model produces normalized embeddings with length 1. In that case, dot-product and cosine-similarity are equivalent. dot-product is preferred as it is faster. Euclidean distance is proportional to dot-product and can also be used.

Intended uses

The model is intended to be used for semantic search: It encodes queries / questions and text paragraphs in a dense vector space. It finds relevant documents for the given passages.

Note that there is a limit of 512word pieces: Text longer than that will be truncated. Further note that the model was just trained on input text up to 250word pieces. It might not work well for longer text.

### Training procedure

The full training script is accessible in this current repository: `train_script.py`.

Used the pretrained “**distilbert-base-uncased model**” for training.

The concatenation from multiple datasets is used to fine-tune our model. In total there are about 215M (question, answer) pairs. Each dataset is sampled by given a weighted probability whose configuration is detailed in the `data_config.json` file. The model was trained with Multiple Negatives RankingLoss using Mean-pooling, cosine-similarity as similarity function, and a scale of 20.

## 4.2 GENSIM

Gensim is a Python library renowned for its capabilities in natural language processing (NLP) tasks, particularly focused on topic modeling, document indexing, and similarity retrieval using large-scale textual data. Offering efficient and scalable implementations of

popular NLP algorithms, Gensim facilitates tasks like Latent Semantic Analysis (LSA), Latent Dirichlet Allocation (LDA), and Word2Vec. These algorithms enable users to uncover latent patterns and structures within text data, making it invaluable for tasks such as topic extraction, document clustering, and word embedding generation. Gensim also provides functionality for computing document similarity, allowing for semantic comparison between documents using techniques like cosine similarity. Its memory-friendly implementations support processing of large-scale text corpora, leveraging streaming and incremental processing techniques for efficiency. Gensim's extensibility and intuitive API enable users to develop custom models and pipelines, seamlessly integrating with other Python libraries for enhanced flexibility and usability in various NLP applications and data science workflows. Overall, Gensim stands as a versatile and powerful tool for extracting meaningful insights and knowledge from textual data, empowering users to unlock the full potential of their text mining endeavours.

### **4.3 WORD2VEC**

Word2Vec is a popular technique in natural language processing (NLP) used to learn dense vector representations, or embeddings, of words from large text corpora. Developed by Google researchers, Word2Vec aims to capture the semantic meaning and contextual relationships between words in a continuous vector space. The core idea behind Word2Vec is based on the distributional hypothesis, which posits that words appearing in similar contexts are likely to have similar meanings. Word2Vec consists of two main architectures: Continuous Bag of Words (CBOW) and Skip-gram. In the CBOW architecture, the model predicts the target word based on the context words surrounding it. Conversely, in the Skip-gram architecture, the model predicts the context words given a target word. During training, Word2Vec learns to map words to dense vector representations in such a way that similar words are located close to each other in the vector space. This allows for efficient computation of semantic similarity between words using vector operations, such as cosine similarity. Word2Vec embeddings have found widespread applications in various NLP tasks, including word similarity calculation, text classification, sentiment analysis, and information retrieval. These embeddings capture rich semantic information and can be used as features in downstream machine learning models, enhancing their performance on text-related tasks. Overall, Word2Vec represents a powerful technique for learning distributed representations of words, enabling effective modeling of semantic relationships and contextual similarities in natural language data.

## 4.4 TEXTGEARS

TextGears API allows you to integrate the latest technologies for text analyzing virtually to any product. From simple mobile apps to bulky enterprise developments. The API allows you to check text for all kinds of errors in a flexible manner, determine the readability of the text, evaluate the approximate vocabulary of the author, and much more.

What can the API do?

- Grammar check
- Spellchecking
- Auto correction
- Readability
- Text analysis
- Language detection
- Summarization
- Extracting keywords

**4.4.1 GRAMMAR CHECK:** TextGears offers a comprehensive grammar check feature designed to assist users in identifying and correcting grammatical errors in their written content. This grammar check functionality analyses the text for various grammatical issues, including punctuation errors, subject-verb agreement, sentence structure, verb tense consistency, and other syntactical inaccuracies. It helps users maintain grammatical correctness and coherence in their writing, ensuring that the text adheres to established grammatical rules and conventions. Additionally, TextGears provides detailed suggestions and explanations for each identified error, empowering users to understand and rectify their mistakes effectively. By leveraging advanced linguistic algorithms and natural language processing techniques, TextGears' grammar check feature enhances the quality and professionalism of written content, making it a valuable tool for writers, editors, and professionals striving for grammatical accuracy and clarity in their communications.

**4.4.2 SPELLCHECKING:** TextGears includes a robust spell check feature to assist users in identifying and correcting spelling errors in their written content. This spell check functionality scans the text for misspelled words and provides suggestions for correct replacements. It helps users ensure the accuracy and professionalism of their

writing by highlighting and correcting common spelling mistakes, typos, and errors. Additionally, TextGears offers a comprehensive dictionary and language database to support spell checking in multiple languages, further enhancing its utility for a diverse range of users. Overall, the spell check feature in TextGears contributes to improving the readability and credibility of written content by ensuring correct spelling throughout the text.

## 4.5 FAISS

Faiss, which stands for "Facebook AI Similarity Search," is an open-source library developed by Facebook AI Research (FAIR) for efficient similarity search and clustering of high-dimensional data. It is particularly well-suited for applications involving large-scale datasets, such as image retrieval, text search, and recommendation systems. The key feature of Faiss is its implementation of state-of-the-art algorithms for approximate nearest neighbour (ANN) search, which enables fast and scalable similarity search in high-dimensional spaces. Faiss supports various ANN algorithms, including the popular Locality Sensitive Hashing (LSH) methods, as well as more advanced techniques like Product Quantization (PQ) and Hierarchical Navigable Small World (HNSW) graphs. Faiss provides a user-friendly interface for building and querying indexes of high-dimensional vectors, making it easy to integrate into existing machine learning pipelines and applications. It offers both CPU and GPU implementations for efficient computation, allowing users to leverage the computing resources that best suit their needs. Additionally, Faiss supports advanced functionalities such as indexing of billion-scale datasets, real-time updates of indexes, and distributed computing for large-scale deployments. It also includes tools for evaluating the performance of ANN algorithms and optimizing index parameters for specific use cases.

Overall, Faiss is a powerful and versatile library for similarity search and clustering, offering efficient solutions for handling high-dimensional data and enabling a wide range of applications in machine learning, information retrieval, and data analysis.

## 4.6 NLTK

NLTK, short for Natural Language Toolkit, is a robust Python library designed to facilitate various natural language processing (NLP) tasks. Developed by researchers at the University of Pennsylvania, NLTK provides a comprehensive suite of tools and resources for working with human language data. Its functionalities span a wide range of NLP tasks, including tokenization, stemming, lemmatization, part-of-speech tagging, parsing, and

semantic analysis. NLTK's extensive collection of text processing modules enables users to preprocess and analyse textual data effectively, while its support for part-of-speech tagging and parsing allows for deeper linguistic analysis and syntactic understanding. Additionally, NLTK includes a wealth of corpora and lexical resources, making it a valuable asset for both academic research and practical applications in fields such as information retrieval, sentiment analysis, and machine translation. With its user-friendly interface and powerful features, NLTK remains a popular choice for developers, researchers, and educators working in the field of natural language processing.

## **4.7 DJANGO**

Django is a high-level web framework for Python that simplifies the development of web applications by providing a robust set of tools and conventions. It follows the Model-View-Template (MVT) architectural pattern, emphasizing modularity and reusability of code. With Django, developers can quickly build web applications with features such as a built-in ORM for database interaction, an admin interface for managing site content, and a powerful template engine for dynamic HTML rendering. Django's URL routing mechanism, views, and form handling capabilities further streamline the development process, enabling developers to create secure and scalable web applications efficiently. Additionally, Django prioritizes security, offering protection against common web vulnerabilities and providing tools for user authentication and authorization. Its comprehensive documentation, vibrant community, and extensive ecosystem of plugins and packages make Django a popular choice for building a wide range of web applications, from simple blogs to complex enterprise systems.

# **5 PROJECT DESCRIPTION**

## **5.1 PROJECT DEFINITION**

### **5.1.1 PREPROCESSING AND EMBEDDING**

The document undergoes preprocessing to enhance its readability and structure. This may include steps such as tokenization, lowercasing, removing stop words, and punctuation. Paragraphs are embedded using state-of-the-art transformer-based models such as BERT (Bidirectional Encoder Representations from Transformers) or RoBERTa (Robustly optimized BERT approach). Embedded

paragraphs are stored efficiently using the FAISS (Facebook AI Similarity Search) library, ensuring fast and scalable retrieval.

### **5.1.2 RETRIEVAL OF RELEVANT ANSWER**

When a specific question is provided, the system retrieves relevant answers from the preprocessed and embedded document. This retrieval is based on semantic similarity between the question and paragraphs in the document. Techniques such as approximate nearest neighbour search are employed to efficiently retrieve relevant answers from a large corpus.

### **5.1.3 STUDENT RESPONSE EVALUATION**

Upon receiving the student's answer to the question, the system evaluates it using various similarity metrics. These metrics include Jaccard Similarity, Word Mover's Distance (WMD) Similarity, BLEU Score, Meteor Score, Soft Cosine Similarity, and SIF Embedding Similarity. Each metric offers a unique perspective on the similarity between the student's answer and the relevant paragraphs from the document.

### **5.1.4 PENALIZATION FOR GRAMMATICAL ERRORS**

After evaluating the student's response based on similarity metrics, the system further analysis the answer for grammatical errors and spelling mistakes. This analysis is carried out using tools such as the TEXTGEARS API, which provides comprehensive checks for grammar, spelling, and punctuation errors.

### **5.1.5 FINAL SCORE CALCULATION:**

The scores obtained from the similarity metrics are combined using a weighted average approach, where each metric is assigned a weight based on its importance. The weighted average score is then adjusted based on the degree of penalization for grammatical errors and spelling mistakes. The resulting score represents the overall quality and relevance of the student's response to the given question.

### **5.1.6 OUTPUT**

The final score is returned to the user, providing feedback on the student's answer in an automated and efficient manner. Additionally, the system may provide detailed insights into areas for improvement, such as suggestions for correcting

grammatical errors or expanding on key points. Overall, the automated system streamlines the process of evaluating student answers, saving time for educators and providing valuable feedback to students to enhance their learning experience.

## **5.2 METRIC DESCRIPTION**

### **5.2.1 JACCARD SIMILARITY**

Jaccard Similarity is a measure of similarity between two sets, often used in text analysis and information retrieval. It quantifies the similarity by comparing the intersection and union of the elements in the sets. The formula for Jaccard Similarity is defined as the size of the intersection divided by the size of the union of the sets.

In the context of text analysis, the sets represent the unique words or tokens present in two pieces of text. Jaccard Similarity disregards the order of words and only focuses on whether words are present in both texts or not. It's particularly useful for tasks such as document retrieval, plagiarism detection, and recommendation systems where the arrangement of words may vary but their presence matters.

### **5.2.2 WMD SIMILARITY**

Word Mover's Distance (WMD) is a measure of semantic similarity between two pieces of text, which takes into account the meaning of words and their relationships. Unlike traditional distance metrics like Euclidean distance, WMD considers the semantic meaning of words when computing the distance between documents. WMD measures how similar two documents are by calculating the minimum "cost" of transforming one document into the other, where the cost is defined as the minimum amount of "work" needed to move each word from one document to another. WMD similarity is particularly effective when comparing text documents that may have different lengths, vocabularies, or word choices but convey similar meanings. It is especially useful in applications such as document retrieval, information retrieval, and natural language processing tasks like document clustering and text summarization.

### **5.2.3 BLUE SCORE**

BLEU (Bilingual Evaluation Understudy) is a metric commonly used to evaluate the quality of machine-translated text by comparing it to one or more reference translations. Originally designed for machine translation tasks, BLEU has since been adapted for use in various text generation and natural language processing



applications. BLEU operates by comparing n-grams (sequences of n consecutive words) between the candidate translation and reference translations. It computes precision scores for each n-gram size (typically up to 4-grams) and combines them using a weighted geometric mean to compute the overall BLEU score. The weights assigned to each n-gram precision score are often uniform, although they can be adjusted based on specific preferences or requirements. BLEU penalizes for brevity by comparing the length of the candidate translation to the lengths of the reference translations. This penalty discourages generating overly short translations that may achieve high precision scores simply by omitting content. BLEU scores range from 0 to 1, with higher scores indicating better translation quality. However, it's essential to interpret BLEU scores cautiously and consider them alongside other evaluation metrics, as BLEU has limitations, such as sensitivity to tokenization, inability to capture semantic equivalence, and dependence on reference translations.

#### 5.2.4 METEOR SCORE

METEOR (Metric for Evaluation of Translation with Explicit Ordering) is a metric commonly used to evaluate the quality of machine translation and text generation systems. It combines precision, recall, and alignment-based measures to provide a comprehensive evaluation of the similarity between a candidate text and one or more reference texts. Unlike some other metrics like BLEU, METEOR considers not only exact word matches but also stems, synonyms, and paraphrases. This makes METEOR more robust to variations in word choice and sentence structure, capturing a broader range of linguistic similarities between the candidate and reference texts.

The METEOR score is computed based on three main components:

**Unigram Precision:** Measures the percentage of words in the candidate text that are also present in the reference text(s).

**Unigram Recall:** Measures the percentage of words in the reference text(s) that are also present in the candidate text.

**Alignment-Based Measures:** Account for exact word matches and alignments between the candidate and reference texts.

METEOR combines these components using a weighted harmonic mean to compute the overall score. It also includes a penalty term to account for differences in word

length between the candidate and reference texts, penalizing for longer or shorter translations. METEOR scores range from 0 to 1, with higher scores indicating better translation quality. METEOR is widely used in machine translation research and evaluation due to its ability to capture a broader range of linguistic phenomena compared to simpler metrics like BLEU.

### **5.2.5 SIF EMBEDDING SIMILARITY**

SIF (Smooth Inverse Frequency) Embedding Similarity is a metric utilized to gauge the semantic similarity between two text segments based on their embeddings. Unlike simple embedding-based comparisons, SIF embedding similarity incorporates inverse document frequency (IDF) weighting to diminish the impact of common words and elevate the significance of rare terms. The process of computing SIF embedding similarity typically involves several steps. First, each word in the text is converted into a high-dimensional vector using pre-trained word embeddings like Word2Vec or GloVe. These embeddings encapsulate semantic information regarding each word's meaning and context. Next, the embeddings of individual words in the sentence are aggregated to construct a unified vector representation of the entire sentence. This aggregation can be achieved by methods like averaging the word embeddings or employing weighted averages based on word importance. IDF scores are then calculated for each word in the text corpus to quantify its importance in distinguishing between documents. Words with higher IDF scores, which are common across documents, are down weighted, while those with lower IDF scores, indicating rarity, are upweighted. The sentence embeddings are further adjusted using IDF weighting to derive SIF embeddings. This adjustment serves to diminish the impact of common words and amplify the semantic content of the text. Finally, the similarity between two SIF embeddings, representing two sentences, is computed using a standard similarity metric such as cosine similarity. Cosine similarity measures the cosine of the angle between the two vectors, indicating their directional similarity while discounting the influence of common words or noise. SIF embedding similarity proves particularly effective for tasks requiring semantic understanding of text, such as text classification, document clustering, and information retrieval. By incorporating IDF weighting, SIF embedding similarity enhances the discernment of semantic content while mitigating the interference of common terms.

## 5.2.6 SOFT COSINE SIMILARITY

Soft Cosine Similarity is a metric used to measure the similarity between two text documents based on their content. Unlike traditional cosine similarity, which operates on vector representations of words or documents, soft cosine similarity incorporates semantic information and adjusts for the similarity between words.

**1. Embedding Generation:** Words in the text are represented as vectors using pre-trained word embeddings, capturing semantic relationships between words.

**2. TF-IDF Weighting:** Term Frequency-Inverse Document Frequency (TF-IDF) scores are calculated for each word in the documents. TF-IDF reflects the importance of words in a document relative to a corpus, emphasizing words that are frequent in the document but rare in the corpus.

**3. Vectorization:** Each document is represented as a vector in the TF-IDF space, where each dimension corresponds to a unique word in the corpus.

**4. Soft Cosine Similarity Calculation:** Soft cosine similarity computes the cosine similarity between two document vectors, considering not only the angle between the vectors but also the similarity between the words they represent. Unlike traditional cosine similarity, soft cosine similarity incorporates a similarity matrix that quantifies the similarity between all pairs of words in the vocabulary.

**5. Normalization:** Soft cosine similarity is normalized by the geometric mean of the norms of the document vectors to ensure that the similarity score falls within the range of  $[0, 1]$ .

Soft cosine similarity is advantageous because it can handle synonyms, typos, and word variations effectively. By incorporating semantic information and adjusting for word similarity, soft cosine similarity provides a more nuanced measure of similarity between text documents. Soft cosine similarity finds applications in various natural language processing tasks, including document clustering, information retrieval, and text classification. Its ability to capture semantic similarity while accommodating variations in word usage makes it a valuable tool for tasks requiring fine-grained analysis of text similarity.

## 6 SYSTEM DESIGN

Unified Modeling Language (UML) is a general-purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite like blueprints used in other fields of engineering. UML is not a programming language, it is rather a visual language. We use UML diagrams to portray the behavior and structure of a system. UML helps software engineers, businessmen, and system architects with modeling, design, and analysis. The Object Management Group (OMG) adopted Unified Modelling Language as a standard in 1997. It's been managed by OMG ever since. The International Organization for Standardization (ISO) published UML as an approved standard in 2005. UML has been revised over the years and is reviewed periodically.

### **Why we need UML:**

- Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.
- Businessmen do not understand code. So, UML becomes essential to communicating with non-programmers' essential requirements, functionalities, and processes of the system.
- A lot of time is saved down the line when teams can visualize processes, user interactions, and static structure of the system. UML is linked with object-oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:
  - Structural Diagrams – Capture static aspects or structure of a system. Structural Diagrams include Component Diagrams, Object Diagrams, Class Diagrams, and Deployment Diagrams.
  - Behaviour Diagrams – Capture dynamic aspects or behaviour of the system. Behaviour diagrams include Use Case Diagrams, State Diagrams, Activity Diagrams, and Interaction Diagrams. Building Blocks of the UML Building Blocks of the UML Building Blocks of the UML.

### **6.1 BUILDING BLOCKS OF UML**

The vocabulary of the UML encompasses three kinds of building blocks:

- Things
- Relationships
- Diagrams

Things are abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

**Things in UML:** There are four kinds of things in the UML:

- Structural things
- Behavioural things
- Grouping things
- Annotation things

These things are the basic object-oriented building blocks of the UML. You use them to write well-formed models.

**Structural Things:** Structural things are the nouns of UML models. These are the mostly static parts of a model, representing elements that are either conceptual or physical. Collectively, the structural things are called classifiers. A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics. A class implements one or more interfaces. Graphically, a class is rendered as a rectangle, usually including its name, attributes, and operations.

- Class
- Interface
- Collaboration
- Use Case
- Component

**Behavioural Things:**

Behavioural things are the dynamic parts of UML models. These are the verbs of a model, representing behaviour over time and space. In all, there are three primary kinds of behavioural things Interaction State machine

**Grouping Things:**

Grouping things can be defined as a mechanism to group elements of a UML model together. There is only one grouping thing available. Package – Package is the only one grouping thing available for gathering structural and behavioural things.

### **Annotation Things:**

Annotation involves adding descriptive or explanatory notes to a text, image, or dataset to provide additional context or information. These annotations help clarify content, highlight key points, or categorize data, enhancing understanding and facilitating analysis.

### **Relationships in the UML:**

Relationship is another most important building block of UML. It shows how the elements are associated with each other and this association describes the functionality of an application. There are four kinds of relationships in the UML:

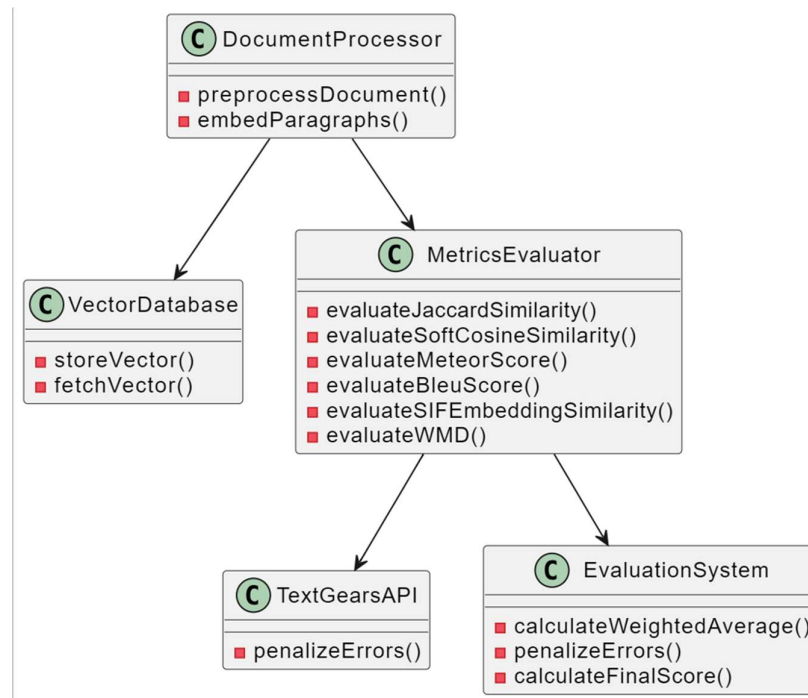
- Dependency
- Association
- Generalization
- Realization

## **6.2 UML DIAGRAMS**

UML diagrams are graphical representations used in software engineering to visualize, design, and communicate system structures, behaviours, and interactions.

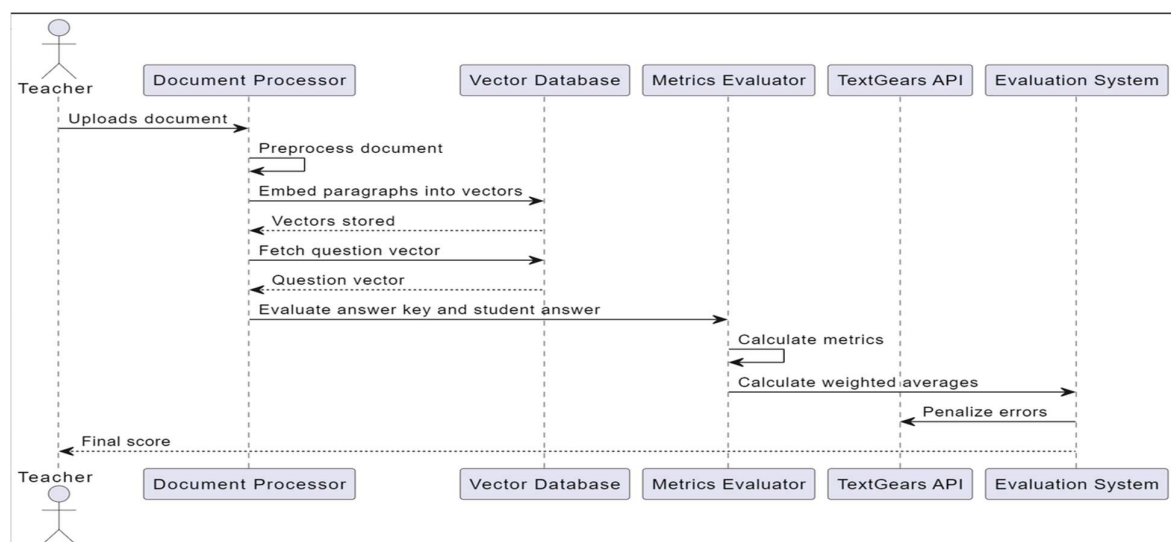
### **6.2.1 CLASS DIAGRAM**

A class diagram visually represents the structure of a system by depicting classes, attributes, methods, and their relationships. It illustrates how classes interact and collaborate to fulfill system functionalities through associations, dependencies, and inheritance. Class diagrams serve as blueprints for designing and communicating software architectures effectively.



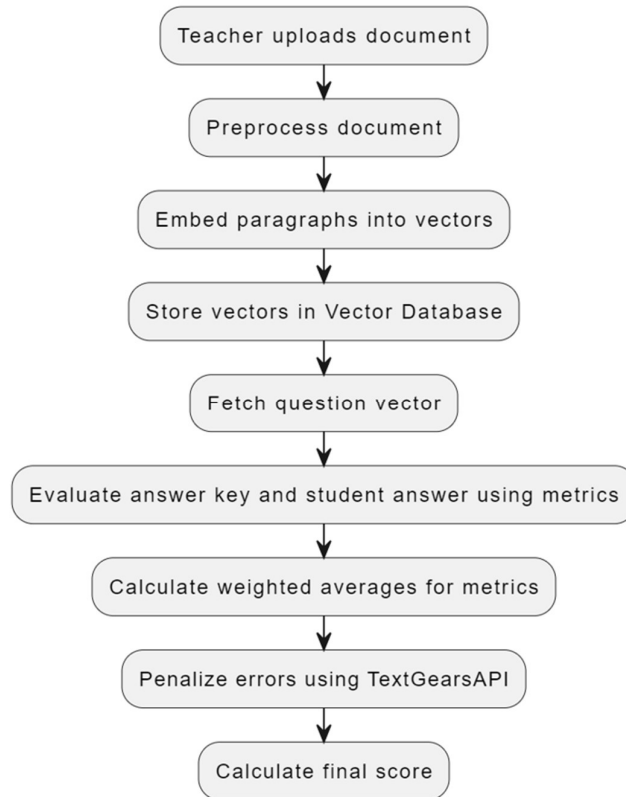
## 6.2.2 SEQUENCE DIAGRAM

A sequence diagram in UML visually depicts the interaction between objects within a system over time. It showcases the sequence of messages exchanged among objects to fulfill a specific scenario or use case. Lifelines represent the objects involved, while arrows indicate the flow of messages between them.



### 6.2.3 ACTIVITY DIAGRAM

An activity diagram in UML represents the flow of activities or actions within a system or process. It illustrates the sequence of actions, decision points, and transitions between activities, providing a visual representation of the workflow.





## 7 DEVELOPMENT

### 7.1 SAMPLE CODE

```
def jaccard_similarity(set1, set2):
    """Calculates the Jaccard similarity between two sets.

    Args:
        set1 (set): The first set.
        set2 (set): The second set.

    Returns:
        float: The Jaccard similarity score between the two sets.
    """

    intersection = len(set1.intersection(set2))
    union = len(set1.union(set2))
    return intersection / union

def wmd_similarity(text1, text2):
    distance = model.wmdistance(preprocess(text1), preprocess(text2))
    embeddings1 = encode_text(text1)
    embeddings2 = encode_text(text2)
    if distance == 0:
        return 1.0 # Default similarity for identical texts

    # Find minimum and maximum WMD for normalization
    min_distance = min(distance, util.pytorch_cos_sim(embeddings1,
embeddings2).item()) # Consider cosine similarity for very similar
cases
    max_distance = max(distance, util.pytorch_cos_sim(embeddings1,
embeddings2).item())

    # Normalize WMD for similarity score (0 for most similar, 1 for
most dissimilar)
    similarity = 1 - (distance - min_distance) / (max_distance -
min_distance)
    return similarity
def soft_cos_sim(text1, text2):
    pt1=preprocess(text1)
    pt2=preprocess(text2)
    documents = [pt1,pt2]
    dictionary = Dictionary(documents)

    print(pt1)
    print(pt2)

    text1_bow = dictionary.doc2bow(pt1)
```

```

text2_bow = dictionary.doc2bow(pt2)

documents_bow = [text1_bow, text2_bow]
tfidf = TfidfModel(documents_bow)

print(text1_bow, text2_bow)

tf1 = tfidf[text1_bow]
tf2 = tfidf[text2_bow]

print(tf1)
print(tf2)

termsim_index = WordEmbeddingSimilarityIndex(model)
termsim_matrix = SparseTermSimilarityMatrix(termsim_index,
dictionary, tfidf)

similarity = termsim_matrix.inner_product(tf1, tf2,
normalized=(True, True))
print()

return round(similarity, 4)
def cal_bleu_score(candidate, reference):
    """Calculates the BLEU score between a candidate text and a
    reference text.

    Args:
        candidate (str): The candidate text.
        reference (str): The reference text.

    Returns:
        float: The BLEU score between the candidate and reference
        texts.
    """

    return corpus_bleu([[reference]], [candidate])

# 3.5 METEOR Score (continued)
def cal_meteor_score(candidate, reference):
    """Calculates the METEOR score between a candidate text and a
    reference text.

    Args:
        candidate (str): The candidate text.
        reference (str): The reference text.

    Returns:
        float: The METEOR score between the candidate and reference
        texts.

```

```

"""
# reference=reference.split()
# candidate=candidate.split()
print(type(reference))
print(type(candidate))
print(meteor_score(candidate,reference))
#return meteor_score([reference], candidate)

def smooth_inverse_frequency_embedding_similarity(model, sentence1,
sentence2):
    """Calculates the Smooth Inverse Frequency Embedding Similarity
(SIF) between two sentences.

    Args:
        model (sentence_transformers.SentenceTransformer): The
SentenceTransformer model.
        sentence1 (str): The first sentence.
        sentence2 (str): The second sentence.
        dictionary (gensim.corpora.Dictionary): The dictionary used
for bag-of-words representation.

    Returns:
        float: The SIF score between the two sentences.
    """

    alpha = 1e-4 # SIF hyperparameter
    sentence1_embedding = model.encode(sentence1,
show_progress_bar=False)
    sentence2_embedding = model.encode(sentence2,
show_progress_bar=False)

    # Calculate word frequencies
    word_freq = {}
    for word in sentence1 + sentence2:
        if word in word_freq:
            word_freq[word] += 1
        else:
            word_freq[word] = 1

    # Calculate smoothed word embeddings
    smooth_sentence1 = []
    for word in sentence1:
        smooth_sentence1.append(model.encode(word,
show_progress_bar=False) / (alpha + word_freq[word]))
    smooth_sentence2 = []
    for word in sentence2:
        smooth_sentence2.append(model.encode(word,
show_progress_bar=False) / (alpha + word_freq[word]))

```

```

# Calculate sentence embeddings
smooth_sentence1 = np.mean(smooth_sentence1, axis=0)
smooth_sentence2 = np.mean(smooth_sentence2, axis=0)

# Remove the projection of the first principal component (for
dimensionality reduction)
pca = PCA(n_components=1)
pca.fit(np.array([smooth_sentence1, smooth_sentence2]))
u = pca.components_[0]
smooth_sentence1 -= np.dot(np.dot(smooth_sentence1, u), u)
smooth_sentence2 -= np.dot(np.dot(smooth_sentence2, u), u)

# Calculate cosine similarity
similarity = np.dot(smooth_sentence1, smooth_sentence2) /
(np.linalg.norm(smooth_sentence1) * np.linalg.norm(smooth_sentence2))
return similarity
def detect_grammar_errors(text, api_key="YOUR_TEXTGEAR_API_KEY"): #
Replace with your actual API key

    client = textgears_api_client.client(api_key)
    try:
        response = client.check(text)
        return len(response.errors)
    except Exception as e:
        print(f"Error using TextGears API: {e}")
        return 0 # Handle API errors gracefully (e.g., return 0
errors)


def calculate_final_score(similarity_scores, weights):
    """Calculates the final score for the student's answer based on
similarity metrics and grammatical errors.

    Returns:
        float: The final score for the student's answer.
    """
    weighted_similarity_sum = sum(weight * score for weight, score in
zip(weights, similarity_scores))
    #error_deduction = 0.1 * num_errors if num_errors <= 10 else 0.15 *
num_errors
    final_score = weighted_similarity_sum #- error_deduction
    return final_score

```

## 7.2 OUTPUT

### 7.2.1 HOME PAGE

 **Subjective Answer Evaluator** [About](#) [Technology](#)

**Upload Document for Analysis**


Document:  
[Choose File](#) | No file chosen

Question:

Answer:

Analysis


### 7.2.2 About Section

 **Subjective Answer Evaluator** [About](#) [Technology](#)

**About**

Assessing subjective papers manually is both challenging and tedious, primarily due to the complexities involved in comprehending and interpreting the data. Previous attempts to automate the scoring of students' answers using computer science have predominantly relied on basic word counting methods. This project proposes an innovative approach that harnesses advanced techniques from machine learning and natural language processing, including Word Mover's Distance (WMD) and Term Frequency-Inverse Document Frequency (TF-IDF), to autonomously evaluate descriptive answers. The evaluation process involves the utilization of solution statements and keywords, and a machine learning model is trained to predict answer grades. Notably, the study incorporates additional measures such as Jaccard Similarity, Smooth Inverse Frequency (SIF) embedding similarity, METEOR score, and BLEU score to enrich the assessment process. Results indicate that WMD outperforms other metrics, highlighting the effectiveness of incorporating diverse measures. With sufficient training, the machine learning model has the potential to operate as a standalone solution, streamlining the assessment of subjective papers and reducing the burden of manual grading.

### 7.2.3 Technology Section

 **Subjective Answer Evaluator** [About](#) [Technology](#)

**Technology**

NLTK

Django

Transformers


Scikit-Learn

Gensim

Textgears API

Python

## 7.2.4 Input 1

**Subjective Answer Evaluator**[About](#)[Technology](#)


**Upload Document for Analysis**

Document:  
 PROJECT ANS DOC.docx


Question:

Answer:  

Photosynthesis is a process by which phototrophs convert light energy into chemical energy, which is later used to fuel cellular activities. The chemical energy is stored in the form of sugars, which are created from water and carbon dioxide.



## 7.2.5 Output 1

**Subjective Answer Evaluator**


**Your question is**  
what is photosynthesis?

**Result**  
Photosynthesis is a process by which phototrophs convert light energy into chemical energy, which is later used to fuel cellular activities. The chemical energy is stored in the form of sugars, which are created from water and carbon dioxide.

**written answer**  
Photosynthesis is a process by which phototrophs convert light energy into chemical energy, which is later used to fuel cellular activities. The chemical energy is stored in the form of sugars, which are created from water and carbon dioxide.

**final score**  
0.997911872092806

## 7.2.6 Input 2

**Subjective Answer Evaluator**[About](#)[Technology](#)

**Upload Document for Analysis**

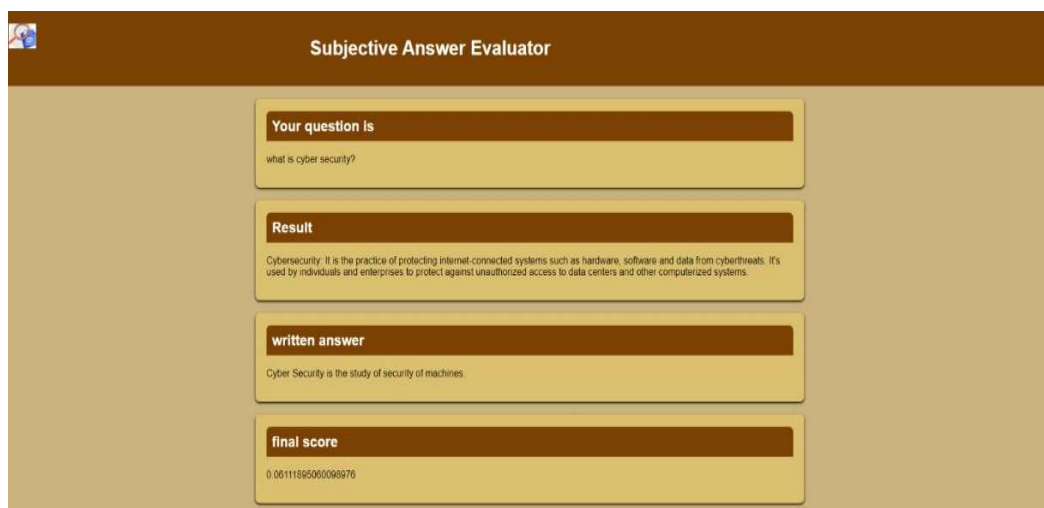
Document:  
 PROJECT ANS DOC.docx

Question:

Answer:  

Cyber Security is the study of security of machines.

## 7.2.7 Output 2



**Subjective Answer Evaluator**

**Your question is**  
what is cyber security?

**Result**  
Cybersecurity: It is the practice of protecting internet-connected systems such as hardware, software and data from cyberthreats. It's used by individuals and enterprises to protect against unauthorized access to data centers and other computerized systems.

**written answer**  
Cyber Security is the study of security of machines.

**final score**  
0.06111895060098976

## 7.3 Testing and Validation

QUESTION	TEACHER'S KEY	STUDENT'S ANS	SCORE	VALID / INVALID
1.What is the National animal of India?	Tiger is the National Animal of India.	Tiger is national animal	0.6647	Valid since student answer has a keyword missing and no punctuation.
2.Write about Ai?	AI, or artificial intelligence, simulates human intelligence in machines, enabling them to learn, reason, and solve problems, revolutionizing industries and daily life.	It is nothing but telling machines and using them for our purpose.	0.1249	Valid because just the answer has the word machine and an indirect purpose without any valid data.
3.Explain about sentence transformers and their usage.	Sentence transformers are models trained to convert textual sentences into fixed-dimensional vectors,	(Empty)	0.0000	Valid since no answer found
4.what is photosynthesis?	In above image	Same as key	0.9918	Valid

## 8 CONCLUSION

The project represents a significant step forward in the automation of subjective paper evaluation through the integration of advanced techniques from machine learning and natural language processing. By leveraging methodologies such as Word Mover's Distance (WMD), BLEU Score, Meteor Score and the project has demonstrated the potential to autonomously assess descriptive answers with remarkable accuracy and efficiency. The incorporation of additional similarity metrics further enriches the evaluation process, providing a more comprehensive understanding of semantic relationships within textual responses. Moreover, the exploration of machine learning models for predicting answer grades marks a pivotal advancement in streamlining the assessment process. The integration of predictive models offers a glimpse into the future of educational evaluation, where subjective papers can be graded with unprecedented speed and consistency. Beyond its immediate implications for educational assessment, the project's findings hold broader significance for the field of natural language processing and machine learning. By showcasing the efficacy of advanced techniques in handling complex textual data, the project contributes to the advancement of these fields, paving the way for future innovations and applications. The successful implementation of Word Mover's Distance and TF-IDF demonstrates their versatility and effectiveness in real-world scenarios, highlighting their potential for various other NLP tasks. Furthermore, the project underscores the importance of embracing technological advancements to address longstanding challenges in education. As the demand for personalized and adaptive learning solutions grows, the integration of automated evaluation systems becomes increasingly imperative. The project sets a precedent for the adoption of such systems in educational institutions, fostering a more efficient and effective learning environment for students worldwide. In conclusion, the project's achievements represent a significant milestone in the automation of subjective paper evaluation, offering promising prospects for enhancing educational assessment methodologies and shaping the future of learning and evaluation.



## 9 FUTURE SCOPE

The project's future scope encompasses a range of possibilities that can further advance automated subjective paper evaluation. One promising avenue is the continual refinement and optimization of machine learning models employed for predicting answer grades. Through ongoing experimentation with diverse algorithms and feature engineering techniques, the system's accuracy and reliability can be significantly enhanced. Integration of deep learning methodologies, such as recurrent neural networks (RNNs) or transformer models, holds promise for improving the system's comprehension of complex textual responses. Moreover, the development of adaptive learning systems presents an opportunity to personalize the educational experience based on individual student performance insights. By analysing students' responses and learning patterns, the system can provide tailored feedback and learning materials to address specific strengths and weaknesses, fostering a more effective learning environment. In addition, the implementation of real-time feedback and analytics features for both students and educators can further enrich the learning experience. Insights into students' progress and areas for improvement can empower educators to adjust their teaching strategies accordingly, while students can receive immediate feedback to guide their learning efforts. Ensuring scalability and seamless integration into existing educational platforms is crucial for the widespread adoption of automated evaluation systems. Compatibility with diverse infrastructures and the ability to handle large volumes of data will be essential considerations in the system's development. Furthermore, addressing ethical considerations, such as fairness, transparency, and bias, is paramount. Continuous monitoring and mitigation of potential biases in the system's predictions, as well as ensuring fairness in grading outcomes, are essential for maintaining trust and integrity in educational assessment. By pursuing these avenues for future development, the project has the potential to evolve into a comprehensive and robust solution that enhances educational assessment methodologies and improves learning outcomes for students. Continued research and innovation in this area will play a vital role in shaping the future of automated subjective paper evaluation and education as a whole.

## 10 REFERENCES

1. J. Wang and Y. Dong, "Measurement of text similarity: A survey", *Information*, vol. 11, no. 9, pp. 421, Aug. 2020. M. Fabietti et al.
2. P. Patil, S. Patil, V. Miniyar and A. Bandal, "Subjective answer evaluation using machine learning", *Int. J. Pure Appl. Math.*, vol. 118, no. 24, pp. 1-13, 2018.
3. M. Kusner, Y. Sun, N. Kolkin and K. Weinberger, "From word embeddings to document distances", *Proc. Int. Conf. Mach. Learn.*, pp. 957-966, 2015.
4. S. Aryal, K. M. Ting, T. Washio and G. Haffari, "A new simple and effective measure for bag-of-word inter-document similarity measurement", *arXiv:1902.03402*, 2019.
5. X. Jin, S. Zhang and J. Liu, "Word semantic similarity calculation based on Word2vec", *Proc. Int. Conf. Control Autom. Inf. Sci. (ICCAIS)*, pp. 12-16, Oct. 2018.
6. R. Sato, M. Yamada and H. Kashima, "Re-evaluating word mover's distance", *arXiv:2105.14403*, 2021.
7. V. Bahel and A. Thomas, "Text similarity analysis for evaluation of descriptive answers", *arXiv:2105.02935*, 2021.
8. [https://www.researchgate.net/publication/356548091\\_Subjective\\_Answers\\_Evaluation\\_Using\\_Machine\\_Learning\\_and\\_Natural\\_Language\\_Processing](https://www.researchgate.net/publication/356548091_Subjective_Answers_Evaluation_Using_Machine_Learning_and_Natural_Language_Processing)
9. Farrukh Bashir, Hamza Arshad, Abdul Rehman Javed, Natalia ryvinska, "Subjective Answer Evaluation Using Machine Learning and Natural Language Processing", 10.1109/ACCESS.2021.3130902, IEEE Access.