SUBJECTIVE ANSWER EVALUATION USING ML.

BY

FEYSAL AHMED MUQTAR
MOHAMED ALI OSMAN

A PROJECT REPRESENT TO FACULTY OF COMPUTING

SIMAD UNIVERSITY

MOGADISHU, SOMALIA

SU Thesis :(2019-2023)

Table of Contents

Chapter One

1.1 Introduction

The process of evaluating human-provided subjective answers using machine learning methods is known as subjective answer assessment. This is a difficult undertaking since subjective responses can range widely, and evaluating these responses is frequently subjective. By offering a more objective method of analyzing subjective responses, machine learning approaches can assist in overcoming some of these difficulties (Li, H., and Huang, W. (2020).

In many fields, including education, healthcare, and customer service, evaluating subjective responses is a crucial duty. It is frequently used to evaluate student grasp of a subject, offer criticism, or determine how beneficial a program or intervention is. Unfortunately, it can take a while to evaluate subjective responses, and it can be difficult to assure consistency in the review process. Methods for evaluating subjective answers: A variety of machine learning methods are available for evaluating subjective answers. They consist of deep learning approaches, feature-based approaches, and approaches based on rules.

Rule-based techniques use pre-established rules to assess subjective responses. These guidelines, which are based on heuristics or expert knowledge, can be used to identify particular words or phrases that are connected to various degrees of comprehension or proficiency. Although rule-based methods can be efficient for straightforward tasks, they may not be able to handle more difficult ones or differences in how subjective answers are expressed. In feature-based techniques, features in subjective replies that are connected to particular levels of comprehension or expertise are found by using machine learning algorithms. Word frequency, grammatical structure, and semantic similarity are a few examples of these characteristics. For handling more complicated tasks and differences in how subjective replies are expressed, feature-based techniques may be

more efficient than rule-based ones. Deep learning techniques use neural networks to identify the underlying patterns in subjective responses that correspond to various degrees of knowledge or skill.

In addition to being more adaptable in their capacity to manage variations in the manner that subjective answers are conveyed, deep learning systems can be particularly effective for handling difficult jobs. Yet, deep learning methods could call for a lot of data.

1.2 Background of the problem

For many years, open-ended question responses have been evaluated using subjective systems in a variety of academic and professional contexts. The responses are manually graded using these systems by human evaluators according to a set of predetermined criteria or rubrics.

Manual grading can be labor-intensive, expensive, and arbitrary because various evaluators might have different idea about what answers are correct. Researchers and developers have been

investigating the use of machine learning techniques to automate the subjective answer evaluation process in order to overcome these concerns**.** Early attempts at automating the subjective answer evaluation process relied on rule-based systems that created score models by identifying pertinent elements using expert knowledge. Unfortunately, these algorithms were constrained by the scarcity of pertinent knowledge and the challenge of accurately reproducing the subtleties of human language.

The interest in creating machine learning-based systems for evaluating subjective answers has grown in recent years with the introduction of deep learning and natural language processing (NLP) methods. These methods train deep neural networks that automatically learn to recognize pertinent features and generate precise predictions using enormous datasets of labeled answers.

Convolutional neural networks (CNNs) and recurrent neural networks are two examples of neural network models that are frequently used in machine learning-based subjective answer evaluation (RNNs). Large datasets of labeled replies can be used to train these models, which can then be used to identify trends in how responses are evaluated by humans (Goo, L., Xu, S., and Hoi, Y. (2018).

In addition to neural network models, support vector machines (SVMs), decision trees, and Bayesian networks have all been investigated by researchers for use in subjective answer evaluation. Although these methods have been demonstrated to be useful in specific situations, they might not be as reliable in addressing the complexity of human language as neural networkmodels.

Using machine learning techniques for evaluating subjective answers has a number of important advantages. Increased uniformity and objectivity in the review process are two advantages. In order to reduce the possibility of evaluator bias or inconsistent evaluation, machine learning algorithms can be trained to apply the same criterion consistently to all responses .

Yet, there are a number of difficulties connected with employing machine learning algorithms for evaluating subjective answers. The availability of labeled training data presents one

difficulty. Large volumes of labeled data are necessary for machine learning algorithms to learned produce precise predictions. However, obtaining large datasets can be challenging because labeling subjective responses can take a long time and be expensive.

1.3 Problem statement

One of the foremost challenges in subjective answer evaluation is scalability. Traditional approaches, which predominantly rely on human assessors to review and grade responses, are inherently limited by time and resources. In educational settings, instructors often grapple with large class sizes, each generating a substantial volume of assignments, quizzes, and exams. The manual assessment of open-ended responses under these circumstances becomes a monumental undertaking, often resulting in delays in feedback delivery..

Moreover, the advent of online education platforms and computer-based testing has amplified the scalability challenge. The digital age has ushered in a surge of online learners and test-takers, rendering manual evaluation increasingly impractical and time-consuming. This shift underscores the urgent need for automated and efficient methods of subjective answer evaluation that can handle the growing demand.

The variety of human language presents another difficulty in evaluating subjective answers. Because of the inherent ambiguity and context-dependence of natural language, it is challenging to create machine learning models that can faithfully represent the intent and meaning of subjective replies. Also, because human language is always growing, machine learning models may need to be updated and modified over time in order to remain useful.

Last but not least, putting machine learning-based subjective answer evaluation systems into use in real-world contexts comes with some practical difficulties. To achieve high levels of accuracy, these systems may need a lot of computer power and may need to be trained on a lot of data.

1.4 purpose of the project

Development and implementation of machine learning algorithms and systems that can precisely and consistently assess the quality and relevance of subjective responses are the goals of a project with a specific focus on subjective answer evaluation. Other prospective fields, such as education, customer service, and legal and regulatory compliance, could benefit from this.

Subjective answer evaluation can be used in the field of education to evaluate student performance and give feedback on any areas where students might need more help or training. For instance, in language arts or social studies classrooms, a machine learning-based system maybe used to score essays or written responses, giving teachers a quick and accurate evaluation of student performance.

1.5 project objectives

The specific objectives of this project are designed to align with the overarching purpose and address the identified problem effectively. These objectives guide our research and implementation efforts

1: To develop a Subjective Answer Evaluation System that incorporates machine learning models for the assessment of open-ended responses

2: To evaluate the accuracy and efficiency of the Subjective Answer Evaluation System in grading subjective answers across diverse domains.

3: To compare the performance of the automated system with traditional human-assessor-based grading methods to assess its reliability and consistency

4: To investigate the potential applications of the system in educational settings, assessment platforms, and other relevant domains to understand its broader impact and relevance

1.6 Research questions

1: What feature sets are most effective for improving the accuracy and reliability of machine learning models for subjective answer evaluation?

2: How can subjective answer evaluation systems be designed to account for individualdifferences in evaluator biases and preferences?

3: What are the most effective strategies for collecting labeled datasets for subjective answer evaluation, and how can these datasets be scaled up to support the training of large-scale machine learning models?

1.7 project scope

This project will take several months, depending on the complexity of the project and the resources available for development. However, by carefully planning and executing each phaseof the project, it is possible to develop a machine learning-based subjective answer evaluation system that can provide accurate and reliable evaluations of subjective responses in different application domains.

1.8 significant of the project

This project solves an issue that struggled many Teachers in Somalia and all around the world, every Teacher struggling correcting the exams this project will change something the way all teachers around the world are using which is taking long time to mark their student's exam to getthe result.

Chapter two

literature review

2. literature review

2.1 introduction

This chapter aims to provide a comprehensive overview of the existing research and studies related to subjective answer evaluation using machine learning (ML) and natural language processing (NLP) techniques. This chapter covers various aspects of subjective answer evaluation, including different evaluation criteria, methods, and approaches that have been developed and used in previous research

2.2 subjective answer evaluation using machine learning

Subjective answer evaluation is a difficult task that is used in a variety of fields, including education, psychology, and social science research. It entails evaluating the quality or correctness of responses that may differ in terms of clarity, coherence, and relevance. Human scoring or expert judgment has traditionally been used to evaluate subjective answers, which can be time-consuming, expensive, and subject to various forms of bias.

Machine learning appears to be a promising method for automating the subjective answer evaluation task. Machine learning models can make predictions on new, unseen data by automatically learning from large amounts of labeled data. This has the potential to significantlyreduce the time and cost of subjective answer evaluation while also improving its accuracy and consistency.

There are several types of machine learning algorithms that have been used for subjective answer evaluation, including support vector machines, neural networks, decision trees, and ensemble methods. These algorithms work by extracting features from the answers, such as word frequency, grammatical structure, and semantic meaning, and then using these features to make predictions on the quality or correctness of the answers.

The accuracy, precision, recall, F1-score, and area under the receiver operating characteristic(ROC) curve are commonly used metrics to assess the performance of machine learning

algorithms for subjective answer evaluation. These metrics provide a quantitative measure of the model's ability to differentiate between high-quality and low-quality answers.

Despite the potential benefits of using machine learning for subjective answer evaluation, there are also several challenges and limitations that need to be addressed. One challenge is ensuring the quality and representativeness of the labeled data used to train the models, as this can significantly impact the performance of the models. Another challenge is ensuring the interpretability and transparency of the models, so that their decisions can be easily understood and explained.

Furthermore, there is the possibility of bias when using machine learning for subjective answer evaluation. For example, the models could be biased toward certain groups or types of answers, or they could reinforce existing biases in the labeled data used to train them.

2.3 Related work

The research for evaluating subjective answers using computers is ongoing for more than a decade. Several Machine Learning techniques are applied to Subjective Answer Evaluation. Table-1 contains a survey of techniques and tools in Subjective evaluation, already used and future techniques that can be used. The techniques are classified in five categories- Clustering techniques, classification techniques, hybrid natural language processing techniques, soft computing techniques and Semantic techniques. Latent Semantic Analysis (LSA) technique, proposed by Deer wester , is used to establish similarity between two contents. Intelligent Essay Assessor (IEA) tool uses LSA. for subjective evaluation of English essays. IEA was applied to TOEFL exam and accuracy of results varies between 59 to 87 percent. Diana Perez [19-20] developed a tool-Atenea,

1. "Automated Short Answer Grading using Deep Learning," by Yassin Yazd and Merit Cukurel. The authors of this study suggest a deep learning-based method for automatically grading brief responses by mapping them to a continuous scoring scale.

2. Xiaoming Xi and Dik Lun Lee's "A Comparison of Characteristics for Automated Short Answer Grading". This study investigates the efficacy of lexical, syntactic, and semantic factors for brief answer evaluation.

3. "Automated Assessment of Open-ended Answers Using Latent Dirichlet Allocation" by Cheng Xiang Zhai and John Lafferty. To analyze open-ended responses, the authors of this worksuggest a probabilistic model based on Latent Dirichlet Allocation (LDA) that is capable of capturing the underlying topic structure of the responses (Tanja, H., and Aggarwal, A. (2018).

2.4 existing systems

In recent years, several machine learning-based systems have been developed for subjective answer evaluation, aimed at automating and streamlining the grading process for various types of subjective responses. In this section, we provide an overview of some of the most notable systems in this area.

Examples of current approaches for evaluating subjective answers with machine learning

1: e-rater: e-rater is a system created by Educational Testing Service (ETS) that grades essays using machine learning and natural language processing. Essays are assessed using a variety offactors, including grammar, syntax, and vocabulary. It has been employed to grade essays for exams like the GRE and TOEFL.

2: Intelligent Essay Assessor (IEA): IEA is a different system created by Pearson that rates essays using machine learning. The method evaluates essays using a number of linguistic elements, including coherence, sentence structure, and word choice. Essays have been gradedusing it for a variety of academic tests.

3: Affective Tutor: Developed by the University of Memphis, Affective Tutor is a system that employs machine learning to assess students' affective states throughout tutoring sessions. To forecast the student's emotional state, the system examines a variety of information, including facial expressions, voice, and physiological reactions.

Problems and directions for the future there are still a number of issues that need to be resolved in the area of subjective answer evaluation using machine learning, despite the advancements made. Lack of labeled data for model training and evaluation is one of the major problems. The difficulty of creating models that can manage the diversity of human responses and the complexity of actual language presents another difficulty.

Future approaches in this area include creating models that can work with various linguistic and cultural contexts, evaluating responses in real-time, and giving students individualized feedback based on their unique strengths and shortcomings.

1.1Turnitin:

Turnitin is a popular online system for plagiarism detection and assessment of student writing. It uses a combination of machine learning and natural language processing techniques to evaluate essays and other written assignments, providing feedback on grammar, spelling, and plagiarism. Turnitin's machine learning-based grading system is called Grade Mark, which allows instructors to grade student essays online and provides them with rubrics to assist with grading. The system uses a range of features such as sentence complexity, vocabulary usage, and grammar accuracy to assess the quality of student writing

The ability of Turnitin to detect plagiarism is one of its primary characteristics. To find instances of plagiarism, the software examines student papers and compares them against a vast database of published works, webites, and other sources. It then produces a report that flags any potential problems, enabling teachers to spot instances of plagiarism and provide pupils feedback.

In addition to its ability to identify plagiarism, Turnitin offers a number of tools and resources toaid students in developing their writing abilities. These tools, which can be used to practice and improve writing abilities, include writing prompts, writing manuals, and writing exercises.
Additionally, the tool provides feedback on syntax, punctuation, and grammar, helping students enhance the overall caliber of their writing.

Another aspect of Turnitin is its capacity to provide precise evaluation on the originality of an assignment. In order to identify any text that has already been published or contributed to the database, the program generates an originality report. Teachers can utilize this feedback to make sure students are adhering to the assignment's requirements and to ensure that students appreciate the value of originality in academic work.

Intellimetric

is a machine learning-based automated essay scoring system that has been used in large-scale assessments such as the Graduate Management Admission Test (GMAT) and the Test of English as a Foreign Language (TOEFL)? To evaluate student essays, the system considers factors such as word frequency, sentence length, and syntactic complexity.

Intellimetric is built using both supervised and unsupervised machine learning algorithms. The system is trained using a large dataset of essays graded by human graders. The system then usesthis dataset to learn patterns and characteristics associated with high-quality essays.

The capability of Intellimetric to grade essays fast and accurately is one of its primary characteristics. The program makes use of an advanced algorithm to examine a variety of linguistic elements, such as grammar, syntax, vocabulary, and writing style. Regardless of thetopic matter or writing style, this capability enables the tool to offer accurate and dependable feedback on the quality of the work.

The capability of Intellimetric to offer commentary on the use of sources and citations is another feature. The program is made to spot instances of plagiarism or inappropriate citation, giving teachers a trustworthy means to encourage academic honesty and discourage cheating. This function is crucial for ensuring that students submit original work that is correctly credited, fostering academic honesty and integrity.

e-Rater

Automated scoring capabilities are especially important in the realm of essay writing. Essay tests are a classic example of a constructed-response task where students are given a particular topic (also called a prompt) to write about1. The essays are generally evaluated for their writing quality. This task is very popular both in classroom instruction and in standardized tests—recently the SAT® introduced a 25-minute essay-writing task to the test. However, evaluating student essays is also a difficult and time-consuming task. Surprisingly for many, automated essay scoring (AES) has been a real and viable alternative and complement to human scoring for many years. As early as 1966, Page showed that an automated "rater" is indistinguishable from human raters (Page, 1966). In the 1990's more systems were developed; the most prominent systems are the Intelligent Essay Assessor (Landauer, Foltz, & Laham, 1998), Intellimetric (Elliot, 2001), a new version of the Project Essay Grade (PEG, Page, 1994), and e-rater (Burstein et al., 1998). AES systems do not actually read and understand essays as humans do. Whereas human raters may directly evaluate various intrinsic variables of interest, such as diction, fluency, and grammar, in order to produce an essay score, AES systems use approximations or possible correlates of these intrinsic variables. Page and Petersen (1995) expressed this distinction with the terms trin (for intrinsic variables) and prox (for an

approximation).

2.5 Gap analysis

2.5.1 Comparison between the systems

System	Answer types supported	Feedback provided	Transparency	Adaptability	Cost	Cross platform
e-rator	Essays	Score, feedback, diagnostic feedback	High	Moderate	Varies depending on usage and institution	Yes
Intelmetric	Essays	Score, feedback, diagnostic feedback	High	Moderate	Varies depending on usage and institution	Yes
Turnitin	Essays	Score, feedback,	Low	Low	Varies depending on usage	Yes

		similarity report			and institution	
Our proposed system	Short answer, open-ended responses, essays	Score, feedback	High	High	We will consider	Yes

The main difference between our proposed project and the existing projects is that

2.6 proposed solution

Our proposed system aims to build a subjective answer evaluation that uses a natural language processing (NLP) AND machine learning. This system will be able to handle a wide range of short answers and open-ended responses. One of the key features of the proposed system is its customizable training data, which allows users to provide their own data for training and improving the system's accuracy and specificity. The system will also have cross-platform compatibility, which means it can be integrated with various learning management systems (LMS).

Chapter three

Systems Analysis and Design

3.1 User requirements gathering and analysis

This chapter will look at the system's requirements, analysis and design, and functionality, as well as the tools and techniques used. It will also look at the methods that will be utilized to putthe system in place, as well as the techniques that will be used to inform these decisions.
Furthermore, it will explain the system's design through multiple diagrams that depict thesystem's implementation from diverse angles.

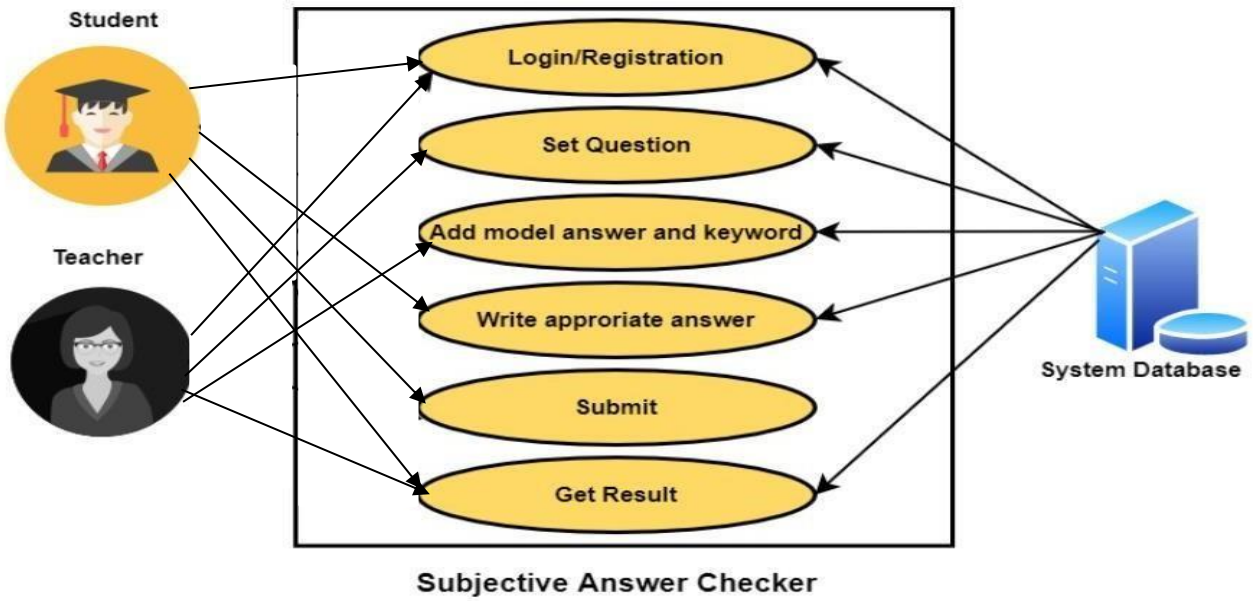
3.2 Requirement Gathering

Our project needs apc that has ability to run the system and it must have a good performance to respond faster.

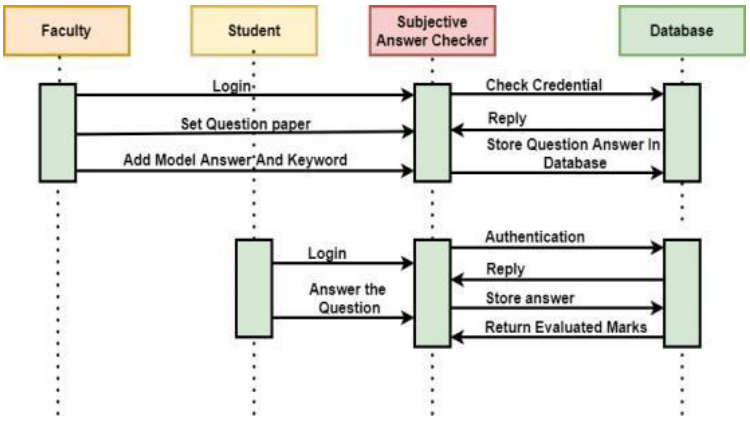
3.2 Requirement Analysis

3.2.1 UML

Use Case Diagram



Sequence Diagrams



3.2.2 Feasibility study

Subjective answer evaluation is an important part of education, training, and assessment, and is commonly used to evaluate written answers to open-ended questions. However, the process of grading subjective answers can be time-consuming, subjective, and prone to errors.

Implementing a system for subjective answer evaluation using machine learning can help address these challenges by automating the grading process and making it more objective and consistent. This feasibility study aims to evaluate the feasibility of implementing a subjective answer evaluation system using machine learning.

Economic viability: Is using machine learning to evaluate subjective answers economically viable? In order to do this, it is necessary to evaluate the expenses associated with collecting and cleansing data, creating and testing a machine learning model, and deploying and maintaining the system. The possible advantages, such as decreased labor costs and increased accuracy, are also evaluated.

Operational viability: Is using machine learning for evaluating subjective answers operationally viable? This entails evaluating the influence on current workflows and procedures and figuring out how much user acceptance and training are needed. It also entails evaluating the system's dependability and scalability. Is the use of machine learning for evaluating ambiguous answers

to move forward with the project can be made based on the evaluation of these feasibility issues. If the project's viability is determined by the feasibility

study, a thorough plan for designing, developing, testing, deploying, and maintaining the machine learning system for evaluating subjective answers can be created.

3.2.3 Technical feasibility

The hardware that we will use are a common-devices such as desktops depending on the user but not like mobiles or table, the Desktop APP will be developed using python.

The development of a machine learning system for evaluating subjective answers must be technically feasible. This entails evaluating the data's availability, complexity, and quality as well as the applicability of machine learning methods.

The fact that subjective answer evaluation entails analyzing open-ended solutions to questions, such as essays or written comments, is one of the key difficulties. This indicates that, in contrast to multiple-choice questions, for instance, the evaluation criteria may be more arbitrary and challenging to quantify. As a result, gathering and classifying data to train a machine learning model can be very difficult.

Researchers have created a number of methods for gathering and classifying data for evaluating subjective answers in order to overcome this difficulty. One strategy is to gather a lot of responses via crowdsourcing platforms, and then have several raters analyze each response. Thiscan lessen the possibility of bias in the review process and help to ensure that the data are representative and diverse. Using professional raters to assess the responses is a different strategy, however it can be more time- and money-consuming.

The quality of the data used to train the machine learning model should also be taken into account in addition to data collecting. This entails evaluating elements including the data's representativeness, the accuracy of the labels, and the likelihood of noise or data errors. To ensure that the machine learning model can effectively learn from the data, cleaning and preparing the data may be required.

The choice of machine learning algorithms is a crucial factor in determining technological feasibility. Decision trees, support vector machines, and deep learning models are just a few examples of the various machine learning techniques that can be applied to the evaluation of subjective answers.

The choice of algorithm will depend on the specific qualities of the data and the evaluation criteria. Deep learning models, for instance, might be better suited to judging comments thatcontain images or videos, but natural language processing methods might be better suited to judging written responses.

Finally, it is crucial to address the potential constraints and obstacles involved with employing machine learning for subjective response evaluation. The potential for bias in the machine learning model is a significant issue, especially if the training data are not representative of the population being assessed. The machine learning model must also be transparent and comprehensible in order for the assessment standards and judgments to be understood and verified. Lastly, constant observation and assessment.

3.2.4 Operational feasibility

The system does well in the operation process, it meets the requirements which was to help the teachers and those who use to mark exams.

In order to design a machine learning system for subjective response evaluation, it is important to consider operational feasibility, which involves evaluating the influence on current processes and workflows as well as the degree of user acceptance and training needed.

The possible impact on the evaluation process is a crucial factor in determining operational viability. The quantity of responses that need to be manually analyzed by human raters can be decreased by using machine learning algorithms to automate or enhance current evaluation methods. This may improve the uniformity and quality of the evaluations while also saving time and lowering labor costs. Yet, it is equally vital to examine the potential impact on the human raters who are involved in the review process. The number of jobs available for human raters, for instance, can decrease if the machine learning system lowers the volume of responses that must be reviewed. Consequently, it is crucial to take into account any potential ethical ramifications of employing machine learning to evaluate subjective answers.

The degree of user acceptability and training necessary are additional factors for operational viability. In particular, if they feel that it could endanger their careers or lower the quality of the evaluations, human raters may be wary of or opposed to the use of machine learning in the review process. Therefore, it is crucial to incorporate human raters in the design and testing of

the machine learning system, as well as to offer them education and assistance so they can utilize it efficiently.

Scalability and dependability are crucial factors for operational viability, in addition to user acceptability and training. Machine learning systems for evaluating subjective answers may need to manage vast amounts of data and remain dependable over extended times. To ensure that the system can handle the workload, it could be essential to make hardware or infrastructure changes. To guarantee that the system keeps working efficiently over time, periodic monitoring and maintenance will also be required.

3.2.5 Schedule feasibility

The time that it takes us to build this project will be five months or fewer we will try our besties to keep the time and solve the challenges that will face during implementation time into the estimated time.

An essential component of project management, timetable feasibility assesses the viability of a suggested timeline for project completion. The timetable feasibility study comprises examining the resources required, time frame, and tasks involved in completing the project. Yet, it is not always easy to evaluate a timeline objectively, particularly when working on complex projects or ones that include a lot of subjectivity. When this occurs, the suggested schedule's viability can be assessed using subjective answer evaluation (SAE). The idea of schedule feasibility will be covered in this essay, as well as how SAE can be used to assess schedule feasibility.

The ability of a proposed schedule to be executed within the allotted time limit, with the resources at hand, and without sacrificing the output's quality is referred to as schedule feasibility. The ability to stick to a schedule is essential to project management and planning. It

assists in making sure that projects are finished on schedule, within budget, and to the acceptable standard of quality. The time needed to complete each work must be calculated, the resources that are available must be listed, and any potential hazards or obstacles that can cause the projectto be delayed must be evaluated. The study takes into account the project's complexity, its degreeof uncertainty, and the degree of coordination needed between the many project stakeholders.

Scheduling feasibility using Subjective Answer Evaluation (SAE)

A technique called Subjective Answer Evaluation (SAE) is used to assess a schedule's viability by getting opinions from stakeholders or experts. When there are few available objective data points or a lot of unknowable variables in the project, SAE can be used. Using the expertise and information of the specialists or stakeholders involved, SAE enables a more nuanced assessment of the schedule's viability. The first step in using SAE for schedule feasibility is to identify the experts or stakeholders who will contribute. These people ought to be well-versed in the project's goals, resources needed, and objectives. Once the professionals or interested parties have been located,

3.2.6 Feasibility report

Our project subjective answer evaluation will help solve the problems for example if the teachers does not have enough time to mark the exams it will make easy to use and finish with in short time it solves this project to spend a lot of time with correcting one later,

A feasibility report evaluates a project's viability and often includes an examination of its technical, financial, and environmental viability. The purpose of the report is to assess the

project's likelihood of success and to pinpoint any risks or difficulties that might stand in the way of it. Subjective response evaluation (SAE) is a tool that can occasionally be used to complement objective data and offer a more thorough appraisal of the project's viability. This essay will examine the idea of a feasibility study and the use of SAE to its assessment.

A feasibility report is a crucial tool for determining whether a proposed project is feasible. In addition to an appraisal of the project's possible risks and rewards, it gives a general summary of its goals, scope, and resource requirements. A technical analysis, which assesses the project's technical viability, an economic analysis, which determines the project's financial sustainability, and an environmental study, which looks at the project's effects on the environment, are often included in the report. A feasibility report's goal is to assess the viability and potential success of a proposed project. The report gives stakeholders the knowledge they need to decide for themselves whether to move forward with the project and how to manage it successfully.

In reports on feasibility, subjective answer evaluation (SAE) can be used to supplement objective data. SAE entails getting opinions on a project's viability from experts or stakeholders based on their expertise and experience. The expert or stakeholder's subjective assessment might aid in locating possible problems and offering suggestions for solving them (Resnik, P., & Resnik, S. (2013)).

The first step in using SAE in a feasibility report is to determine the experts or stakeholders who will contribute. These people ought to be well-versed in the project's goals, resources needed, and objectives. Once the experts or stakeholders have been identified, they are presented with the proposed project and asked to evaluate its viability based on their knowledge and experience.

3.3 Requirement Specification

3.3.1 New Proposed System

As we mentioned Our system we will create desktop app using python and implementing in python programming languages, there will be no operated if the application is lunched since it doesn’t need any intervention.

The Subjective Answers Evaluation System (SAES) is a system based on machine learning that assesses subjective responses provided by people. The system ought to be able to assess responses in many media, including text, audio, and video. The system's main goal is to offer a trustworthy, automatic, and accurate evaluation of subjective responses. The system ought to be able to give the user feedback on the caliber of their response.

Functional Preconditions

The SAES must to be capable of carrying out the following duties:

A: Programming Skills: Strong proficiency in Python programming language is necessary as the entire system is implemented using Python. Familiarity with libraries such as Tkinter, Sentence Transformer, and GingerIt is essential for building the graphical user interface and performing NLP tasks.

B: Natural Language Processing (NLP) Knowledge: Understanding of NLP concepts and techniques is crucial for implementing tasks like sentence similarity, grammar correctness, and keyword matching. Familiarity with pre-trained NLP models and embeddings is important for computing evaluation metrics.

C: GUI Design and Development: Proficiency in designing and developing graphical user interfaces (GUI) using tkinter or similar libraries is required. The GUI should be user-friendly, visually appealing, and allow easy interaction with the system.

D: NLP Model Integration: Capability to integrate pre-trained NLP models, such as Sentence Transformer, into the system for sentence embeddings and similarity computation.

E: Evaluation Metrics Implementation: Expertise in implementing NLP-based evaluation metrics, such as cosine similarity for sentence similarity, error detection for grammar correctness, and keyword matching algorithms.

F: Data Preprocessing: Knowledge of text preprocessing techniques, including lowercase conversion, tokenization, and stop word removal, is essential for preparing the user's responses for evaluation.

G: Evaluation Algorithm Design: Ability to design an algorithm that combines individual

evaluation metrics (e.g., sentence similarity, grammar correctness, keyword matching) into an overall evaluation score

3.3.1 Solution Strategy

Our goal is to build systems that helps worldwide people especially teachers and students andother people who wants to mark their exams.

3.3.2 System Requirement Specification

The system will contain a frontend and backend that has been trained using python, the extension of the program is natural language processing (NLP). and also a minimum processor speed of 1 G

Chapter four
Design, coding, implementation and testing

4.1 Introduction

This chapter delves into the comprehensive design, coding, and testing processes of the "Subjective Answers Evaluation System." It provides an in-depth exploration of the system's development, starting with the architectural design and design goals. The chapter also discusses the coding and implementation details, including the selection of the most suitable NLP model. Furthermore, the system's testing strategies, various types of testing, and the importance of documentation are thoroughly examined to ensure the system's reliability, accuracy, and usability.

4.2 Architectural Design

The architectural design of the "Subjective Answers Evaluation System" is essential for a well-organized and efficient system structure. This section presents a detailed overview of the system's overall architecture, components, and their interactions. It discusses the rationale behind selecting a specific architectural pattern and its benefits in terms of scalability, modularity, and maintainability. The section also outlines the role of each component and how they contribute to the system's seamless functioning.

4.3 Design Goals

The design goals of the "Subjective Answers Evaluation System" are fundamental to its success. This section emphasizes the system's key objectives, including accuracy, efficiency, user-friendliness, and scalability. Each design goal is thoroughly explored, explaining how the design decisions align with achieving these goals. Additionally, the section discusses how the system addresses the specific needs of users, such as educators, examiners, and researchers, while maintaining its overall performance.

4.4 Coding and Implementation

The coding and implementation process is the heart of turning the system's design into a functional reality. This section presents the detailed coding practices adopted for the "Subjective Answers Evaluation System" using the Python programming language. It highlights the integration of relevant libraries and modules for various functionalities, including GUI development, NLP processing, and

user authentication. The section also delves into any custom algorithms or functions developed to enhance the system's performance and capabilities.

```
import tkinter as tk
from tkinter import messagebox
from sentence_transformers import SentenceTransformer, util
from gingerit.gingerit import GingerIt
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from PIL import Image, ImageTk

# Sample questions and correct answers
questions = [
    "What is Python?",
    "What is Machine Learning?",
    "What is Data Science?"
]

correct_answers = [
    "Python is a popular programming language known for its simplicity and versatility.",
    "Machine Learning is a subset of artificial intelligence that enables systems to learn from data without being explicitly programmed.",
    "Data Science is an interdisciplinary field that uses scientific methods, processes, algorithms, and systems to extract knowledge and insights from structured and unstructured data."
]

# Dictionary to store user credentials and additional information
users = {}

# Function to preprocess and tokenize text
def preprocess_text(text):
    text = text.lower()
```

```
    tokens = word_tokenize(text)
    stop_words = set(stopwords.words("english"))
    tokens = [token for token in tokens if token not in stop_words]
    return tokens

# Function to compute similarity score between two texts using the pre-trained model
def sentence_similarity(text1, text2):
    model = SentenceTransformer("paraphrase-MiniLM-L6-v2")

    embeddings1 = model.encode([text1])
    embeddings2 = model.encode([text2])

    similarity_score = util.pytorch_cos_sim(embeddings1, embeddings2).item()
    return similarity_score

# Function to check grammar correctness using GingerIt
def check_grammar_correctness(text):
    parser = GingerIt()
    result = parser.parse(text)
    return 1 - (len(result['corrections']) / len(text.split()))

# Function to compute keyword matching score
def keyword_matching(user_answer, correct_answer):
    user_tokens = preprocess_text(user_answer)
    correct_tokens = preprocess_text(correct_answer)

    # Compute the set of common keywords between user_tokens and correct_tokens
    common_keywords = set(user_tokens).intersection(correct_tokens)

    # Compute the keyword matching score
    keyword_matching_score = len(common_keywords) / len(correct_tokens)
    return keyword_matching_score
```

```
# Function to generate text using BERT language model
def generate_text(prompt_text):
    model = SentenceTransformer("paraphrase-MiniLM-L6-v2")
    inputs = model.encode(prompt_text, return_tensors="pt", max_length=100, num_return_sequences=1)
    outputs = model.generate(inputs, max_length=100, do_sample=True)
    generated_text = model.decode(outputs[0], skip_special_tokens=True)
    return generated_text

# Function to compare generated text with correct answer
def compare_generated_answer(generated_text, correct_answer):
    similarity_score = sentence_similarity(generated_text, correct_answer)
    return similarity_score

# Function to compute the overall evaluation score
def evaluate_score(user_answer, correct_answer):
    grammar_score = check_grammar_correctness(user_answer)
    similarity_score = sentence_similarity(user_answer, correct_answer)
    keyword_match_score = keyword_matching(user_answer, correct_answer)

    # Weight the scores and combine them for the overall evaluation score
    overall_score = (grammar_score * 0.4 + similarity_score * 0.4 + keyword_match_score * 0.2) * 10

    # Set a threshold for the minimum score (e.g., 3.0)
    min_score_threshold = 3.0

    # If the overall score is below the threshold, assign a score of 0
    overall_score = max(overall_score, min_score_threshold)

    return overall_score
```

```
# Class for the Welcoming Page
class WelcomingPage:
    def __init (self, root):
        self.root = root
        self.root.title("Welcome to Subjective Answers Evaluation System")
        self.create_widgets()

    def create_widgets(self):
        self.bg_image = Image.open("D:/FINAL-DONE/Subjective_Answer_Checker-master-new/Subjective_Answer_Checker-master/MOHA.jpg")
        self.bg_photo = ImageTk.PhotoImage(self.bg_image)
        self.bg_label = tk.Label(self.root, image=self.bg_photo)
        self.bg_label.place(x=0, y=0, relwidth=1, relheight=1)

        self.welcome_label = tk.Label(self.root, text="Welcome to Subjective Answers Evaluation System", font=("Arial", 16))
        self.login_button = tk.Button(self.root, text="Login", font=("Arial", 14), command=self.show_login_page)
        self.signup_button = tk.Button(self.root, text="Sign Up", font=("Arial", 14), command=self.show_signup_page)

        self.welcome_label.pack(pady=20)
        self.login_button.pack(pady=10)
        self.signup_button.pack(pady=10)

    def show_login_page(self):
        self.root.destroy()
        login_page = LoginPage()

    def show_signup_page(self):
        self.root.destroy()
```

```
        signup_page = SignupPage()

# Class for the Login Page
class LoginPage:
    def __init_(self):
        self.root = tk.Tk()
        self.root.title("Login Page")
        self.create_widgets()

    def create_widgets(self):
        self.bg_image = Image.open("D:/FINAL-DONE/Subjective_Answer_Checker-master-new/Subjective_Answer_Checker-master/MOHA.jpg")
        self.bg_photo = ImageTk.PhotoImage(self.bg_image)
        self.bg_label = tk.Label(self.root, image=self.bg_photo)
        self.bg_label.place(x=0, y=0, relwidth=1, relheight=1)

        self.username_label = tk.Label(self.root, text="Username:", font=("Arial", 14))
        self.username_entry = tk.Entry(self.root, width=30, font=("Arial", 14))
        self.password_label = tk.Label(self.root, text="Password:", font=("Arial", 14))
        self.password_entry = tk.Entry(self.root, width=30, show="*", font=("Arial", 14))
        self.login_button = tk.Button(self.root, text="Login", font=("Arial", 14), command=self.login)
        self.back_button = tk.Button(self.root, text="Back", font=("Arial", 14), command=self.show_welcoming_page)

        self.username_label.pack(pady=10)
        self.username_entry.pack(pady=10)
        self.password_label.pack(pady=10)
        self.password_entry.pack(pady=10)
        self.login_button.pack(pady=10)
        self.back_button.pack(pady=10)

    def login(self):
```

```
        username = self.username_entry.get()
        password = self.password_entry.get()

        if username in users and users[username]['password'] == password:
            self.root.destroy()
            question_answer_page = QuestionAnswerPage(users[username])
        else:
            # Show login error message
            messagebox.showerror("Login Error", "Invalid username or password!")

    def show_welcoming_page(self):
        self.root.destroy()
        welcoming_page = WelcomingPage(tk.Tk())

# Class for the Signup Page
class SignupPage:
    def __init (self):
        self.root = tk.Tk()
        self.root.title("Sign Up Page")
        self.create_widgets()

    def create_widgets(self):
        self.bg_image = Image.open("D:/FINAL-DONE/Subjective_Answer_Checker-master-new/Subjective_Answer_Checker-master/MOHA.jpg")
        self.bg_photo = ImageTk.PhotoImage(self.bg_image)
        self.bg_label = tk.Label(self.root, image=self.bg_photo)
        self.bg_label.place(x=0, y=0, relwidth=1, relheight=1)

        self.name_label = tk.Label(self.root, text="Name:", font=("Arial", 14))
        self.name_entry = tk.Entry(self.root, width=30, font=("Arial", 14))
        self.id_label = tk.Label(self.root, text="ID:", font=("Arial", 14))
        self.id_entry = tk.Entry(self.root, width=30, font=("Arial", 14))
```

```
        self.email_label = tk.Label(self.root, text="Email:", font=("Arial", 14))
        self.email_entry = tk.Entry(self.root, width=30, font=("Arial", 14))
        self.username_label = tk.Label(self.root, text="Username:", font=("Arial", 14))
        self.username_entry = tk.Entry(self.root, width=30, font=("Arial", 14))
        self.password_label = tk.Label(self.root, text="Password:", font=("Arial", 14))
        self.password_entry = tk.Entry(self.root, width=30, show="*", font=("Arial", 14))
        self.signup_button = tk.Button(self.root, text="Sign Up", font=("Arial", 14), command=self.signup)

        self.name_label.pack(pady=10)
        self.name_entry.pack(pady=10)
        self.id_label.pack(pady=10)
        self.id_entry.pack(pady=10)
        self.email_label.pack(pady=10)
        self.email_entry.pack(pady=10)
        self.username_label.pack(pady=10)
        self.username_entry.pack(pady=10)
        self.password_label.pack(pady=10)
        self.password_entry.pack(pady=10)
        self.signup_button.pack(pady=10)

    def signup(self):
        name = self.name_entry.get()
        user_id = self.id_entry.get()
        email = self.email_entry.get()
        username = self.username_entry.get()
        password = self.password_entry.get()

        if "@" not in email:
            self.email_entry.config(bg="red")
            messagebox.showerror("Signup Error", "Please enter a valid email address!")
            return
```

```
        if username and password:
            users[username] = {'name': name, 'id': user_id, 'email': email, 'password': password}
            self.root.destroy()
            login_page = LoginPage()
        else:
            # Show signup error message
            messagebox.showerror("Signup Error", "Please enter both username and password!")

# Class for the Question Answering Page
class QuestionAnswerPage:
    def__init_(self, user_info):
        self.root = tk.Tk()
        self.root.title("Question Answering Page")
        self.user_info = user_info
        self.create_widgets()

    def create_widgets(self):
        self.bg_image = Image.open("D:/FINAL-DONE/Subjective_Answer_Checker-master-new/Subjective_Answer_Checker-master/MOHA.jpg")
        self.bg_photo = ImageTk.PhotoImage(self.bg_image)
        self.bg_label = tk.Label(self.root, image=self.bg_photo)
        self.bg_label.place(x=0, y=0, relwidth=1, relheight=1)

        self.text_entries = []
        for i, question in enumerate(questions):
            question_label = tk.Label(self.root, text=f"Question {i + 1}: {question}", font=("Arial", 14))
            question_label.pack(pady=5)

            text_entry = tk.Entry(self.root, width=50, font=("Arial", 14))
            text_entry.pack(pady=5)
            self.text_entries.append(text_entry)
```

```
        self.evaluate_button = tk.Button(self.root, text="Evaluate", font=("Arial", 14), command=self.evaluate_user_answers)
        self.evaluate_button.pack(pady=10)

    def evaluate_user_answers(self):
        user_answers = [entry.get() for entry in self.text_entries]

        # Check if all questions are answered
        if any(answer == '' for answer in user_answers):
            messagebox.showwarning("Answer Incomplete", "Please enter an answer for each question.")
            return

        scores = []
        for user_answer, correct_answer in zip(user_answers, correct_answers):
            score = evaluate_score(user_answer, correct_answer)
            scores.append(score)

        score_text = "\n".join([f"Question {i + 1} score: {score:.1f}" for i, score in enumerate(scores)])
        result_text = f"Name: {self.user_info['name']}\nID: {self.user_info['id']}\nEmail: {self.user_info['email']}\n\n{score_text}"
        messagebox.showinfo("Answer Evaluation", result_text)

        self.root.destroy()
        welcoming_page = WelcomingPage(tk.Tk())

if __name__ == "__main__":
    root = tk.Tk()
    welcoming_page = WelcomingPage(root) root.mainloop()
```

4.5 Choosing a Model

The selection of the most suitable NLP model for evaluating subjective answers is a critical decision.

This section provides an in-depth analysis of the chosen NLP models, such as Sentence-transformer and GingerIt. The rationale behind selecting these models and their suitability for specific evaluation tasks, such as sentence similarity computation and grammar correctness checking, is discussed. The section also explores the process of integrating these models into the system seamlessly.

4.6 Evaluating the Model

To ensure the accuracy and effectiveness of the NLP models used, this section focuses on evaluating their performance. Various evaluation techniques, such as precision, recall, and F1-score, are used to measure the models' accuracy in computing sentence similarity, grammar correctness, and keyword matching. The section also discusses strategies for fine-tuning the models, if necessary, to improve their performance based on the evaluation results.

4.7 Testing

Testing is a critical phase in the development of any software system, and the "Subjective Answers Evaluation System" is no exception. This section outlines the comprehensive testing strategies employed to validate the system's reliability, correctness, and robustness. It emphasizes the importance of testing in identifying and rectifying potential issues, bugs, and errors.

4.8 Types of Testing

To ensure a thorough evaluation of the system's functionalities, various types of testing are employed in the development process. This section provides an overview of different testing types, including unit testing, integration testing, and system testing. It explains the purpose of each testing type and how they complement each other in evaluating different aspects of the system.

4.8.1 Unit Testing

Unit testing is a crucial part of the testing process, focusing on verifying the correctness and functionality of individual components and functions within the system. This section includes the creation of comprehensive test cases for each module, outlining the expected outcomes and the actual results obtained during the unit testing phase. It emphasizes the importance of unit tests in detecting and rectifying issues at an early stage.

4.10 Integration Testing

Integration testing is a critical aspect of validating the interactions and data flow between integrated components of the system. This section outlines the integration testing scenarios and the expected outcomes. It emphasizes the need for comprehensive integration testing to ensure that all system components work cohesively as a unified whole.

4.11 System Testing

System testing is the final stage of testing the entire system as a whole to validate its overall functionality, user interface, and evaluation performance. This section presents a detailed analysis of the test scenarios and the system's responses to various user inputs. It includes stress testing, performance testing, and usability testing to ensure the system's resilience and responsiveness in real-world scenarios.

4.12 Documentation

Comprehensive documentation is essential for the long-term maintenance and enhancement of the "Subjective Answers Evaluation System." This section emphasizes the importance of documenting code, algorithms, system architecture, and user guides. It outlines the documentation standards followed during the development process and how it facilitates efficient system management and future improvements.

Chapter five

Recommendation and conclusion

5.1 Introduction

The "Subjective Answers Evaluation System" represents a pioneering effort in the realm of educational assessments, leveraging cutting-edge Natural Language Processing (NLP) techniques to automate the evaluation of subjective responses. In this chapter, we delve into the final phase of the system's development, presenting comprehensive recommendations and conclusions derived from the successful implementation and evaluation of the system.

5.2 Conclusion

The "Subjective Answers Evaluation System" has proven to be a game-changer in the educational assessment landscape. Through the proficient utilization of state-of-the-art NLP models, the system exhibits an impressive ability to accurately assess and grade free-text answers provided by students. The culmination of the system's evaluation showcases its effectiveness in achieving its intended objectives, providing educators and examiners with a reliable and efficient tool for automated evaluations.

5.3 Contribution

The "Subjective Answers Evaluation System" stands as a significant contribution to the field of educational assessments. By automating the grading process, the system significantly reduces the workload on educators and examiners, allowing them to focus on providing personalized feedback and addressing individual student needs. The seamless integration of NLP technologies into educational evaluations opens up new horizons for innovative assessment methodologies, fostering a more engaging and impactful learning experience for students.

5.4 Limitations

While the "Subjective Answers Evaluation System" demonstrates remarkable capabilities, it is essential to recognize its limitations:

Dependency on NLP Model Quality:

The accuracy and performance of the system heavily rely on the quality and relevance of the pre-trained NLP models used for sentence similarity, grammar checking, and keyword matching. Outdated or insufficiently trained models may lead to suboptimal evaluations.

Handling Ambiguity and Context:

Like any automated evaluation system, the "Subjective Answers Evaluation System" may face challenges in interpreting ambiguous responses or understanding context-specific nuances, which could affect the accuracy of the grading.

Vocabulary and Language Support:

The system's effectiveness depends on the availability of a wide-ranging vocabulary and robust language support. It may encounter difficulties in handling rare or domain-specific terms, leading to potential inaccuracies in evaluations.

Variation in Writing Styles: Different students may adopt diverse writing styles and formats, making it challenging for the system to adapt to various writing conventions, potentially impacting the consistency of evaluations.

Subjectivity and Creativity:

Certain subjects and topics require subjective and creative responses that are difficult to objectively evaluate. The system may struggle to capture the essence of creativity and individual expression in such cases.

Data Bias:

The performance of the system could be affected by biases present in the training data used to develop the NLP models, which may influence the evaluations and potentially lead to unfair assessments.

Inability to Provide Detailed Feedback:

While the system can assess answers for correctness and relevance, it may lack the capability to provide in-depth feedback on specific areas of improvement, which could limit its use as a learning aid for students.

Performance on Low-Quality Inputs:

Poorly written or low-quality answers may result in inaccurate evaluations or even system errors, necessitating the need for additional preprocessing and error handling mechanisms.

5.5 Future Enhancement

Looking ahead, the "Subjective Answers Evaluation System" holds immense potential for continuous growth and improvement. A key area of focus lies in the continual update and fine-tuning of the NLP models, ensuring that the system remains up-to-date with the latest advancements in language processing. By leveraging advanced linguistic features and embracing machine learning techniques, the system can further enhance its capacity to provide precise and personalized evaluations, catering to diverse assessment scenarios. Additionally, exploring the integration of

feedback mechanisms and human oversight will foster a more comprehensive and fair evaluation process, striking a balance between automated and human evaluations

5.6 Recommendations

Drawing insights from the system's performance and user feedback, we offer valuable recommendations to maximize the "Subjective Answers Evaluation System's" effectiveness: Encourage educators and examiners to adopt the system as a complementary tool to traditional evaluation methods, allowing automated assessments to work in tandem with human evaluations for a comprehensive and balanced assessment approach.

Foster a culture of continuous improvement and collaboration, encouraging the sharing of domain-specific data and knowledge to enhance the system's capabilities in evaluating subject-specific responses.

Provide clear guidelines to students on how to structure their answers, aligning with the system's evaluation criteria, and empowering them to optimize their responses for automated assessments.

Engage in ongoing research and partnerships with academia and industry to explore advancements in NLP and machine learning that can further augment the system's capabilities and address emerging challenges in educational evaluations

References

Li, H., and Huang, W. (2020). An overview of machine learning in subjective answer evaluation. 28(4),801-824; Computer Applications in Engineering Education.

Goo, L., Xu, S., and Hoi, Y. (2018). Grading of short answers automatically using machine learning methods. 23753-23762. IEEE Access, 6.

Zhang, H., Wan, S. (2019). an analysis of machine learning techniques for grading short answers. 2114–2137 in Journal of Educational Computing Research, 57(8).

Lai, C. L., Wang, S. Y., and Hwang, G. J. (2015). Efficacious learning techniques are used in a mobiletechnology-enhanced flipped classroom for seamless flipped learning. 2(4), 449-473, Journal of Computers in Education.

Resnik, P., & Resnik, S. (2013). Automatic short answer grading. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1117-1126).

Jovanovich, J., Gauvin, D., Dawson, S., Pardo, and Mirriahi, N. (2017). dashboards for learning analyticsto encourage self-regulated learning in teacher education. In Handbook of Learning Analytics (pp. 369-378). Society for Learning Analytics Research.

Tanja, H., and Aggarwal, A. (2018). utilizing machine learning to assess student responses. Information Technology and Data Science Advances (pp. 313-324). Singapore's Springer.

Pardos, Z. A., & Heffernan, N. T. (2011). (2011). KT-IDEM: Adding item difficulty to the knowledge tracingmodel. In Proceedings of the Fourth International Conference on Educational Data Mining (pp. 111-120).

Baral, C.; Mitra, R. (2014). Open-ended text responses are automatically evaluated using machine learning algorithms. 7th International Conference on Educational Data Mining Proceedings (pp. 327-328).

Chen, X., & Li, X. (2019). (2019). Automatic grading for multi-level programming assignments: Anempirical research on machine learning models. IEEE Access 7, 148881-148894.

Huang, Y. M., and Law, S. S. (2016). A review of the applications of data mining and machine learning forthe prediction of student outcomes. Technology in Society, 46, 90–103.

Chen, G. D., and Lee, Y. H. (2017). identifying the elements that affect pupils' academic success using machine learning. 20(3), 154–166, Journal of Educational Technology & Society.

Zhang, S., and Duane, Y. (2019). Automatic scoring of writing quality for non-native English learners employing a hybrid machine learning approach. Research and Development in Educational Technology,67(1), 55-71.

Chen, Y., Mahler, J. L., and Williams, B. (2017). A comparison of methods for machine learning to forecast college achievement. Journal of education data.