PROJECT PROPOSAL

SMARTGRADE: SEMANTIC ANSWER EVALUATION SYSTEM

Proposed By:
Qurat-ul-Ain Akhter

Problem Statement

In traditional educational systems, evaluating subjective and open-ended responses remains a highly manual and time-consuming process. Teachers and evaluators are often required to review hundreds of answer scripts, leading to significant delays in result processing, increased cognitive fatigue, and the risk of inconsistent or biased grading due to personal perceptions or exhaustion. This manual approach also lacks scalability, particularly in online or large-scale standardized assessments, where thousands of responses must be evaluated within tight timeframes.

Moreover, in many educational institutions—especially in developing regions—there is often no standardized evaluation rubric. This results in unfair assessments, diminishing student confidence and motivation. Teachers may unintentionally favor certain writing styles, overlook grammar and language quality, or fail to assess the depth of understanding beyond surface-level keyword matching. These inefficiencies contribute to significant gaps in learning feedback and hinder the overall effectiveness of the educational evaluation process.

References:

- Automatic Subjective Answer Evaluation
- Subjective Answer Evaluation
- Subjective Answers Evaluation Using Machine Learning and Natural Language Processing
- Automatic evaluation of open-ended questions for online learning. A systematic mapping

Solution

To address the challenges associated with the manual evaluation of subjective and open-ended responses, a comprehensive Automated Answer Evaluation System is proposed. This system leverages advanced Natural Language Processing (NLP) and Machine Learning (ML) techniques to evaluate textual responses in a scalable, consistent, and unbiased manner. The solution is designed to function independently or integrate with existing Learning Management Systems (LMS), educational platforms, or examination software.

Core Components and Features

The Automated Answer Evaluation System comprises four key modules to ensure accurate, scalable, and unbiased assessment of subjective responses:

1. Multi-Dimensional Evaluation Framework
   Evaluates answers across various dimensions including:
   - Content relevance through semantic similarity
   - Conceptual understanding using deep language models
   - Language quality (grammar, spelling, punctuation)
   - Coherence and structure
   - Keyword matching for key term identification

2. AI and NLP Integration
   Leverages advanced NLP techniques such as:
   - Transformer models (e.g., BERT, RoBERTa) for deep contextual understanding
   - Semantic Textual Similarity (STS) to compare answers
   - Named Entity Recognition (NER) to check concept inclusion
   - Text classification for scoring and categorization

3. Automated Scoring and Feedback Generation
   - Dynamic scoring based on customizable rubrics
   - Automated feedback highlighting strengths and weaknesses
   - Explainable evaluation with justification for each score

4. Plagiarism and Originality Detection
   - Detects copied content by comparing answers against existing databases, academic sources, and the web to ensure originality

Project Scope: Semantic Answer Scoring System

Objective

To automatically evaluate and score short- to medium-length subjective answers by analyzing their semantic relevance, keyword usage, and language quality, in comparison to a reference answer, using Natural Language Processing (NLP) techniques.

Core Features

1. Content Relevance Scoring

- Utilize sentence embedding models (e.g., Sentence-BERT or MiniLM) to convert both the student's answer and the reference answer into vector representations.
- Compute cosine similarity between these vectors to measure semantic closeness.
- Generate a semantic score on a predefined scale (e.g., 0 to 5 or 0 to 10).

2. **Keyword Matching**
- Extract important keywords or phrases from the reference answer.
- Check the presence of these terms in the student's answer.
- Assign a partial score based on the number of keywords matched.
  - Example: If 7 out of 10 keywords are present, assign 70% of the maximum keyword score.

3. **Language Quality Score**
- Use tools such as spaCy or LanguageTool to evaluate the grammatical correctness, punctuation, spelling, and sentence structure.
- Assign a language score on a simple scale (e.g., 0 to 2):
  - 0: Poor grammar
  - 1: Acceptable
  - 2: Excellent

4. **Final Score Calculation**
- Compute the overall score using a weighted average of the three components:
  Final Score = 0.6 × Semantic Score + 0.3 × Keyword Match Score + 0.1 × Language Score

Tech Stack

🔹 **Programming Language**
- Python – main language for development

🔹 **NLP & Machine Learning Libraries**
- Transformers (Hugging Face, OpenAI): Load pre-trained models like BERT, RoBERTa, or MiniLM
- Sentence-Transformers: Easily compute sentence embeddings (semantic similarity)
- spaCy: Tokenization, POS tagging, Named Entity Recognition
- LanguageTool: Grammar and language quality checking

🔹 **Supporting Libraries**
- NumPy / SciPy: Cosine similarity, mathematical operations
- Pandas: Data handling, if working with datasets
- Sklearn: If you use basic ML classification (for answer quality tiers)

🔹 **UI**
- Streamlit Build a lightweight web app for input/output

🔹 **Environment**
- Jupyter Notebook / Google Colab For development, testing, and experimentation
- VS Code / PyCharm For full code development environment

6-Week Plan

Week 1: Requirement Analysis and Dataset Preparation

- Define project scope, evaluation criteria, and scoring rubrics.
- Collect or prepare a dataset of subjective questions, model answers, and student responses.
- Set up the development environment and required libraries.

Week 2: Semantic Similarity Implementation

- Integrate a pre-trained embedding model (e.g., Sentence-BERT or MiniLM).
- Compute cosine similarity between student and reference answers.
- Normalize similarity scores to a defined scale (e.g., 0–5 or 0–10).

Week 3: Keyword Matching

- Extract key terms from model answers.
- Match them against student responses and assign partial scores based on coverage.
- Test the scoring logic on multiple examples.

Week 4: Language Quality Scoring

- Use tools like spaCy or LanguageTool to evaluate grammar, punctuation, and structure.
- Score language quality on a simple scale (e.g., 0–2).
- Validate the grammar scoring mechanism with varied input texts.

Week 5: Score Aggregation and Integration

- Combine all scoring components using a weighted formula:
- Final Score = 0.6 × Semantic Score + 0.3 × Keyword Score + 0.1 × Language Score
- Integrate all modules and optionally create a basic interface for testing.
- Run the system on diverse student answers for consistency.

Week 6: Evaluation and Finalization

- Test the system against a set of manually evaluated responses.
- Analyze performance, adjust scoring weights if needed.
- Document the implementation, methodology, results, and limitations.
- Prepare final report and presentation.