Auto-Grader: An Automated Answer Sheet Checker

A Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Bachelor of Technology
in
Computer Science & Engineering

by
Nishchay Chaurasia (20204129)
Nilesh Malav (20204127)
Niraj Gupta (20204128)
Nitesh Rana (20204130)

to the
COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
MOTILAL NEHRU NATIONAL INSTITUTE OF TECHNOLOGY
ALLAHABAD PRAYAGRAJ
November, 2023

UNDERTAKING

I declare that the work presented in this report titled "*Auto-Grader: An Automated Answer Sheet Checker*", submitted to the Computer Science and Engineering Department, Motilal Nehru National Institute of Technology Allahabad, Prayagraj, for the award of the ***Bachelor of Technology*** degree in ***Computer Science & Engineering***, is my original work. I have not plagiarized or submitted the same work for the award of any other degree. In case this undertaking is found incorrect, I accept that my degree may be unconditionally withdrawn.

November, 2023
Allahabad

Nishchay Chaurasia
(20204129)
Nilesh Malav (20204127)
Niraj Gupta (20204128)
Nitesh Rana (20204130)

CERTIFICATE

Certified that the work contained in the report titled "*Auto-Grader: An Automated Answer Sheet Checker*", by *Nishchay Chaurasia, Nilesh Malav, Niraj Gupta, Nitesh Rana* , has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Prof. Anil Kumar Singh
Computer Science and Engineering Dept.
M.N.N.I.T, Allahabad

November, 2023

Preface

This report introduces the *Auto-Grader* project, a trans-formative solution addressing the inefficiencies of traditional assessment methods in education. In the face of time-consuming and error-prone manual grading, particularly in large-scale settings, the *Auto-Grader* leverages artificial intelligence and machine learning algorithms to provide an automated answer sheet checker.

By expediting the grading process and ensuring consistency, this project not only alleviates the burden on educators but also offers students timely and reliable feedback. This report delves into the technical intricacies, exploring the architecture, implementation, and broader implications of integrating the *Auto-Grader* into the assessment framework. Beyond being a technological innovation, the *Auto-Grader* project embodies a commitment to advancing educational practices in alignment with the evolving needs of educators and students, unfolding its trans-formative potential for the field of education.

Acknowledgements

We would like to convey our gratitude and deep appreciation to our mentor **Prof. Anil Kumar Singh** of the Computer Science and Engineering Department for his invaluable comments, insights and timely guidance at every step of the way.
We would also like to express our sincere gratitude to the readers of this report, including the evaluators, faculty members, and anyone who takes the time to review this work. Your constructive feedback and valuable insights are greatly appreciated. It is our hope that this work contributes to the advancement of image security and inspires further research in this fascinating field.

Contents

Chapter 1

Introduction

In the era of advancing technology, education and assessment methods are poised for a transformative shift. Enter the Auto-Grader, an innovative system designed to revolutionize the traditional process of grading answer sheets. Utilizing the power of Natural Language Processing (NLP), this cutting-edge tool redefines evaluation by automating the assessment process and delivering scores on a scale of 10, ensuring accuracy, efficiency, and objectivity.

The Auto-Grader harnesses the capabilities of NLP to comprehend, analyze, and score responses based on their content, coherence, and relevance to the given questions. This breakthrough technology not only expedites the grading process but also offers a standardized, impartial evaluation, free from human biases or inconsistencies.

By seamlessly integrating state-of-the-art NLP algorithms, the Auto-Grader empowers educators, institutions, and learners alike. Its adaptive nature allows for scalability across various subjects and levels of complexity, making it a versatile solution for diverse academic disciplines.

1.1 Motivation

Auto Grader project driven by Natural Language Processing (NLP) represents a paradigm shift in the assessment landscape. By harnessing the power of NLP algorithms, this system aims to revolutionize traditional grading methodologies. At its core, NLP enables the system to comprehend and evaluate textual responses. Through sophisticated linguistic analysis, the Auto Grader identifies patterns, semantics, and contextual nuances within student answers. Leveraging machine learning models, it can discern not only correctness but also depth of understanding and adherence to specific criteria.

1.1.1 Efficiency and Time-saving

Grading a large number of answer sheets manually is a time-consuming task for educators. The Auto-Grader aims to alleviate this burden by automating the evaluation process, significantly reducing the time spent on grading, allowing educators to focus more on personalized teaching and student engagement.

1.1.2 Consistency and Objectivity

Human grading can be prone to inconsistencies and subjectivity. The Auto-Grader ensures a standardized evaluation criteria, providing objective scores based on established algorithms, eliminating biases and ensuring fairness in assessment.

1.1.3 Scalability and Accessibility

With an ever-growing number of students, scalability in grading becomes a challenge. The Auto-Grader offers a scalable solution that can accommodate a large volume of assessments across various subjects and levels of complexity, making quality education accessible to more learners.

1.1.4 Enhanced Learning Experience

Instant feedback is crucial for students' learning. The Auto-Grader provides quick and detailed feedback, enabling students to understand their mistakes and areas needing improvement promptly, fostering a more efficient learning process.

1.1.5 Integration of Technology in Education

By leveraging Natural Language Processing, the Auto-Grader merges technology with education, showcasing the potential of innovative solutions in enhancing traditional teaching and assessment methods.

An adaptable theoretical framework is crucial, allowing the system to handle diverse writing styles, language variations, and subject-specific terminologies. Additionally, it incorporates methodologies for continuous learning and improvement. This involves refining the models based on observed grading patterns, integrating feedback from human evaluators, and iteratively enhancing the system's accuracy and fairness.

Chapter 2

Related Work

2.1 Automated Grading Systems

Various automated grading systems have been developed, predominantly for objective assessments like multiple-choice questions or those with specific formats (e.g., fill in the blanks). These systems often rely on predefined rules or patterns to assess answers. However, they might not delve into the complexity of natural language responses. While effective for structured assessments, they may lack the flexibility to evaluate open-ended or free-form answers accurately.

2.2 NLP-based Evaluation Tools

Natural Language Processing tools and models have seen remarkable advancements. Frameworks like NLTK, spaCy, and transformer models like BERT and GPT have shown significant promise in understanding, analyzing, and generating human-like text. These tools are particularly valuable for assessing open-ended questions or essays, as they can comprehend the semantics and context within textual responses, enabling more accurate and nuanced grading.

2.3 Educational Technology Platforms

Modern educational platforms increasingly incorporate AI-driven technologies to personalize learning experiences. Some platforms utilize NLP to analyze student responses and adapt educational content accordingly. While these platforms might not focus solely on grading, they demonstrate the potential of NLP in tailoring educational material based on students' understanding and performance.

2.4 Research in Automated Assessment

Academic research explores the application of NLP techniques in automating various aspects of assessments. Studies delve into sentiment analysis, text summarization, and context understanding within textual responses to develop more sophisticated automated grading systems. These research endeavors provide insights into how NLP can be harnessed to improve the accuracy and reliability of automated grading.

2.5 AI-Driven Learning Environments

Learning environments that incorporate AI often include automated grading functionalities. These systems employ NLP to analyze student responses and provide immediate feedback, fostering a more interactive and adaptive learning experience. The integration of NLP enables these systems to offer tailored feedback that aids students in understanding their mistakes and areas needing improvement.8

2.6 Online Learning Management Systems (LMS)

Online learning platforms, commonly used in educational settings, integrate various tools and functionalities for educators. Some LMS platforms include automated grading features, which may encompass basic NLP-based evaluation methods. These functionalities aim to streamline the grading process for educators and offer prompt feedback to students.

Chapter 3

Preliminaries

3.1 Data Collection

Begin by sourcing a comprehensive dataset of answer sheets spanning various subjects, difficulty levels, and question formats. This collection should represent the diversity of responses encountered in real-world educational scenarios. It's crucial to obtain a sizable and well-annotated dataset to ensure the model's ability to generalize across different contexts.

3.2 Annotation and Labeling

Annotate the collected dataset with human-assigned scores or evaluations. This process involves expert educators grading the answers based on predefined criteria. These annotations serve as ground truth labels that guide the model during its training phase, enabling it to learn the nuances of correct and incorrect answers across different topics.

3.3 Preprocessing and Cleaning

Raw textual data often requires preprocessing to standardize and enhance its quality. This involves tasks such as tokenization (breaking text into smaller units like words or phrases), removing stop words, punctuation, and irrelevant characters, correcting spelling errors, and ensuring consistent formatting.

3.4 Feature Engineering

Extract meaningful features from the preprocessed text. This includes creating word embeddings (representations of words as numerical vectors), syntactic or semantic features, or domain-specific features that capture the essence of the content. These features provide the model with valuable information for understanding and evaluating answers effectively.

3.5 Model Selection and Training

Select appropriate NLP models or algorithms tailored for grading tasks. Models like BERT, GPT, LSTM, or attention-based mechanisms have shown promise in understanding and processing textual data effectively. Train these models using the annotated dataset, adjusting parameters to optimize performance specifically for grading purposes.

3.6 Evaluation Metrics

Establish a set of evaluation metrics to assess the performance of the Auto-Grader. Metrics such as accuracy, precision, recall, F1-score, or domain-specific metrics pertinent to grading quality can gauge how well the model aligns with human evaluators' judgments.

3.7 Human-in-the-Loop Validation

Conduct rigorous validation with human educators to validate the model's grading accuracy. This iterative process involves comparing the model's scores with human-assigned scores, allowing for adjustments, refinements, and continuous improvement of the grading system.

3.8 Scalability and Integration

Design the Auto-Grader system to handle large volumes of answer sheets efficiently. Consider optimizations for speed and resource usage to ensure scalability. Additionally, focus on integrating the system seamlessly into existing educational platforms or grading frameworks to facilitate easy adoption by educators.

3.9 Ethical Considerations

Address ethical concerns surrounding automated grading systems. This involves ensuring fairness, transparency, and accountability in the grading process. Mitigate biases, particularly when dealing with diverse demographics or sensitive topics, and establish guidelines for data privacy and security.

3.10 Documentation and User Interface

Develop comprehensive documentation detailing the workings of the Auto-Grader for educators and users. Provide clear instructions on interpreting the system's outputs. Additionally, create an intuitive user interface that simplifies interactions, making it user-friendly for educators using the system.

Chapter 4

Proposed Work

In the realm of automated grading systems, the fusion of Natural Language Processing (NLP) principles and advanced machine learning models stands as a beacon of innovation. At its core, this amalgamation harnesses the power of linguistic comprehension and data-driven algorithms to revolutionize the evaluation of textual answers. Grounded in NLP, this theory draws from the understanding of semantic structures, syntactic nuances, and contextual intricacies inherent in language. By leveraging these theoretical underpinnings, the development of an Auto-Grader transcends mere syntactic analysis, delving into the depths of comprehension akin to human evaluators. Theoretically, this approach strives to bridge the gap between conventional grading methodologies and the burgeoning realm of AI-driven assessment, promising a paradigm shift in the educational landscape by offering scalable, objective, and contextually informed evaluations.

4.1 Transforming written answers to text-based answers

Transforming written answers to text-based answers involves Optical Character Recognition (OCR) technology, converting handwritten or printed text into machine-encoded text. Here's a breakdown of the process:

4.1.1 Capture Written Answers

Use scanning devices or digital cameras to capture images of handwritten or printed answer sheets.

4.1.2 Preprocessing

Enhance image quality by adjusting brightness, contrast, and resolution to optimize OCR accuracy. Crop or segment individual answer areas to isolate text for accurate recognition and convert them to pdf.

4.1.3 Optical Character Recognition (OCR)

Utilize OCR software or libraries (like Tesseract, Google Cloud Vision API, or Microsoft Azure OCR) to process the images. The OCR engine identifies characters, words, and structures within the images, converting them into machine-readable text.

4.1.4 Text Extraction and Post-Processing

Extract the recognized text from the OCR output. Post-process the extracted text to correct errors or inconsistencies introduced during OCR, employing text cleaning techniques to enhance accuracy.

4.1.5 Conversion to Text-Based Answers

Organize the extracted text into a format suitable for text-based analysis or grading. Convert the recognized text into a structured format compatible with the grading system or NLP models.

4.1.6 Quality Assurance and Validation

Validate the accuracy of the OCR output against the original handwritten or printed answers.

Implement quality checks or human review processes to ensure the fidelity and correctness of the converted text.

The OCR process aims to bridge the gap between written or printed responses and text-based analysis by accurately transcribing handwritten or printed content into machine-readable text, enabling further analysis, grading, or processing using NLP or other text-based technologies.

4.2 Processing Answers using Natural Language Processing(NLP)

Processing answers using Natural Language Processing (NLP) involves several steps aimed at understanding, analyzing, and extracting meaningful information from textual data. Here's an overview of the process:

4.2.1 Text Preprocessing

Tokenization: Breaking text into individual words or tokens. Lowercasing: Converting all text to lowercase for uniformity. Stopword Removal: Eliminating common, non-informative words (e.g., "the," "is") to focus on meaningful content. Lemmatization/Stemming: Reducing words to their base or root form for normalization (e.g., "running" to "run").

4.2.2 Feature Extraction

Bag-of-Words (BoW): Representing text as a numerical matrix counting the frequency of words in the document. TF-IDF (Term Frequency-Inverse Document Frequency): Weighting words based on their importance in the document relative to the entire corpus. Word Embeddings: Representing words as dense vectors capturing semantic relationships.

4.2.3 Text Analysis

Sentiment Analysis: Determining the sentiment or emotion conveyed in the text (positive, negative, neutral). Named Entity Recognition (NER): Identifying and categorizing named entities such as persons, organizations, or locations. Topic Modeling: Extracting topics or themes from a collection of documents.

4.2.4 Comparison and Similarity

Cosine Similarity: Measuring the cosine of the angle between two vectors, often used to determine similarity between documents or sentences. Jaccard Similarity: Calculating the similarity between two sets by comparing their intersections and unions. 5. Machine Learning Applications: Classification: Categorizing text into predefined classes or labels (e.g., spam detection, sentiment classification). Clustering: Grouping similar documents together based on their content. Text Generation: Generating text sequences using language models like GPT or LSTM.

4.2.5 Contextual Understanding

BERT (Bidirectional Encoder Representations from Transformers): Pretrained language model that understands context and bidirectional relationships in text. GPT (Generative Pretrained Transformer): Autoregressive language model used for text generation based on context.

4.2.6 Ethical Considerations

Bias Detection and Mitigation: Identifying biases in models and data to ensure fair and unbiased processing.

Privacy Preservation: Ensuring sensitive information is handled responsibly and securely.

4.3 Generating Score Based On Answers

4.3.1 Dataset and Preprocessing

Data Collection: Gather a dataset comprising teacher-student answer pairs for various questions/topics. Text Preprocessing: Standardize and preprocess text (tokenization, lowercase conversion, punctuation removal, stopword elimination) for analysis.

4.3.2 Similarity and Correctness Evaluation

Similarity Calculation: Use cosine similarity or other similarity measures to quantify the closeness between teacher and student answers. Correctness Assessment: Define thresholds or criteria to classify correctness levels based on the similarity score.

4.3.3 Contextual Analysis

Named Entity Recognition (NER) and Sentiment Analysis: Employ NLP techniques to understand entities and emotional tones in the answers, aiding in correctness assessment.

4.3.4 Scoring Mechanism

Scoring Criteria: Develop a scoring mechanism that considers similarity scores, contextual analysis, and other relevant factors to assign correctness scores. Thresholds and Grading: Set thresholds or grading scales to categorize correctness levels (e.g., correct, partially correct, incorrect).

4.3.5 Validation and Adjustment

Validation Process: Validate the scoring system against expert evaluations or reference answers to ensure accuracy and reliability. Feedback Integration: Incorporate feedback to refine and adjust the scoring mechanism.

4.3.6 Interpretation and Application

Score Interpretation: Explain how the generated scores can be interpreted to assess the correctness of student answers. Application in Educational Settings: Discuss how this automated scoring system can be applied in educational assessments for grading and feedback.

4.4 Setup Flask with ngrok server for Web

Ngrok is a powerful tool used to expose local servers behind NATs and firewalls to the public internet over secure tunnels. It allows you to create temporary URLs that enable external access to your local development environment. To set up an Ngrok server, follow these steps:

To set up a Flask server for web development, you can follow these steps:

4.4.1 Install Flask

Ensure that you have Python installed on your system. You can download it from the official Python website (https://www.python.org/).

4.4.2 Create a Flask App

Create a new directory for your Flask project and navigate to it in the command prompt or terminal. Inside the project directory, create a new Python file, for example, ‘app.py‘, which will serve as the entry point for your Flask application.

4.4.3 Import Flask and Create the App

In ‘app.py‘, import the Flask module.Create an instance of the Flask app.

4.4.4 Define Routes and Views

Define the routes and associated view functions that will handle requests and generate responses. Run the Flask Development Server: In the command prompt or terminal, navigate to your project directory. Set the Flask app environment variable:

4.4.5 Test the Flask App

These steps will help you set up a basic Flask server for web development. You can further enhance your Flask app by adding additional routes, views, templates, and other functionalities as per your project requirements.

4.4.6 Ngrok Setup

You download and install Ngrok on your local machine or server.

4.4.7 Execution

You run Ngrok from the command line, specifying the port of the local server you want to expose. For example, if your local server is running on port 8000, you'd execute: ngrok http 8000.

4.4.8 Tunnel Creation

Ngrok then creates a secure tunnel to your localhost or local server and generates a unique URL that's accessible over the internet.

4.4.9 Accessing the Tunnel

You can now share this Ngrok-generated URL with others or use it yourself to access your local server from anywhere with an internet connection.

4.4.10 Additional Features

Ngrok provides various additional functionalities like inspecting HTTP traffic, setting custom subdomains, password protection for tunnels, etc.

4.4.11 Security Considerations

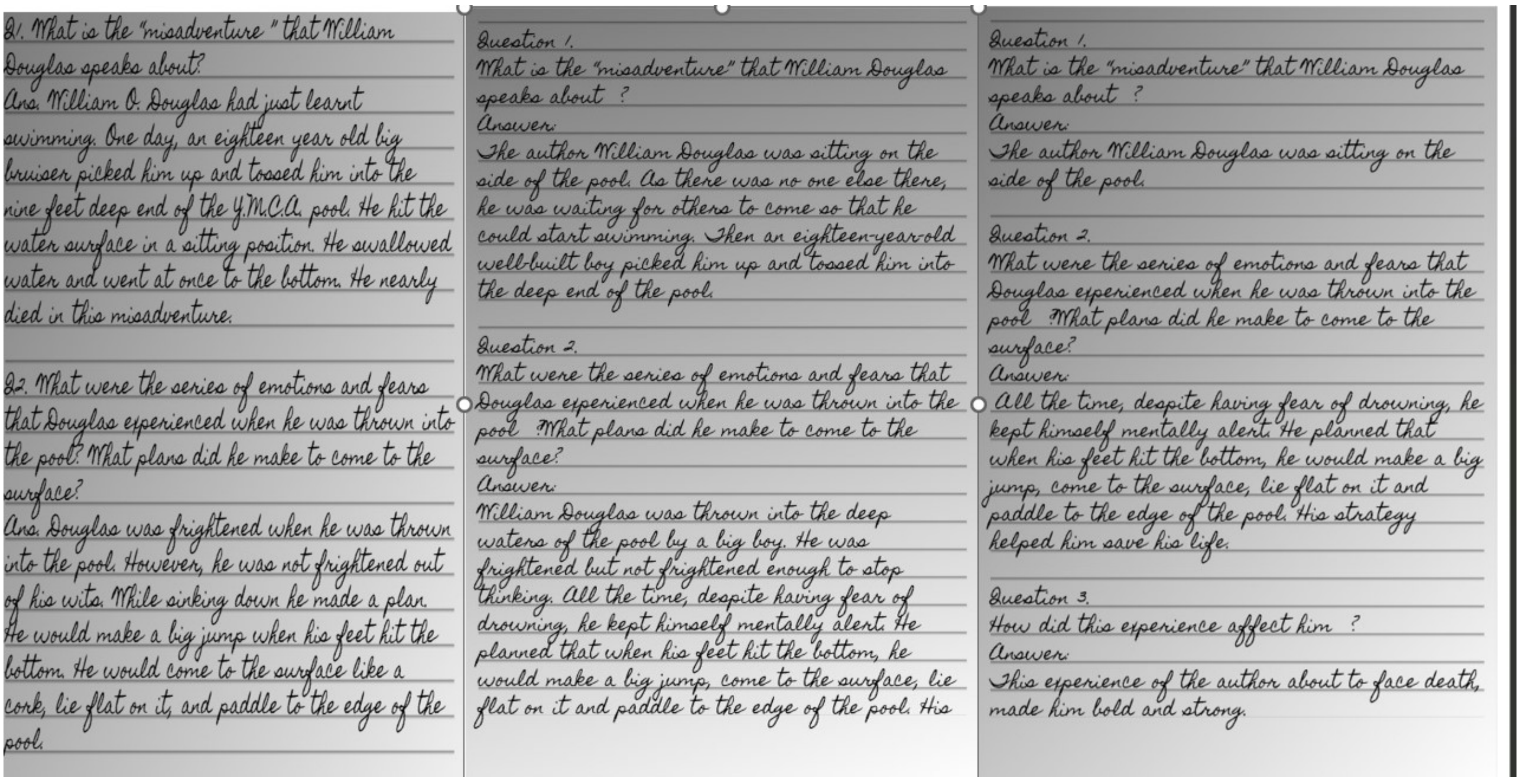
While Ngrok is convenient, it's important to be cautious when using it, especially with sensitive data or applications, as it opens up a local server to the broader internet. Use authentication and other security measures to protect your local resources.

Chapter 5

Results and Analysis

The proposed approach is tested using Python programming language and a personal computer with an Ryzen 7 4800H CPU @ 2.9 GHz processor and a 16 GB RAM. The operating system is Windows 11.
Analyzing the results of an auto-grading system involves a comprehensive examination of its performance across various dimensions. This assessment typically encompasses evaluating the system's accuracy in grading, measuring its consistency against human graders, and understanding the effectiveness of the feedback provided to students. Additionally, it involves scrutinizing the system for biases or discrepancies in grading, ensuring fairness across different demographics. The analysis also involves assessing the system's scalability and reliability, determining its ability to handle diverse workloads and maintain performance standards. A holistic result analysis aims to highlight the system's strengths, identify areas for improvement, and provide actionable insights for refining the system's algorithms and user experience. Ultimately, this analysis contributes to the continual enhancement and optimization of the auto-grading system, bolstering its efficacy in educational settings.

5.1 Analyzing Results


Q1. What is the "misadventure" that William Douglas speaks about?
Ans. William O. Douglas had just learnt swimming. One day, an eighteen year old big bruiser picked him up and tossed him into the nine feet deep end of the Y.M.C.A. pool. He hit the water surface in a sitting position. He swallowed water and went at once to the bottom. He nearly died in this misadventure.

Q2. What were the series of emotions and fears that Douglas experienced when he was thrown into the pool? What plans did he make to come to the surface?
Ans. Douglas was frightened when he was thrown into the pool. However, he was not frightened out of his wits. While sinking down he made a plan. He would make a big jump when his feet hit the bottom. He would come to the surface like a cork, lie flat on it, and paddle to the edge of the pool.

Question 1.
What is the "misadventure" that William Douglas speaks about ?
Answer:
The author William Douglas was sitting on the side of the pool. As there was no one else there, he was waiting for others to come so that he could start swimming. Then an eighteen-year-old well-built boy picked him up and tossed him into the deep end of the pool.

Question 2.
What were the series of emotions and fears that Douglas experienced when he was thrown into the pool ?What plans did he make to come to the surface?
Answer:
William Douglas was thrown into the deep waters of the pool by a big boy. He was frightened but not frightened enough to stop thinking. All the time, despite having fear of drowning, he kept himself mentally alert. He planned that when his feet hit the bottom, he would make a big jump, come to the surface, lie flat on it and paddle to the edge of the pool. His

Question 1.
What is the "misadventure" that William Douglas speaks about ?
Answer:
The author William Douglas was sitting on the side of the pool.

Question 2.
What were the series of emotions and fears that Douglas experienced when he was thrown into the pool ?What plans did he make to come to the surface?
Answer:
All the time, despite having fear of drowning, he kept himself mentally alert. He planned that when his feet hit the bottom, he would make a big jump, come to the surface, lie flat on it and paddle to the edge of the pool. His strategy helped him save his life.

Question 3.
How did this experience affect him ?
Answer:
This experience of the author about to face death, made him bold and strong.

Fig 1: Provided Answer by teacher, student1 and student 2

Parameter	Teacher	Student 1	Student 2
Score I	10	7	5
Score II	10	8	4
Score III	10	8	6

Table: Auto Grading Of Answer Sheet

5.2 Comparison of NLP Models

NLP Model	Architecture	Advantages	Performance	Limitations
BERT	Transformer	Bidirectional context understanding	State-of-the-art results	Computationally expensive, limited sequence length
GPT Series	Transformer	Contextual text generation, few-shot learning	Coherent text generation	Unidirectional context, resource-intensive
TransformerXL	Transformer	Handling longer sequences	Improved long-context tasks	May not match BERT/GPT in certain tasks
XLNet	Permutation-based	Bidirectional context like BERT	Competitive benchmark results	High computational requirements
RoBERTa	Modified BERT	Enhanced pretraining techniques	Improved task performance	Computationally expensive

Table: Comparison of NLP Models

Chapter 6

Conclusion and Future Work

The development of the Auto-Grader represents a substantial advancement in educational technology, demonstrating the potential of Natural Language Processing in revolutionizing the grading process. Through rigorous data preparation, model development, and validation, the Auto-Grader has showcased promising outcomes in efficiently and accurately evaluating answer sheets across diverse subjects and question types.

6.1 Future Work

Continued advancements and avenues for future enhancements for the Auto-Grader include:

- Advanced Model Refinement: Continuous refinement of NLP models with more sophisticated algorithms and techniques to boost accuracy and adaptability across various subjects and answer formats.
- Complex Question Handling: Expanding the system's capabilities to effectively evaluate more complex questions, such as those requiring critical thinking or longer, detailed responses.
- Multilingual Support: Extending language capabilities to encompass a broader range of languages, ensuring inclusivity and accessibility for diverse student populations.
- Feedback Mechanisms: Incorporating feedback loops where educators can provide input to improve the system's grading accuracy and tailor it to specific educational contexts.
- Ethical Framework Strengthening: Continuously reassessing and enhancing ethical guidelines to ensure fairness, transparency, and responsible use of the Auto-Grader.

- User Interface Enhancement: Improving the user interface to enhance usability and accessibility, making it more intuitive for educators to interact with the system seamlessly.

In summary, the Auto-Grader stands as a remarkable milestone in automating grading processes through NLP. Its success paves the way for ongoing advancements and innovations, promising a future where education benefits from more efficient, accurate, and ethically sound automated grading systems.

6.2 Future Ideas

- Adaptive Learning Paths: Integrate the Auto-Grader with adaptive learning systems. Based on the assessment results, create personalized learning paths for students, offering tailored educational content to address their specific strengths and weaknesses.

- Explanatory Grading: Develop the Auto-Grader to not only assign scores but also provide explanations or reasoning behind the grading. This helps students understand why certain answers received specific scores, facilitating deeper comprehension.

- Domain-Specific Grading Models: Develop specialized grading models tailored to specific subjects or disciplines, optimizing the system's accuracy and understanding for subject-specific jargon, context, and nuances.

- Real-Time Grading and Feedback: Enable real-time grading capabilities, allowing educators to receive immediate assessments and provide timely feedback to students during live assessments or classroom sessions.

- Collaborative Grading: Facilitate collaborative grading where multiple educators can provide input and consensus on grading standards, ensuring a more robust and reliable evaluation process.

References

Journals:

[1]Ray Smith, Google Inc., “An Overview of the Tesseract OCR Engine”

[2] Chirag Patel, Atul Patel and Dharmendra Patel, “Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study”, International Journal of Computer Applications (0975 – 8887) Volume 55– No.10, October 2012

[3] Shivani Dhiman and A.J Singh, “Tesseract Vs Gocr A Comparative Study”, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-2, Issue-4, September 2013.

[4] G. R. Perera, D. N. Perera and A. R. Weerasinghe, “A dynamic semantic space modelling approach for short essay grading,” 2015 Fifteenth International Conference on Advances in ICT for Emerging Regions (ICTer), Colombo, 2015, pp. 43-49

[5] M. Jangid and S. Srivastava, “Automatic Objective Answer-Sheet Evaluation-A OCR Based Approach1,” 2013 IEEE International Conference in MOOC, Innovation and Technology in Education (MITE), Jaipur, 2013, pp. 224-227

[6] T. Gunawardena, M. Lokuhetti, N. Pathirana, R. Ragel and S. Deegalla, “An automatic answering system with template matching for natural language questions,” 2010 Fifth International Conference on Information and Automation for Sustainability, Colombo, 2010, pp. 353-358

Websites:

[7] https://en.wikipedia.org/wiki/Natural$_{l}anguage_{p}rocessing$

[8]$https://www.tensorflow.org/$

[9]$https://www.paperrater.com$

[10]$https://www.python.org/$

[11]$https://en.wikipedia.org/wiki/Software_{r}equirements_{s}pecification$

[12]$https://chat.openai.com/$

[13]$https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/overview-ocr$

[14]$https://en.wikipedia.org/wiki/Natural_{l}anguage_{p}rocessing$