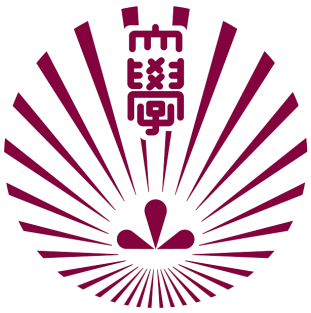
Bridging Human Insight and AI in Education: Teacher Observations for Academic Performance Prediction and Retrieval-Augmented Assessment

メンナトアッラヒ ファティーン



Kyushu University

Graduate School of Information Science and Electrical Engineering
Department of Information Science and Technology

Bridging Human Insight and AI in Education: Teacher Observations for Academic Performance Prediction and Retrieval-Augmented Assessment

By

Menatallah Fateen

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Supervised by

Associate Professor Tsunenori Mine

March 2025

Menatallah Fateen

Bridging Human Insight and AI in Education: Teacher Observations for Academic Performance Prediction and Retrieval-Augmented Assessment

Supervisor: Associate Professor Tsunenori Mine

Advisory Committee: Associate Professor Tsunenori Mine, Professor Yutaka Arakawa, Associate Professor Shogo Fukushima and Associate Professor Daisuke Ikeda

Kyushu University

Graduate School of Information Science and Electrical Engineering

Department of Information Science and Technology

In the name of Allah, the Most Gracious, the Most Merciful

Abstract

Educational assessment often reduces student performance to numerical scores. While these scores are useful for benchmarking, they fail to capture the deeper context of student learning observed by educators or provide the detailed, formative feedback essential for learning. Similarly, AI tools that aim to predict student performance or automatically assess answers often focus on quantitative metrics only. This thesis explores how integrating human expertise with AI-based tools can address this gap, focusing on two core areas: student performance prediction and automatic short-answer scoring with feedback.

The first contribution presents the Teacher Observation for Performance Prediction (TOPP) framework, which enhances student performance prediction by leveraging the rich insights in teacher observation reports. Using advanced representation learning, TOPP captures qualitative insights from teacher comments and integrates them with traditional academic metrics. Evaluated across four modeling approaches, TOPP consistently improved prediction accuracy—achieving up to a 32% increase in performance—demonstrating how effectively human expertise can enhance AI-driven educational assessment.

The second contribution advances automatic short answer scoring (ASAS) through two complementary studies. The first study compares instance-based and reference-based scoring methods, examining their effectiveness in both traditional and zero-shot learning scenarios. While instance-based methods achieve higher accuracy in traditional scenarios, reference-based approaches maintain more consistent performance across both contexts—a crucial insight for deploying ASAS systems in data-scarce environments.

Building on these insights, we introduce a novel retrieval-augmented generation framework that dynamically incorporates human-scored examples to provide both scores and detailed feedback (ASAS-F-RAG). By aligning

with human judgment while minimizing fine-tuning requirements, our approach outperforms traditional baselines—achieving 9% higher accuracy on unseen questions and 1% on unseen answers—while reducing computational demands.

This thesis establishes new methodologies for meaningfully integrating human expertise into AI-powered educational assessment. Through teacher observations and retrieval-augmented techniques, we demonstrate how AI systems can better capture educational context, enhance prediction accuracy, and provide detailed feedback—advancing toward more responsive learning environments that benefit both educators and students.

Acknowledgements

All praise is due to God, the Lord of the worlds. First, I would like to thank my supervisor, Prof. Tsunenori Mine. His constant guidance, support and encouragement over the past 5 years have been invaluable. Not only did I learn from his knowledge, but I learned a lot from his kind and patient personality. I am grateful for his exceptional confidence in my abilities and his encouragement. I consider myself very fortunate to have had the opportunity to work with him and learn from his expertise.

I would also like to thank Prof. Yutaka Arakawa, Prof. Shogo Fukushima and Prof. Daisuke Ikeda for their continuous support and guidance throughout my research.

I want to acknowledge my colleagues and labmates whom I have had the pleasure to work with. Despite my limited time in the lab, primarily working remotely, I have always felt welcomed and supported by them. Every discussion with them was valuable to me, broadening my perspective and enriching my research experience.

A special mention goes to Ms. Fukuda Yuko, Ms. Hiranaka Yukiko, and Ms. Yoshimoto Takayo whose efforts have ensured that I could focus on my research without worrying about the administrative tasks.

To my husband, Mostafa Rushdi, who has been my source of strength, and my children, who have been my source of joy and meaning, I am forever thankful.

I am grateful to my parents, my role models, whom I owe everything to. I will never be able to repay them for their love, support and guidance.

Contents

List of Figures

List of Tables

Acronyms

1 Introduction

1.1 Background

Imagine two students who both scored 75% on their mathematics exam. On paper, their performances appear identical. Yet, one student demonstrated remarkable improvement after struggling all semester, while the other's score signifies an unexpected decline. Their teacher, who has observed their daily efforts and challenges, understands these distinctions—insights that the numerical score alone fails to capture. This scenario illustrates a challenge in educational assessment: the gap between what we quantify through scores and what we comprehend through human observation.

The limitations of purely quantitative assessment have become increasingly apparent in modern education [1]. While numerical scores provide clear benchmarks, they often fail to capture the complex narrative of student learning and development. A student's test score might tell us where they stand, but it doesn't tell us how they got there or where they might be headed. This limitation has sparked growing interest in more holistic assessment approaches that combine numerical measures with qualitative insights and detailed feedback [2].

The integration of Artificial Intelligence (AI) in education offers new possibilities for addressing these assessment challenges. Advances in deep learning and natural language processing have enabled the development of sophisticated systems capable of analyzing student work at scale. These AI-powered educational tools, ranging from intelligent tutoring systems to automated scoring platforms, are transforming both summative and formative assessment. AI's potential to provide immediate, personalized feedback is particulary promising since it can offer students timely insights into their learning progress, helping them identify strengths and weaknesses and adjust

their learning strategies accordingly. This is a crucial element of effective formative assessment that has been constrained by practical limitations [3].

However, the application of AI in educational assessment presents both opportunities and challenges. The effectiveness of AI systems is fundamentally bounded by their underlying methods and training data. For instance, student performance prediction models, while powerful tools, often rely primarily on numerical assessment scores as input features, thereby reducing complex learning experiences to quantitative metrics. Similarly, automatic scoring systems typically employ classification or regression techniques that, while capable of assigning scores to student responses, fail to provide explanations or reasoning for these scores. This lack of transparency and interpretability, while computationally efficient, fails to capture the nuanced understanding that experienced educators bring to assessment.

To bridge these gaps, there is a need to develop AI systems that effectively integrate qualitative insights with quantitative data, enhancing the accuracy, usability, and interpretability of educational assessments. By leveraging human expertise in the form of teacher observations and combining it with advanced AI techniques, we can create tools that not only predict student performance more accurately but also provide meaningful feedback to support learning.

This thesis aims to address these challenges by exploring how human expertise can enhance AI applications in education. Specifically, it investigates methods for integrating qualitative teacher observations into student performance prediction models and develops a system for automatic short-answer scoring that align with human judgments while providing detailed feedback. Through this work, the thesis contributes to bridging gaps in educational AI, demonstrating how combining human insight with AI can create more effective and supportive learning environments.

1.2 Research Objectives and Questions

This thesis aims to advance AI applications in education by addressing two core objectives, focusing on how human expertise can enhance the accuracy, usability, and interpretability of AI tools for classrooms.

The first objective seeks to enhance student performance prediction models by integrating qualitative insights from teacher observations with traditional academic metrics. Performance prediction models typically rely on assessment methods such as previous test scores as predictive features for predicting future academic outcomes. However, the utilization of qualitative teacher observations, which provide rich contextual insights into students' academic progress, has no been explored.

Teacher reports as a systematic assessment method captures nuanced information that numerical data alone cannot capture, offering a more comprehensive view of student progress. This research develops and applies advanced representation learning techniques to effectively combine these qualitative insights with traditional quantitative inputs, to obtain improved predictive accuracy. The study addresses the following research question: **TQ1: How can representation learning techniques effectively integrate qualitative teacher observations with traditional academic metrics to improve student performance prediction accuracy?** To achieve this, four distinct models are developed and evaluated under the Teacher Observation Performance Prediction (TOPP) framework.

The second objective explores the challenges of automatic short-answer assessment, focusing on improving both scoring accuracy and feedback quality. Current methods often rely on extensive training data and struggle to provide actionable, context-specific feedback. This research investigates the merits of instance-based and reference-based scoring methods, particularly in data-scarce contexts, and introduces a retrieval-augmented generation system for automatic short-answer scoring and feedback generation. This novel system aims to align scoring with human judgments while delivering meaningful feedback to enhance student learning. Two key research questions guide this exploration: **TQ2.1: What are the merits of instance-based and reference-based methods in automatic short answer scoring particularly**

in terms of their performance in traditional versus zero-shot learning scenarios? and **TQ2.2: How can we develop a system for automatic short answer scoring that aligns with human judgements and provides detailed, context-specific feedback?**

By addressing these objectives and answering these research questions, this thesis contributes to bridging gaps in educational AI. It demonstrates how human expertise can be leveraged to enhance AI tools, creating impactful solutions. Ultimately, this research aims to provide educators with systems that reduce administrative burdens, improve teaching quality, and foster more supportive learning environments.

1.3 Thesis Outline

This thesis is organized into several key chapters that explore the integration of AI in education, focusing on student performance prediction and automatic short-answer grading.

In Chapter 2, we provide a comprehensive review of the literature on representation learning and generative models. This chapter lays the theoretical foundation for the methodologies employed in the subsequent chapters.

Chapter 3 introduces the TOPP framework, which explores the integration of teacher observation reports into student performance prediction models. The chapter describes the dataset, feature extraction methods, and machine learning models used in TOPP studies. Additionally, it discusses the evaluation metrics employed across experiments to assess the impact of incorporating teacher reports on predictive accuracy.

Chapter 4 presents the foundational models for TOPP: the Reports Model and the Students Model. These models generate report-level and student-level predictions, demonstrating the feasibility and effectiveness of incorporating qualitative teacher observations.

Building on this foundation, Chapter 5 outlines the methodology for extracting linguistic, psychological, and sentiment features from teacher comments

and demonstrates how these enriched features enhance the predictive power of the model.

Chapter 6 introduces an alternative approach to integrating teacher observation reports using representation learning techniques. Specifically, it details the training of a deep Siamese neural network to learn sentiment-based representations from teacher comments, further improving the performance of prediction models.

Chapter 7 shifts focus to Automatic Short Answer Scoring (ASAS), providing a comparative analysis of instance-based and reference-based grading methods. It evaluates these methods in traditional and zero-shot learning scenarios, with particular attention to Arabic short-answer scoring, offering insights for data-scarce contexts.

Chapter 8 describes the development of a modular Retrieval Augmented Generation (RAG) system for automatic short-answer grading. This chapter compares zero-shot and few-shot selection methods to demonstrate the advantage of the proposed RAG approach. It also highlights the system's ability to provide detailed, elaborative feedback to enhance student learning.

Finally, Chapter 9 concludes the thesis with a discussion of the key findings, and recommendations for future research directions.

2 Fundamental Concepts

2.1 Introduction

How can we represent words so that a computer can "understand" them? This fundamental question lies at the heart of Natural Language Processing (NLP). Consider the word “broke.” Its meaning can vary significantly depending on the context in which it is used; it may refer to a state of financial difficulty or represent the past tense of the verb “break.” Distinguishing between these meanings becomes even more complex when considering different usages of the verb, such as in the sentences:

- "I broke the vase."
- "I broke the record."
- "I broke the news."
- "I broke even."

To effectively address this challenge, researchers have developed various representation learning techniques that capture not just the meaning of words but also their contextual relationships. The foundation of this exploration is the distributional hypothesis introduced in 1954, that words appearing in similar contexts tend to have similar meanings [4]. This hypothesis has driven the development of models from Term Frequency-Inverse Document Frequency (TF-IDF), and further to more advanced techniques like Bidirectional Encoder Representations from Transformers (BERT).

2.2 Representation Learning

2.2.1 Static Embeddings

Bag of Words (BoW)

The **Bag of Words (BoW)** model is a fundamental representation technique in NLP that characterizes text as an unordered collection of words. This model simplifies text representation by disregarding grammar and word order while focusing solely on the frequency of words. Each document in a corpus is transformed into a vector, where each element corresponds to a unique word in the vocabulary derived from the entire corpus.

The primary steps involved in constructing a BoW representation begin with vocabulary creation. In this initial step, all unique words in the corpus are identified. For instance, given the sentences “I broke the vase.” and “I broke even.”, the resulting vocabulary can be represented as:

$$\text{Vocabulary} = \{\text{I}, \text{broke}, \text{the}, \text{vase}, \text{even}\} \tag{2.1}$$

Following the creation of the vocabulary, the next step is vector representation. Each document is represented as a vector of word frequencies. Specifically, the vector for the first sentence “I broke the vase.” would be denoted as [1, 1, 1, 1, 0], indicating the frequency of each word in the document. In a similar manner, the vector for the second sentence “I broke even.” would be represented as [1, 1, 0, 0, 1]. Mathematically, the representation of a document D can be expressed as:

$$D = [f_1, f_2, \ldots, f_n] \tag{2.2}$$

where f_i represents the frequency of word w_i in the document.

Despite its straightforward approach to text representation, the BoW model has notable limitations. One critical shortcoming is its disregard for the order of words in a sentence. For example, the sentences “The cat sat on the mat”

and “The mat sat on the cat” would be represented identically in the BoW model, leading to a loss of syntactic and semantic meaning. Additionally, the dimensionality of the vector representation increases with the size of the vocabulary, resulting in high-dimensional and often sparse vectors. This sparsity can pose challenges for machine learning algorithms, as it may lead to inefficiencies in computation and difficulties in generalization.

Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF is a weighting technique that enhances the BoW model by incorporating the significance of words across an entire corpus [5]. This method addresses the limitations of BoW by not only considering the frequency of terms within individual documents but also evaluating their importance in relation to the corpus as a whole.

The computation of the TF-IDF score for a term t in a document d involves two primary components: Term Frequency (TF) and Inverse Document Frequency (IDF).The first component, TF measures how frequently a term t appears in document d. It is calculated as follows:

$$\text{TF}(t, d) = \frac{f(t, d)}{\sum_{t' \in d} f(t', d)} \tag{2.3}$$

where $f(t, d)$ represents the frequency of term t in document d, and the denominator is the sum of the frequencies of all terms in the document, providing a normalized value that reflects the term’s relative importance.

The second component, IDF, quantifies the importance of a term within the context of a corpus by analyzing how frequently it appears across multiple documents. The rationale behind IDF is that terms that occur in many documents are less informative than those that appear in a limited number of documents. Therefore, IDF helps to downweight common terms that provide little value in distinguishing between documents. The IDF for a term t is computed as:

$$\text{IDF}(t) = \log\left(\frac{N}{|\{d : t \in d\}|}\right) + 1 \tag{2.4}$$

In this equation, N represents the total number of documents in the corpus, and $|\{d : t \in d\}|$ counts the number of documents that contain the term t. The logarithmic transformation helps to scale the IDF values, preventing them from becoming too large. The IDF formula stated differs from the standard textbook notation that defines it as $log(\frac{N}{|\{d:t\in d\}|})$. The purpose of adding 1 is to allow terms that have zero IDF (terms appearing in all documents) not to be ignored and it is the same equation used in the sklearn implementation [6].

The final TF-IDF score for a term t in document d is given by the product of the TF and IDF values: $\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$. This score reflects both the frequency of the term in the document and its relative rarity across the corpus, effectively reducing the influence of common words while emphasizing unique terms within each document.

Word2Vec and Glove

Both BoW and TF-IDF are sparse, count-based vectors. Neural word embeddings, particularly Word2Vec introduced by Mikolov et al. in 2013 [7] and GloVe by Pennington et al. in 2014 [8], learned semantic representations by predicting word contexts. Word2Vec's CBOW and Skip-gram models captured word relationships geometrically, while GloVe factorized a co-occurrence matrix. These dense, low-dimensional embeddings enabled operations like king $-$ man $+$ woman $=$ queen, establishing foundations for contextual representations in models like BERT.

2.2.2 Contextual Embeddings

While Word2Vec and GloVe were limited in the sense that they produced static embeddings, the field of NLP quickly realized the potential of using pre-trained embeddings on large datasets. **ELMo (Embeddings from Language Models)** introduced by Peters et al. in 2018 [9] was the first model to generate **contextual embeddings** by training a bidirectional LSTM on a language modeling task. ELMo embeddings captured word meaning based on the entire sentence context, allowing for more nuanced representations.

Transformers

The development of the Transformer model, introduced by Vaswani et al. in 2017 [10], revolutionized natural language processing by providing a highly effective mechanism for capturing long-range dependencies in text without relying on recurrent architectures. Transformers utilize a **self-attention** mechanism that allows each word to attend to every other word in a sentence, enabling the model to capture contextual relationships between words at various distances.

Specifically, the Transformer model consists of an **encoder-decoder architecture**, where the **encoder** processes the input sequence first by adding an embedding to a **positional encoding** which represents the position of the token in the input window. The sequence of tokens are then passed through a **multi-head attention** layer. This layer computes attention scores between all pairs of words in the input sequence, allowing the model to weigh the importance of each word based on its relationship with other words. For a given input sequence of tokens $X = \{x_1, x_2, \ldots, x_n\}$, each token x_i is transformed into three vectors: the query vector Q_i, the key vector K_i, and the value vector V_i. These vectors are created by multiplying the word embeddings by three matrices that were previously trained. The attention score for each token pair (i, j) is computed using the scaled dot-product:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.5}$$

where d_k is the dimension of the query and key vectors, Q and K are the matrices of all query and key vectors, respectively, and V is the matrix of all value vectors. The softmax function normalizes the scores, and scaling by $\sqrt{d_k}$ helps maintain stable gradients.

The output of the final transformer layer is is then passed to the language modeling head, which predicts the next word in the sequence by taking the output of the last token and producing a probability distribution over the vocabulary. More specifically, th head applies an **unembedding** matrix to the output of the final layer to generate the **logits**, which are then passed through a softmax function to produce the probability distribution.

The Transformer was quickly able to replace the usage of **Long Short Term Memory Networks (LSTMs)** and **Recurrent Neural Networks (RNNs)** in many NLP tasks due to its parallelization capabilities and ability to capture long-range dependencies. However, the Transformer model was initially designed for forward language modeling, for example tasks like machine translation, where only the left context is used. The ELMo language model was bi-directional but became outdated due its usage of LSTMs.

BERT

Building on this, BERT introduced by Devlin et al. in 2018 [11] expanded the capabilities of the Transformer to create contextualized word embeddings. BERT was pre-trained on two main objectives, **Masked Language Modeling (MLM)** and **Next Sentence Prediction (NSP)**. MLM enables learning bidirectional representations by masking certain tokens in the input sequence and training the model to predict these tokens. This kind of training where we add noise to the input and have the model remove the noise is referred to as **denoising**. Let $X = \{x_1, x_2, \ldots, x_n\}$ represent the input sequence, where some tokens are masked. For each masked token x_i , the probability of predicting the correct word is calculated as $P(x_i|X_{\text{masked}}) = \frac{\exp(h_i \cdot e_{x_i})}{\sum_{w \in V} \exp(h_i \cdot e_w)}$, where h_i is the hidden representation of the masked token x_i, e_{x_i} represents the embedding of the correct word, and V denotes the vocabulary. The MLM loss is then computed as the cross-entropy between predicted and actual masked tokens, expressed by $\mathcal{L}\text{MLM} = -\sum i \in M \log P(x_i|X_{\text{masked}})$, where M denotes the set of masked positions. Only the masked tokens in BERT, which constitute 15% of the tokens play a role in the loss function.

The second training objective, NSP, is designed to capture relationships between sentences, where BERT is tasked with determining whether two given sentences follow sequentially. The NSP task uses the [CLS] token's embedding, denoted as $h_{[\text{CLS}]}$, to assess this relationship, with the probability of the second sentence following the first given by $(P(\text{isNext}|h[\text{CLS}]) = \text{softmax}(W_{\text{NSP}} h_{[\text{CLS}]} + b_{\text{NSP}}))$, where W_{NSP} and b_{NSP} are weights and bias terms specific to the NSP classifier. The NSP loss is the binary cross-entropy loss defined as $\mathcal{L}_{\text{NSP}} = -\left(y \log P(\text{isNext}|h[\text{CLS}]) + (1-y) \log(1 - P(\text{isNext}|h[\text{CLS}]))\right)$, where y is a binary label indicating if the two sentences are sequential.

Combining the MLM and NSP losses results in BERT's total pretraining loss, which is calculated as $\mathcal{L}\text{BERT} = \mathcal{L}\text{MLM} + \mathcal{L}_{\text{NSP}}$. By jointly minimizing this objective, BERT learns to capture both word-level and sentence-level contexts, providing a robust, bidirectional understanding of language that serves as a powerful foundation for downstream NLP tasks.

Building on BERT, models like **RoBERTa** [12] improved pretraining by removing the Next Sentence Prediction task and expanding the training corpus, while **ELECTRA** [13] introduced an efficient pretraining approach by detecting replaced tokens rather than predicting masked ones. These models refined BERT's architecture to achieve better performance on NLP benchmarks through optimized training techniques.

Sentence Embeddings

Unlike word embeddings, which represent individual words, sentence embeddings aim to capture the meaning of entire sentences. This holistic representation is particularly valuable for tasks that rely on understanding the overall meaning of text, such as semantic search, document retrieval, and text similarity. To construct sentence embeddings from models like Word2Vec, a common approach is to take the average or sum of the word embeddings in a sentence. However, this simple aggregation method often fails to capture complex sentence-level semantics. To extract sentence embeddings from BERT, a common approach is to use the hidden state of the [CLS] token as the sentence representation.

Sentence-BERT (SBERT) [14] is a modification of the original BERT model designed specifically for generating high-quality sentence embeddings. When comparing sentence pairs for similarity tasks, vanilla BERT or RoBERTa would require processing both sentences separately. SBERT on the other hand employs a siamese and triplet network structure, enabling faster computation by learning semantically meaningful sentence embeddings that can be compared using cosine similarity. This structure reduces the time to find the most similar pair among 10,000 sentences from 65 hours with BERT to approximately 5 seconds with SBERT, while retaining BERT's accuracy. SBERT adds a pooling operation (mean pooling, max pooling, or [CLS] pooling) to derive fixed-size embeddings. It then minimizes cross-entropy loss by concatenating

sentence embeddings u and v with their element-wise difference $|u - v|$, weighted by a trainable parameter W_t.

2.3 Generative Models

Generative models or generative AI are a class of models that can create realistic images and videos or generate human-like text that resembles the data on which they were trained. Unlike **discriminative models** that focus on distinguishing between classes or categories, generative models aim to learn the underlying distribution of the data and generate new samples that are indistinguishable from the original data.

2.3.1 Foundational Approaches

The development of generative models in AI began with foundational ideas like Alan Turing's concept of machine intelligence, which he formalized with the Turing Test [15], and early conversational systems like ELIZA [16], which used basic pattern matching to mimic human dialogue. As machine learning advanced, probabilistic models such as Naive Bayes [17] and Hidden Markov Models (HMMs) [18] became essential tools for tasks like text classification and speech recognition, by modeling data distributions and sequences.

While these early models were interpretable and computationally efficient, they struggled with complex, long-range dependencies. This limitation spurred the development of Recurrent Neural Networks (RNNs) and Long Short-Term Memory networks (LSTMs) [19], which introduced memory mechanisms to capture longer contextual information. LSTMs, in particular, addressed the vanishing gradient problem that affected standard RNNs, enabling the processing of longer text and speech sequences. However, their limitations in efficiency and scalability ultimately led to the creation of the Transformer architecture.

Generative Adversarial Networks (GANs)

The field of generative models saw a substantial advancement with the introduction of **Generative Adversarial Networks (GANs)** by Goodfellow et al. in 2014 [20]. GANs consist of two neural networks: a **generator** that creates realistic samples from random noise and a **discriminator** that attempts to distinguish real data from generated samples. Through an adversarial training process, the generator learns to produce high-quality, realistic data as it competes against the discriminator. GANs quickly became a powerful tool in image and video generation. However, their applications in NLP have been more limited and complex, mainly because text generation requires sequential coherence and contextual understanding over long spans, which GANs were not originally designed to handle effectively.

2.3.2 Large Language Models

Large Language Models (LLMs) represent a paradigm shift in generative AI, due to their ability to process and generate human-like text through massive-scale training on internet-scale datasets. These models, built on the Transformer architecture [10], have revolutionized NLP through their sophisticated understanding and generation capabilities.

Autoregressive Generation

Masked language models like BERT and RoBERTa consist of encoder only architectures where the models predict the word based on both past and future words. While these are early examples of large language models, the term LLMs is often used to refer to **autoregressive models** or **causal language models**. Autoregressive models generate text conditioned on an input piece of text, or **a prompt**, and the LLM continues to generate text one token at a time, conditioned on the previously generated tokens. This autoregressive generation allows the model to capture long-range dependencies and generate coherent, contextually relevant text. This task is referred to as **conditional generation**.

To determine the next word to generate, a simple method is to generate word that are the mostly likely to occur next given the context, which is known as **greedy decoding**. This algorithm makes a choice that is only locally optimal, that is at each step, the output y_t is calculated as $y_t = \arg\max_y P(y|y_{1:t-1}, x)$, where $y_{1:t-1}$ is the previously generated tokens and x is the input. However, more sophisticated **sampling** methods are often used, where the next word is chosen according to its probability in context as defined by the model. Examples of sampling methods include **top-k sampling**, where the model samples from the top k most likely words, and **nucleus sampling**, where the model samples from the smallest set of words that have a cumulative probability of at least p. **Temperature sampling** is a method inspired by thermodynamics where systems at higher temperatures are more likely to explore many possible states, and systems at lower temperatures are more likely to explore a subset of lower energy states. In the context of text generation, temperature sampling scales the logits before applying the softmax function, allowing the model to explore more diverse word choices at higher temperatures and generate more deterministic text at lower temperatures.

Pre-training LLMs

The foundation of LLMs is the pretraining phase, in which models are exposed to internet-scale datasets, often comprising billions of words from diverse sources such as books, articles, and web content. During pretraining, the model is asked to predict the next word at each time step t, or **self-supervised learning**, where the model essentially "teaches" itself by minimizing errors in its word predictions. Because the training corpora are so vast, they are likely to contain many examples that cacn be helpful for many different tasks. This process enables the model to learn the underlying structure of language, including syntax, semantics, and context, without the need for task-specific labels. The pretrained model can then be **finetuned** on smaller, task-specific datasets to adapt its knowledge to specific applications or domains.

A striking feature observed in LLMs is their **emergent abilities**. This refers to the model's capacity to handle tasks it was not specifically trained for, such as solving math problems, generating code, and performing zero-shot learning,

where it can respond to tasks it has not encountered before. By scaling up LLMs, not only is performance predictably improved, but abilities that were not present in the smaller models emerge as well [21].

Examples of LLMs

The **Text-To-Text Transfer Transformer (T5)** by Google Research reframes all NLP tasks as text-to-text problems, allowing input and output to be represented uniformly as text [22]. This versatile approach enables T5 to handle tasks like translation, summarization, and classification within a single model architecture by using task-specific prefixes (e.g., "translate English to French:"). With its encoder-decoder Transformer structure, T5 achieves high performance across varied applications, making it a powerful, unified model for diverse NLP tasks.

Developed by OpenAI, the **Generative Pre-trained Transformer (GPT)** GPT models are some of the most well-known autoregressive language models. Beginning with GPT-1, OpenAI introduced a model capable of generating coherent text through autoregressive generation on a large, unlabelled dataset. GPT-2 and GPT-3 followed, scaling up model size significantly and exhibiting a broad range of abilities, including **few-shot and zero-shot learning**. GPT-3, with 175 billion parameters, demonstrated an impressive capacity to perform diverse tasks, from language translation to question answering, without task-specific training data. Recently, GPT-4 extended these capabilities with multimodal inputs, allowing it to process both text and images, expanding the range of potential applications even further.

LLaMA (Large Language Model Meta AI), developed by Meta, is a family of large language models optimized for efficiency and accessibility in research settings [23]. Unlike GPT, LLaMA is designed with a focus on academic use, offering smaller, fine-tunable models that can achieve high performance with fewer computational resources. LLaMA retains the core transformer-based structure but emphasizes parameter efficiency, making it more practical for applications that require adaptable, resource-efficient models.

Mistral, and **Mixtral** developed by the Mistral AI research lab, represent a new generation of open-weight LLMs focused on efficiency and usabil-

ity. Mistral's architecture is optimized for both pretraining and fine-tuning on specific tasks, with models like Mistral 7B offering high performance even with relatively fewer parameters compared to larger models such as GPT-3. This approach has made Mistral especially valuable in scenarios where computational resources are limited but high-quality text generation is required.

2.3.3 Prompting LLMs

Interacting with LLMs through **prompts** has become a popular method for controlling and directing text generation. Prompts are input strings, instructions, that guide the model to generate specific types of text relying on contextual generation. Prompts can contain examples or **demonstrations** that help make the instructions clear. Prompting is also referred to as **in-context learning**-a type of learning that can improve the performance without involving gradient updates to the model. However, to follow natural instructions, LLMs are often then finetuned on an instruction dataset to align the model with user's expectations. This process is known as **instruction tuning**.

Finding the most effective prompt for a given task is often an iterative process, requiring experimentation to achieve some user goal. This process is known as **prompt engineering**. When prompting an LLM for different tasks, a **prompt template** is first designed which includes task-specific text and placeholders for input to be processed. After the autoregressive decoding process, the output can be used directly or an answer can be extracted from the output.

To improve the performance of a prompt, demonstrations or labeled examples are often included in the prompt. This technique is referred to as **few-shot prompting**. The number of these demonstrations do not need to be large. In fact, adding too many examples, similar to overfitting, can lead to poor generalization. To select demonstrations, a labeled training set is used. One of the popular ways to select demonstrations is programmatically by choosing the ones that most increase the performance of the model. Recent research propose methods to algorithmically optimize prompts and find the optimal set

of demonstrations by searching over a large space of possible demonstrations [24], [25]. Another approach is to use demonstrations that are similar to the current input by dynamically retrieving them based on their similarity to the current example. Many prompting techniques have also been promposed to improve the performance of LLMs on a wide range of tasks. For example, **chain-of-thought prompting** [26] aims to imropve performance on difficult reasoning tasks by encouraging models to break them down into steps.

2.3.4 Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) is a framework that enhances the performance of LLMs by integrating an Information Retrieval (IR) component. While LLMs are trained on extensive datasets, they can struggle to accurately generate responses for niche topics or recent events that fall outside their training data. RAG addresses this limitation by incorporating a retrieval mechanism, allowing the model to pull relevant information in real-time from a predefined document collection or knowledge base. This retrieved content augments the model's prompt, grounding its responses in external data, which is particularly valuable for applications requiring factual accuracy, such as customer support, medical information, and legal advice.

Information Retrieval Systems

An effective **Information Retrieval (IR)** system is essential to RAG's functionality, as it supplies the documents that provide factual grounding for generated responses. In an IR system, a user poses a **query**, and the system retrieves an ordered set of **documents** from a larger **collection**, often using ranking algorithms to prioritize relevance. These documents may vary in form, ranging from web pages to short paragraphs.

Early IR techniques used methods like TFIDF and BM25 to rank documents. TF-IDF scores words giving more weight to unique, informative terms. BM25 extends TF-IDF by introducing parameters for term frequency saturation and document length normalization, improving performance across documents of varied lengths. These techniques became foundational in IR, particularly

for matching exact keywords, but they do not account for synonyms or semantically related terms.

To address the **vocabulary mismatch problem**, where semantically similar words are not matched, BERT and other dense retrieval models are used in IR. These models encode queries and documents into dense embeddings, allowing for retrieval based on contextual similarity rather than exact word matches. However, this approach can be computationally prohibitive, as every document must be re-encoded with each new query.

To balance efficiency and accuracy, some IR systems employ a **bi-encoder** approach, where two separate encoder models process the query and documents independently. This allows for pre-computation of document embeddings, improving retrieval speed, though at the cost of some accuracy compared to the single, contextually-aware encoder.

Another approach, known as ColBERT [27] separately encodes query and document. Instead of encoding the entire query or document into one vector, it encodes each into contextual representations for every token. The BERT representations of each document token can be prestored for efficiency and a score getween the query and the document can be calculated for retrieval. This method achieves a balance between the efficiency of bi-encoders and the accuracy of single-encoder models.

2.4 Summary

NLP aims to develop techniques for understanding the complex, context-dependent nature of human language. Early approaches, such as BoW and TF-IDF, have evolved into sophisticated methods like word embeddings (e.g., Word2Vec, GloVe). The introduction of Transformer-based architectures, particularly BERT, has significantly advanced NLP by enabling the modeling of long-range dependencies and bidirectional context. Transformers have laid the foundation for Large Language Models (LLMs) such as GPT, T5, and LLaMA. These models, trained on vast datasets, excel at text generation but face challenges in maintaining factual accuracy and integrating external

knowledge. Techniques like RAG address these limitations by combining LLMs with retrieval systems to produce grounded, accurate responses.

These advancements in NLP have profound implications for AI in education. Language models and techniques like RAG enable the development of intelligent systems capable of personalized learning, automatic grading, and feedback generation. By leveraging contextual embeddings and retrieval-based enhancements, we can provide more nuanced, accurate, and contextually appropriate support for educational tasks.

3

TOPP: Teacher Observations for Performance Prediction

3.1 Introduction

Predicting student performance has long been a central topic in education, with the goal of enhancing learning outcomes for both students and institutions [28]. Teachers, who play a pivotal role in shaping academic success, traditionally rely on assessments such as exams, homework, and class participation to evaluate student progress. However, these tools are often time-consuming and may not capture the full range of a student's abilities or potential struggles. More importantly, they may not allow for timely interventions to address any issues that arise.

To address these limitations, researchers have explored the use of predictive modeling techniques that incorporate a broad range of data sources, including academic history, student demographics, learning behaviors, and interactions with educational technologies [28]. Most studies have focused on utilizing results of various assessments, such as test scores, peer evaluations, and self-assessments, to predict student performance [29]. However, one assessment technique that has been overlooked as a potential data source of performance prediction models is teacher observation reports.

Teacher observation reports represent a critical, yet underutilized, source of insight in educational assessment. Unlike traditional metrics such as test scores or standardized assessments, these reports offer a window into a student's learning process that is both contextual and continuous [30]. Teachers observe students' day-to-day behaviors, responses, and interactions within the authentic setting of the classroom. As a result, their observations

capture an array of student attributes, from perseverance and collaboration to emotional engagement and adaptability—elements of learning that are frequently difficult to quantify but invaluable for understanding each student's unique learning journey. These detailed observations highlight not just a student's academic performance, but also their emotional development, resilience, and approach to problem-solving. Teacher observation reports can therefore provide a multidimensional view that is rarely visible in standard assessments.

Despite their richness, teacher observation reports are often overlooked in favor of quantitative data that is easier to process and analyze. Yet, when systematically recorded, these reports can offer a depth and flexibility that would be highly valuable for predictive modeling. By including the qualitative aspects of student learning, teacher observations help to address the limitations of standardized assessments, which often fail to reflect individual learning contexts or complex behaviors. Teachers' firsthand insights allow models to incorporate elements like incremental improvements, patterns of engagement, and incidental behaviors that signal both current understanding and future potential. Thus, the inclusion of teacher observation reports in performance models would not merely add additional data; it would enrich the models by embedding them in the real-world contexts in which learning actually occurs.

In this chapter, we introduce **Teacher Observation for Performance Prediction (TOPP)**, a study designed to investigate how teacher reports can be integrated into student performance prediction models. By using this unstructured data, TOPP seeks to enhance the accuracy of predictions, enabling educators to identify at-risk students earlier and allocate resources more effectively. We first describe the current state of literature in the field of performance prediction and identify the gap we aim to fill with TOPP. We proceed with describing the TOPP dataset in addition to the feature sets adopted throughout all TOPP studies. Finally, the common machine learning model employed is described along with the evaluation metrics used across the experiments.

The following chapters explore different methodologies for processing and analyzing teacher observation reports. These include foundational models

in Chapter 4 [31, 32], feature extraction techniques that offer more interpretable insights by quantifying sentiments and psychological features in Chapter 5 [33], and similarity learning in Chapter 6 [34], each offering a distinct approach to representing and utilizing this unstructured data. Through these investigations, TOPP aims to demonstrate the value of teacher reports as a tool for improving predictive accuracy and providing timely insights that benefit both educators and students.

3.2 Related Work

3.2.1 Background

The field of student performance prediction began in the early 1900s with simple psychological assessments [35, 36] and has since grown into a sophisticated area of research. A notable milestone in this evolution came from computing education, where Davies [37] pioneered the use of systems that record how students work through programming tasks on the famous 'rainfall problem' [38]. This shift toward systematic data collection, combined with modern technology, has opened new possibilities for understanding and predicting student performance [39]. The integration of diverse data sources has enriched the predictive power of these models, allowing for more nuanced insights into the factors that influence academic success.

Researchers have pursued performance prediction for several practical reasons: to help students choose suitable majors, to improve university admissions decisions, to provide better academic advising, to identify students who need additional support, and to enhance course design. By identifying students at risk of underperforming early, educators can provide targeted interventions, such as tutoring or mentoring, which help keep students on track. This proactive approach not only benefits individual students but also supports institutions in achieving better overall academic outcomes. Additionally, as educational environments become more diverse, performance prediction allows for more personalized learning paths. By understanding each student's unique strengths and challenges, educators can adapt course content and teaching methods, making learning more effective and inclusive.

This adaptability is especially valuable in large classes or online learning settings, where direct, one-on-one guidance is often limited.

3.2.2 Predictive Factors

The factors used to predict student performance can be broadly categorized into three main types: academic, personal, and socio-economic factors [29]. These categories represent different aspects of a student's background and circumstances that may influence their academic success. Academic factors are the most commonly studied predictors. Score-related measures include previous grades and test scores, while other academic variables include selected majors and completed credits. Course-specific information such as assignment scores and participation rates have also been shown to be strong predictors of performance [40].

Personal factors encompass individual characteristics and behaviors, including basic demographics, study habits, leadership abilities, and communication skills [41]. Researchers have also investigated psychological aspects such as emotional intelligence, resilience, and motivation as potential predictors of academic success [42].

Socio-economic factors consider the student's background and environment, including family characteristics such as size and income level, parents' educational background, and living conditions [43]. Additionally, some studies examine institutional factors, though these are less common. These include considerations such as university resources, program quality, and infrastructure quality [29].

Another way the factors can be categorized into is structured and unstructured data analysis methods [44]. Structured data analysis focuses on numerical and categorical data, including grades, test scores, attendance rates, demographics, course selections, and online learning analytics [45]

Despite the demonstrated potential of unstructured data in performance prediction models, as seen in studies utilizing textual data from student reflections and advisor notes [46, 47, 48], no research to date appears to have leveraged teacher observation reports in this context. The focus has

been largely on student-generated content or institutional records. Therefore, the potential of teacher-generated insights has been overlooked. This gap in the literature provides an opportunity for research to investigate the potential of teacher observation data as a valuable addition to predictive models. By incorporating these reports, TOPP aims to enhance the accuracy and interpretability of performance prediction models.

3.2.3 Performance Measures

Performance prediction studies vary in their target outcomes or measures of prediction. Most approaches are score-based approaches that focus on academic performance indicators such as final grades and GPA. Predicting dropout is another common target, as early identification of at-risk students can help prevent them from leaving the program. According to the study by Hellas et. al [28], around 16% of the studies reviewed were retention-based, while 77% were score-based. The field has also expanded to include other important educational outcomes such as knowledge gains [49]. These metrics reflect the different ways researchers and institutions define and measure student success.

3.2.4 Methods of Prediction

The methodological landscape of performance prediction has evolved significantly, progressing from basic statistical methods to sophisticated machine learning consisting of classification and clustering approaches and data mining [28]. This evolution reflects both the increasing complexity of available data and researchers' efforts to gain deeper insights into student performance. Figure 3.1 summarizes the percentage of studies using different factors, measures of prediction, and methods according to the study by Hellas covering 497 papers from 2010-2018.

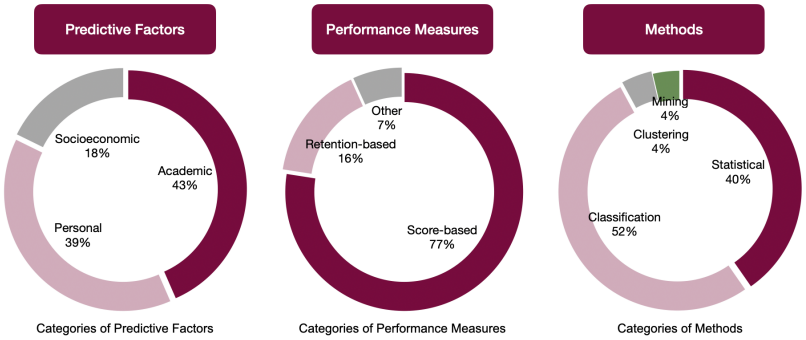


Figure 3.1: Summary of the percentage of studies using different factors, measures of prediction, and methods according to Hellas et al. [28].

Summary

While current research demonstrates the value of both numerical and textual analysis, the potential of using teacher observations for performance prediction remains unexplored. Although teacher observation reports are not necessarily common in all educational settings, they could provide valuable real-time insights into student learning and engagement. This gap has motivated the development of TOPP, which aims to investigate the feasibility and effectiveness of incorporating regular teacher observations into prediction models as a novel approach to understanding and supporting student success.

3.3 Data Description

The teacher observation reports used in our TOPP experiments were sourced from a cram school, or a *juku*, in Fukuoka, Japan, which caters to middle school students. Cram schools, known for their specialized and intensive educational programs, focus on helping students achieve specific academic goals, particularly in preparation for high school or university entrance exams [50]. To ensure confidentiality, all student names and identifying information were excluded from the reports.

In addition to the teachers' qualitative observations, we obtained students' previous exam scores from their regular school assessments. These scores,

alongside the results of simulation exams provided by the cram school, served as performance labels for training and evaluating the machine learning model.

3.3.1 Teacher Observation Reports

The teacher observation reports used in our experiments were collected between May 2020 and December 2021. These reports were primarily intended for the students' parents, offering feedback on their progress from the teachers' perspectives.

Each report contains detailed information such as the topics covered during class, homework completion, an understanding score {0, 30, 60, 80, 100}, and an attitude score {1, 2, 3, 4}, and teacher-written comments. These comments typically consist of around four sentences, averaging 70 words per entry. They generally focus on encouraging the students and emphasizing their strengths, while occasionally pointing out areas in need of improvement. After each class, teachers provide a brief summary of each student's daily progress. Overall, the tone of the comments is supportive and motivating, with occasional references to potential concerns. A sample of the translated teacher observation comments is provided in Table 3.1.

3.3.2 Test Scores

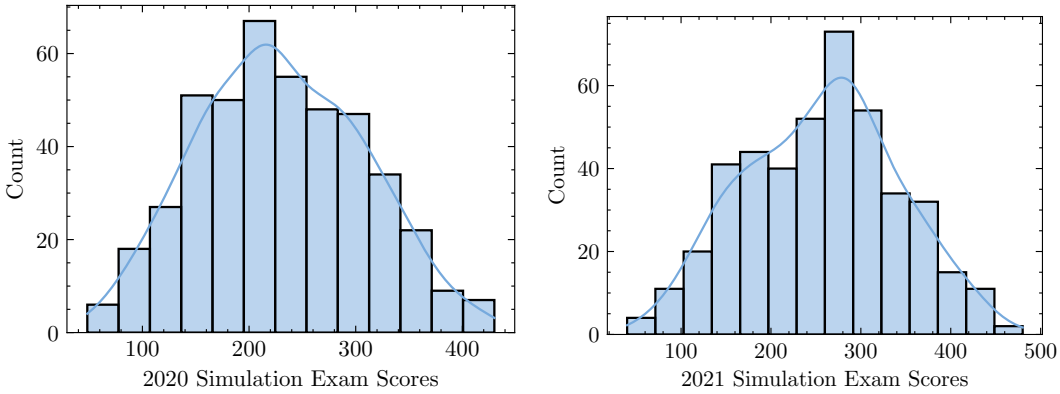
The students attending the cram school are enrolled in various regular schools, and the results of their periodic school exams were recorded and made available. However, only a subset of the cram school students had regular test scores available. These regular test scores were considered traditional features typically used for performance prediction. In our experiments, we compared the performance of using only the regular scores as features versus using both the regular scores and the vector extracted from the teachers' comments.

Since the actual performance on the final entrance exam was not available, we used the students' results from simulation exams conducted prior to

Table 3.1: Sample of teacher observation comments in Japanese with their English translations.

Original Comments (Japanese)	Translated Comments (English)
前回のチェックテストについて、合格できました。本日は試験対策としてワークを進めております。ところどころ苦戦したところがあったようですが、ほとんど自力で解くことができていました。証明もきちんと書くことができていたので、この調子で頑張りましょう。	He was able to pass the last check test. Today, we have been working on the exam preparation. There were some parts that he struggled with, but he was able to solve most of them on his own. He was also able to write the proofs properly, so let's keep up the good work.
本日の学習内容について、理解できているようです。あとは宿題で復習と定着を行えば、今回の内容はばっちりです。分からないところは自分から質問してくれるのでとてもありがたいです。これからも自分から質問してくださいね。	The student seemed to have a good understanding of what they learned today. Now all they need to do is to review and consolidate the content through homework, and they will be all set. I am very grateful that he asks questions when he doesn't understand something. I hope they will continue to ask questions on their own.
テストのやり直しを進めいきました。文法は理解できていますが、英文を読むスピードが少し遅く、時間が足りなかったようです。長文を読むコツを教えながらやり直ししていきました。	We proceeded to redo the test. She understood the grammar, but her reading speed was a little slow and she didn't have enough time. I taught her how to read long sentences and we worked on it.

the actual entrance exam. These simulation exams were administered in September 2020 and again in September 2021. The simulation scores were recorded for each subject, along with the total score. Figures 3.2(a) and 3.2(b) display histograms of the score distributions for both years. As the distributions appear approximately bell-shaped and symmetric around the mean, we assume the scores follow a normal distribution. The standard deviations of the 2020 and 2021 simulation exams were 77.23 and 85.05, respectively, indicating the variability in the students' performance.



(a) September 2020 exam. **(b)** September 2021 exam.

Figure 3.2: Histograms showing the distribution of simulation exam scores.

3.3.3 Feature Selection

In our experimental settings, we adopted three distinct feature sets for the sake of comparison. The first feature set, Feature Set 1 (FS_1), consists of the input vectors derived from the teachers' comments. These are features that represent the qualitative information contained in the teacher observation reports. The method of extraction or representation is explained in detail in each of the subsequent chapters.

The second feature set, Feature Set 2 (FS_2), contains the students' regular test scores. These are obtained from the students' school exams and are used as the more traditional and well-established structured features for performance prediction.

Finally, the third feature set, Feature Set 3 (FS_3), combines FS_1 and FS_2, allowing us to assess whether incorporating teachers' reports improves the prediction accuracy of the model.

3.4 Methodology

3.4.1 Input Representation

In this section, we outline the methodology adopted for integrating teacher observation reports into the performance prediction models developed in the TOPP framework. Our approach consists of several key stages:

Foundational Models: We first establish baseline foundational models where we transform the teacher observation reports into numerical vectors using two main representation techniques. The first technique, the Reports Model, extracts the input vector from a single teacher report using both traditional and BERT models. The second technique, the Student Model, aggregates all teacher reports of a student into a single vector. These baseline models serve as a reference point for evaluating the subsequent models.

Linguistic, Psychological, and Sentiment (LPS) Model: In this model we aim to enhance the accuracy of the performance prediction model by constructing an input vector from the teacher observation reports through extracting different linguistic, psychological and sentiment features.

Sentiment-based Similarity Learning (SBSL) Model: Finally, we investigate the use of similarity learning to improve the performance of the model by applying a contrastive loss function to distinguish between reports with similar sentiment. This approach aims to enhance the model's ability to capture the nuances of the teacher observation reports and improve the prediction accuracy.

3.4.2 Machine Learning Model

In all TOPP models, gradient boosting and a more regularized form of it, XGBoost, were employed as the machine learning models. Gradient boosting is a supervised machine learning technique used for both regression and classification. The term 'boosting' refers to the fact that it aggregates an ensemble of M 'weak learners' which are typically individual decision trees. Given a dataset of n samples $(x_i, y_i)_{i=1}^{n}$, where x_i corresponds to the i^{th} input

value and y_i is the i^{th} observed value, a gradient boosting machine aims to teach a model F to predict $F(x)$ by minimizing its loss function, where x is the input feature vector of all samples. The steps of building a gradient boosting machine are as follows.

Step 1: The model is initialized with a constant predicted value $F_0(x)$ such that:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma) \tag{3.1}$$

where $F_0(x)$ is set to the predicted value γ, that minimizes the sum of the loss function, $L(y_i, \gamma)$ (usually mean-squared error), given the observed value y_i. With this equation, the initial value is set to the average of the observed values.

Step 2: The gradient boosting model is then built in a stage-wise fashion as in other boosting methods where it builds M regression trees. This step is repeated until $m = M$, the number of defined regression trees. To build a tree, the previously predicted values $F(x_i)$ and the observed values y_i are plugged into the negative gradient to calculate the residuals $r_{i,m}$:

$$r_{i,m} = -\left[\frac{\partial(L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad i = 1...n \tag{3.2}$$

A regression tree m is then fit to the $r_{i,m}$ residuals by creating terminal regions $R_{j,m}$ for $j = 1 : J_m$ where J_m is the number of leaves in the m$^{\text{th}}$ tree. The output values $\gamma_{j,m}$ for each leaf j in the tree m, are computed by minimizing the loss function given the previous predictions $F_{m-1}(x_i)$ such that:

$$\gamma_{j,m} = \arg\min_{\gamma} \sum_{x \in R_{j,m}} L(y_i, F_{m-1}(x_i) + \gamma) \tag{3.3}$$

Finally, the predicted values for each sample $F_m(x)$ are updated, based on the previous predictions $F_{m-1}(x)$, the sum of the output leaves $\gamma_{j,m}$ for all the leaves $R_{j,m}$ that a sample x_i can be found in, and by incorporating a learning rate ν to avoid overfitting to the training data. For each sample x, the new prediction is calculated such that:

$$F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{j,m} I(x \in R_{j,m}) \tag{3.4}$$

Step 3: When the number of trees $m = M$, $F_M(x)$ is the output of the gradient boosting model.

In each of the TOPP models, the gradient boosting model was used to predict the students' scores. Specifically, either the sklearn [6] implementation `GradientBoostingRegressor` or the XGBoost [51] library were used. XGBoost is similar to Gradient Boosting where it uses the same gradient boosting principle. However, XGBoost uses a more advanced regularization to control over-fitting which gives better performance. The models were trained on the features extracted from the teacher observation reports, the regular test scores, or a combination of both. The models were then evaluated on the simulation exam scores to assess their predictive performance.

3.4.3 Evaluation Metrics

We evaluate all our experiments using two primary evaluation metrics: Mean Absolute Error (MAE) and Percentage by Tick Accuracy (PTA). MAE is computed using Equation (3.5), where $score_{true,i}$ represents the actual score obtained by student i, and $score_{pred,i}$ denotes the predicted score generated by the model. In the Baseline Models, subject scores are first predicted and then summed to calculate the total score. However, in the LPS Model introduced in Chapter 4, the total score is predicted directly with no subject score prediction since the aim was to create a general and simple model to analyze teachers' reports.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|score_{pred,i} - score_{true,i}| \tag{3.5}$$

PTA is derived from the letter grades assigned to students based on their total scores. The estimated total score is mapped to its nearest corresponding letter grade, as outlined in Table 3.2 [52]. The percentage of grades that fall within x ticks of the actual grades is then calculated. A tick, as defined by [53], represents the difference between two successive letter grades. Specifically, PTA_0 refers to the Percentage by 0 Tick Accuracy, indicating the percentage of instances where the model accurately predicted the letter grade without any error. Conversely, PTA_1 captures the percentage of grades predicted

incorrectly but within one tick of the true letter grade (e.g., predicting an A instead of a B). This metric aligns with those used in previous studies on grade prediction models [54, 53].

Table 3.2: Mapping of letter grades to percentage ranges used for performance prediction.

Grade	S	A	B	C	D	F
%	90-100	80-89	70-79	60-69	50-59	0-49

3.5 Summary

In this chapter, we introduced the TOPP study, which aims to leverage teacher observation reports to enhance student performance prediction models. We reviewed the current state of literature on performance prediction and identified the gap in utilizing teacher observation reports as a data source. We described the TOPP dataset, which includes teacher observation reports and students' regular test scores, and outlined the feature sets used in our experiments. Finally, we presented the machine learning model and evaluation metrics employed in the TOPP study. The following chapters will detail our methodologies and findings.

4

TOPP: Foundational Models

4.1 Introduction

This chapter introduces the foundational models underlying the TOPP framework, which aims to leverage teacher reports and student data for accurate grade predictions. Two primary models are proposed and examined: the Reports Model and the Students Model. The Reports Model treats each teacher's report as an independent input, utilizing a regression approach to predict subject scores based on a student's individual reports. In contrast, the Students Model combines multiple reports into a singular vector representation, capturing the overall performance of each student across different subjects. Both models are evaluated using various feature sets and text representations to determine the most effective approach for predicting student grades. The results demonstrate the potential of these models in enhancing the accuracy of grade predictions

4.2 Experimental Settings

In the Reports and the Students Models, an earlier subset of the data was used for the experiments. Specically, the reports from May 2020 to October 2020 were used which were a total of 11,960 reports for 159 students. Table 4.1 presents the distribution of these reports across the subjects. Mathematics had the highest number of reports, followed by English and Science, while Social Studies and Japanese had comparatively fewer reports.

The total number of reports per student varied based on class attendance, with students receiving an average of 82 reports, ranging from a minimum

Table 4.1: Distribution of teacher reports by subject (May 2020 - October 2020).

	Japanese	Math	Science	Social	English
Number of Reports	1157 (9.7%)	3547 (29.6%)	2428 (20.3%)	1669 (14%)	3159 (26.4%)

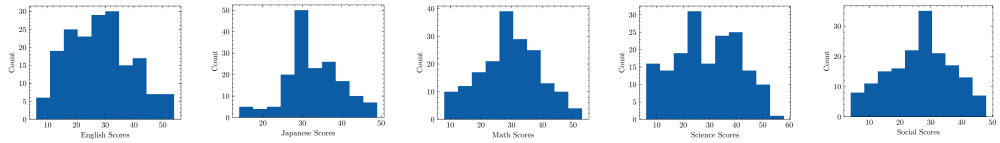


Figure 4.1: Distribution of simulation scores in all subjects

of 24 to a maximum of 206. The variation in the number of reports reflects differences in individual schedules and could influence the predictive power of models depending on report density per student.

To better understand the distribution of the September 2020 simulation exam scores, we created histograms (shown in Figure 4.1) to display the scores for each subject and provided the standard deviation of each subject's scores in Table 4.2.

Feature Selection

In the experimental settings, three feature sets were adopted for the sake of comparison. The first feature set, $\mathbf{FS}_1$, consists of the input vectors derived from the teachers' comments. FS_1 therefore differs between the Reports Model and the Students Model according to the model's design. We describe the specific elements of FS_1 in the following sections. FS_2 represents the student data feature set, in all studies FS_2 exclusively includes the regular scores of the students. However, in the Reports Model, the gender of the student is also included. This was removed in the subsequent models as it had low correlation in feature importance. The third feature set, $\mathbf{FS}_3$, combines the features of each model's FS_1 and FS_2.A comparison of the two models is later presented in Table 4.3.

Table 4.2: Standard deviation of simulation exam scores

	Japanese	Math	Science	Social	English
σ	7.04	9.91	12.24	10.34	10.89

4.3 Methodology

4.3.1 Reports Model

Architecture

In the "Reports Model," each teacher's report is treated as an independent instance. The input vector for this model consists of a single report. For a given subject $s \in$ S, where S = {Japanese, Math, Science, Social Studies, English}, a student i may attend a variable number t of lessons and therefore receive t corresponding reports. To predict the subject score (SubjectScore$_{pred,i,s}$) of student i, each report t is individually fed into a regression model, resulting in an ordered list $X_{i,s,t}$ of predicted scores for that student i. The final predicted score for the subject is calculated using the median, as shown in Formula 4.1.

$$\begin{aligned} \text{SubjectScore}_{pred,i,s} &= Med(X_{i,s,t}) \\ &= \begin{cases} X_i[\frac{t}{2}], & \text{if t is even} \\ \frac{1}{2}(X_i[\frac{t-1}{2}] + X_i[\frac{t+1}{2}]), & \text{if t is odd} \end{cases} \end{aligned} \tag{4.1}$$

The median is chosen over the mean because it is more robust to skewness and outliers. However, if the predictions follow a normal distribution, the median would closely approximate the mean. The total predicted score (TotalScore$_{pred,i}$) for student i is then calculated by summing the predicted scores across all subjects:

$$\text{TotalScore}_{pred,i} = \sum_{s \in S} \text{SubjectScore}_{pred,i,s} \tag{4.2}$$

These steps are illustrated in Figure 4.2, which shows the architecture of the Reports Model. Each subject model processes individual report instances containing teacher comments, a comprehension score (0-30-60-80-100), and an attitude score (1-2-3-4).

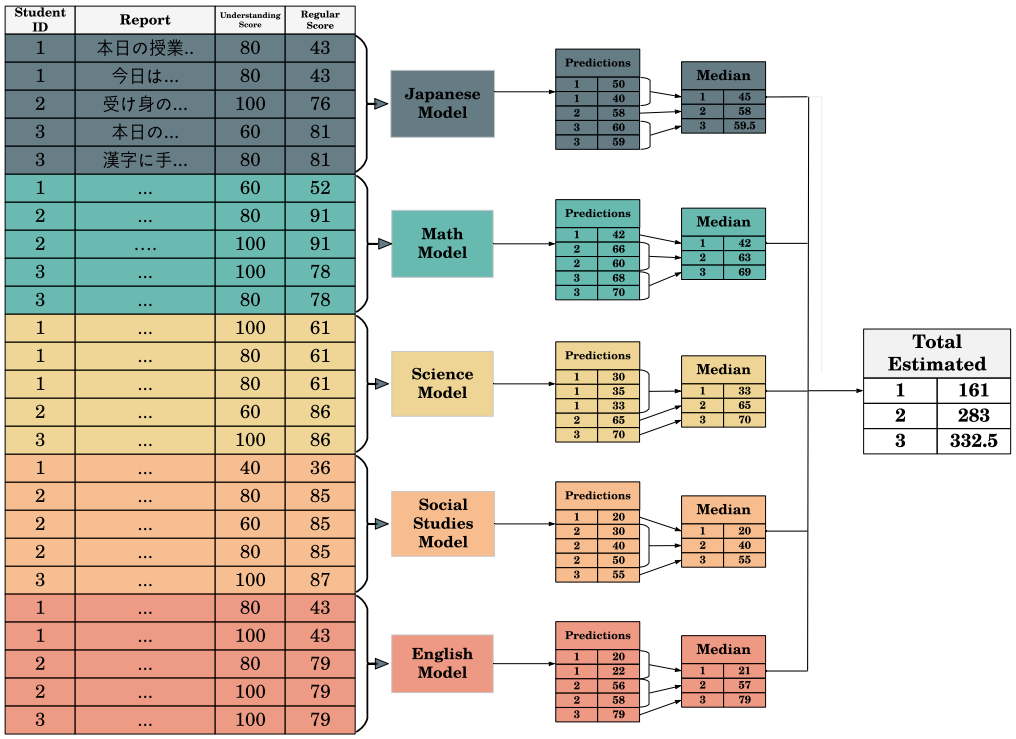


Figure 4.2: Architecture of the Reports Model.

Input Representation

In the Reports Model, teachers' comments are represented using two methods: traditional TF-IDF vectorization and BERT embeddings. The process begins with text preprocessing, starting with tokenization—splitting sentences into words. For English, tokenization is straightforward, done by splitting text at spaces. However, in Japanese, tokenization is combined with morphological analysis since sentences lack spaces. For this, we used the Fugashi parser [55], a wrapper for Mecab [56], a tool for Japanese tokenization and morphological analysis. The parser extracted key parts of speech from each report, including nouns, verbs, auxiliary verbs, adjectives, and adverbs.

The extracted terms from these parts of speech were then used to create a bag-of-words vector, weighted by TF-IDF. Since the comments are in Japanese, the parser was passed to the tokenizer parameter in sklearn, and a predefined list of Japanese stop words was used in the vectorization process.

The second approach for text representation in the model utilized BERT embeddings. The BERT model employed for this task was pretrained by the

Inui Laboratory at Tohoku University[1] using the Japanese Wikipedia corpus, with the same configuration as the original BERT model. In our experiments, the embeddings were based on the BERT [CLS] token.

4.3.2 Students Model

Architecture

In the Reports Model, each subject uses a separate regression model to evaluate a student's performance based on individual reports. However, the Students Model takes a different approach by summarizing a student's overall performance in a single input vector. This model acknowledges that a student's performance can vary between classes due to different factors, so it combines all of the student's reports into one comprehensive vector rather than treating each report individually.

The elements of each vector in the Students Model depend on the selected feature set. To capture the overall meaning of the reports, the Students Model takes a more statistical approach than the Reports Model. For example, in feature set FS_1, the vector includes the number of classes the student attended, the first, second, and third quartiles of their understanding score, and the teachers' comments. Figure 4.3 shows the vector elements when using FS_3 in the model.

Input Representation

In the Students Model, teachers' comments are represented using the traditional BoW method rather than TF-IDF, similar to the approach used in the Reports Model. BoW is also compared with BERT embeddings. To create the BoW vector, nouns and verbs were extracted using the Mecab [56] tokenizer and morphological analysis tool. These extracted words were then used to build a vocabulary corpus for the BoW vector, where each element represents a word's frequency in the sentence. Sklearn's `CountVectorizer` [6] was used to construct the BoW vector, which contained 6033 words.

For BERT embeddings, the BERT [CLS] token embeddings were employed as in the Reports Model, but with two variations in the Students Model: $BERT_1$

[1] `https://github.com/cl-tohoku/bert-japanese`

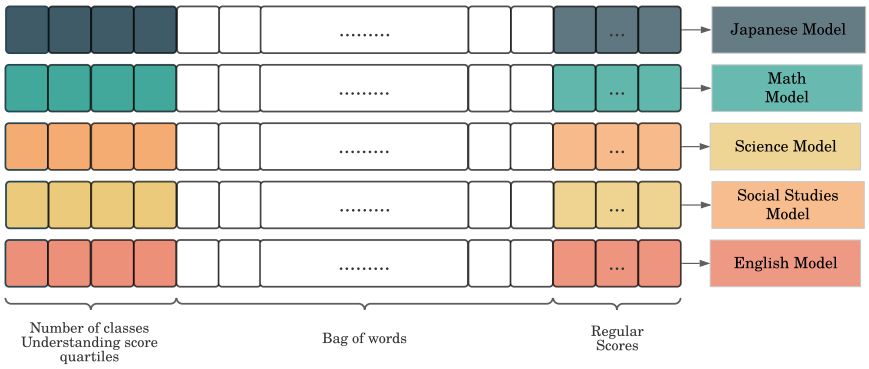


Figure 4.3: FS_3 vectors in the Students Model.

and $BERT_2$. In $BERT_1$, each class's comments were encoded individually and summed to be combined with other features. In $BERT_2$, all comments from each class were concatenated and then encoded as a single BERT representation. A comparison of the Reports Model and Students Model is presented in Table 4.3.

Table 4.3: Comparison of the Reports Model and Students Model.

Aspect	Reports Model	Students Model
Input	Individual report per lesson	Combined reports for each student
Representation	TF-IDF, BERT embeddings	Bag-of-Words, BERT embeddings
Granularity	Lesson-level predictions	Student-level predictions
FS_1	Understanding score, attitude score, teacher's comments	Quartiles of understanding scores, number of attended classes, all comments combined
FS_2	Gender and regular score	Regular score only
Purpose	Predict performance for each lesson separately	Capture overall performance from all reports

4.4 Results

4.4.1 Reports Model

We employed the 'GradientBoostingRegressor' from sklearn to predict student scores across multiple subjects using a group 10-fold cross-validation approach. This setup maximized data usage by ensuring each of the 159 student reports was used for testing exactly once, with reports of 143 students used in each training fold and 16 in the test fold.

Initially, we developed a 'Direct' model, predicting total scores without considering subject-specific information due to missing subject codes in some observations. Once subject codes became available, we built the 'Reports' model, which consists of separate regression models tailored for each subject based on subject-specific reports.

The Direct model was evaluated using the three feature sets and established as a baseline. Teachers' comments were represented using BERT embeddings. Table 4.4 summarizes the average MAE of the total score predictions for both models. As seen in the table, the Reports Model consistently outperformed the Direct model. This can be attributed to the ability of the Reports Model to tailor predictions to the characteristics of each subject. Among the feature sets, FS_3 yielded the lowest MAE across all models and subjects. Specifically, FS_3 improved MAE by 5.62 points compared to FS_2. This performance improvement suggests that teacher observations provide a more holistic view of the student's learning profile.

Table 4.5 further breaks down the results of the Reports Model showing the MAE, PTA_0 and PTA_1 of each subject's score prediction model using both TF-IDF and BERT embeddings. The use of BERT embeddings as a feature representation technique demonstrated superior performance compared to TF-IDF across the feature sets. The BERT-based models consistently achieved lower MAE and higher PTA scores, which confirms the effectiveness of contextualized word embeddings over traditional term-frequency based methods like TF-IDF. Figures 4.4 and 4.5 further illustrate the metrics, showing that FS_3 consistently achieved the best results, with the lowest MAE and

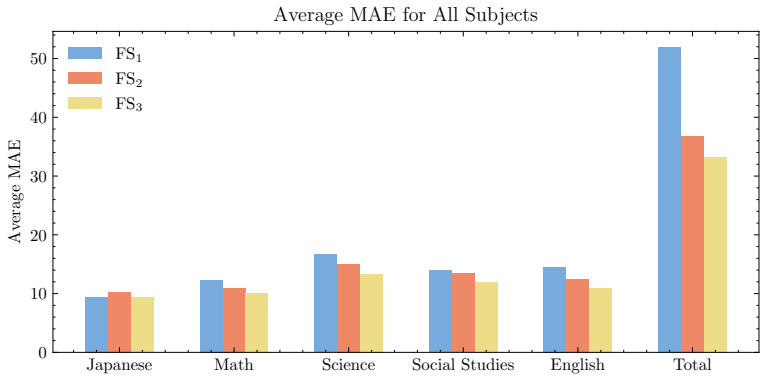


Figure 4.4: Average MAE of subject scores across all feature sets.

highest PTA scores. These results highglight the added predictive value of incorporating teachers' qualitative reports into grade predictions.

Table 4.4: Average MAE of total score prediction with the Direct and Reports Models across feature sets.

	FS_2	**FS_1**	**FS_3**
Direct	42.73	53.81	38.91
Reports	**36.83**	**52.02**	**33.29**

Table 4.5: Evaluation metrics for subject score predictions using TF-IDF and BERT with the Reports Model.

		Japanese			**Math**			**Science**		
		MAE	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**
TFIDF	FS_2	10.32	0.37	0.20	10.96	**0.53**	0.12	15.02	**0.49**	0.12
	FS_1	9.79	0.36	0.22	12.53	0.47	0.10	17.25	0.37	0.09
	FS_3	**9.16**	**0.38**	0.20	10.37	0.50	0.16	14.07	0.44	0.16
BERT	FS_1	9.47	0.27	**0.23**	12.36	0.45	0.07	16.66	0.40	0.11
	FS_3	9.32	0.37	0.22	**10.12**	0.52	**0.18**	**13.31**	0.43	**0.18**
		Social Studies			**English**			**Total**		
		MAE	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**
TFIDF	FS_2	13.43	0.51	0.087	12.48	0.58	0.12	36.83	0.58	0.15
	FS_1	13.57	**0.60**	0.01	14.93	0.52	0.00	54.81	0.47	0.07
	FS_3	12.08	0.56	0.086	12.10	0.58	**0.13**	35.19	0.621	0.14
BERT	FS_1	13.92	0.55	0.02	14.51	0.52	0.02	52.02	0.49	0.07
	FS_3	**12.00**	0.53	**0.095**	**10.99**	**0.62**	0.11	**33.29**	**0.622**	**0.17**

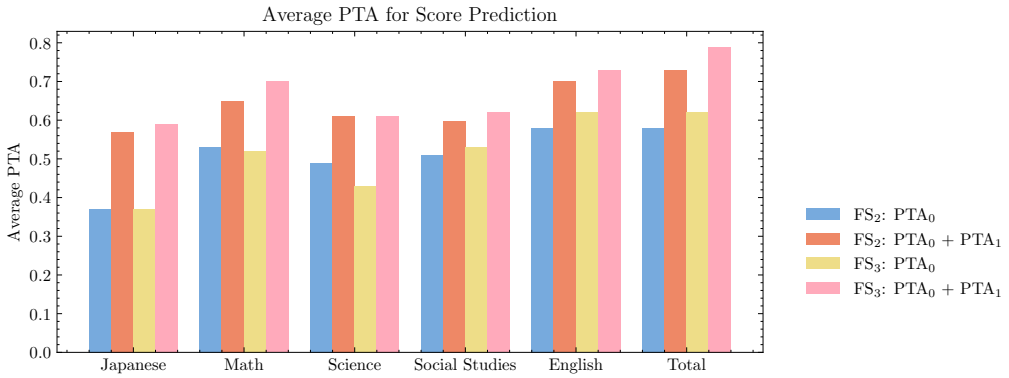


Figure 4.5: PTA metrics for FS_2 and FS_3 across subjects.

4.4.2 Students Model

The Students Model was evaluated using all three feature sets detailed in Chapter 3. Table 4.6 summarizes the results for the model, with bold values indicating the highest-performing metrics for each subject. Notably, FS_3 consistently outperformed the other feature sets in terms of both MAE and PTA scores. Figures 4.6 and 4.7 further illustrate the comparative results, showing that FS_1 and FS_3 generally achieved lower MAE and higher PTA_0 values across subjects. This pattern suggests that models incorporating teachers' qualitative insights into student performance can provide more accurate predictions than those relying solely on quantitative student scores.

Table 4.6: Evaluation metrics for subject score predictions using the 3 feature sets with BoW and BERT in the Students Model.

		Japanese			**Math**			**Science**		
		MAE	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**
BoW	FS_2	11.87	0.48	0.45	18.18	0.34	0.47	23.53	0.21	0.47
	FS_1	10.62	**0.54**	0.42	12.42	0.47	0.46	17.32	0.34	0.48
	FS_3	**9.42**	0.55	**0.43**	**10.32**	**0.49**	**0.48**	**12.15**	**0.47**	**0.48**
		Social Studies			**English**			**Total**		
		MAE	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**
BERT	FS_2	18.62	0.31	0.48	19.28	0.29	0.49	76.67	0.36	0.47
	FS_1	14.35	0.36	0.53	12.65	0.42	**0.50**	48.02	0.59	0.39
	FS_3	**10.92**	**0.54**	0.42	**10.10**	**0.53**	0.42	**31.67**	**0.68**	**0.32**

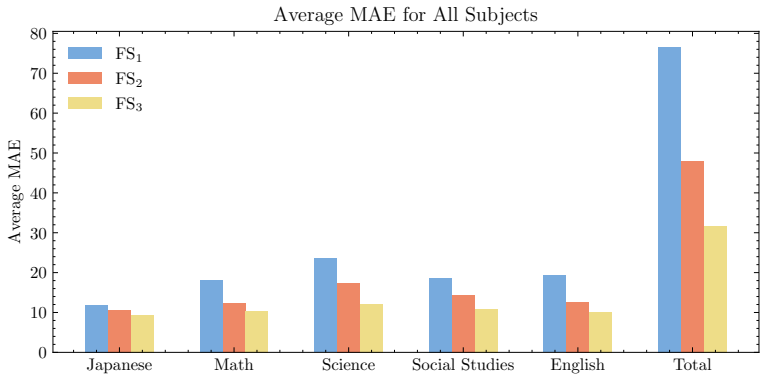


Figure 4.6: Average MAE for all subjects using the 3 feature sets.

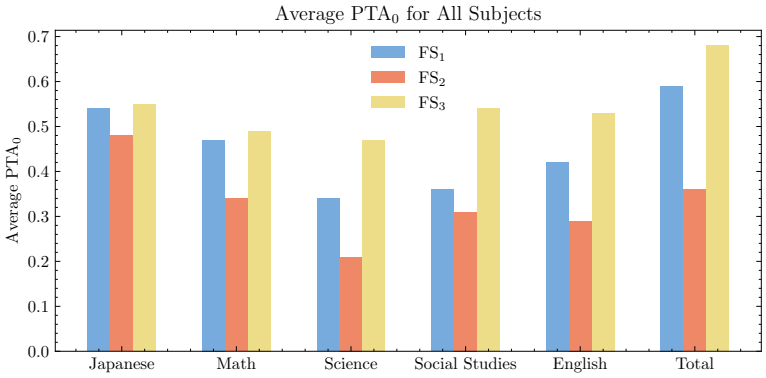


Figure 4.7: Average PTA_0 for all subjects using the 3 feature sets.

In addition, experiments using BERT embeddings in the Students Model were conducted to assess potential improvements in predictive performance similar to the Reports Model. Table 4.7 compares the results of FS_3 when using BoW representations versus BERT embeddings. BERT embeddings were utilized in two main ways: $BERT_1$ and $BERT_2$. In $BERT_1$, the teachers' comments for each class were encoded and then summed up to be combined with the rest of the features. In a different manner, in $BERT_2$, all comments from each class were first concatenated and then represented using BERT embeddings. Interestingly however, the BoW representation consistently yielded lower MAE scores than BERT embeddings, suggesting that while BERT captures nuanced language patterns, BoW representations may better align with this aggregated approach.

Table 4.7: MAE comparison of Students Model using BERT and BoW embeddings with FS_3.

		Japanese	Math	Science	Social Studies	English	Total
MAE	$BERT_1$	9.52	10.93	12.75	11.13	11.07	34.30
	$BERT_2$	9.47	10.73	12.83	11.62	10.87	33.65
	BoW	**9.42**	**10.32**	**12.15**	**10.92**	**10.10**	**31.67**

4.5 Discussion

Table 4.8 presents a summary of the Reports Model results, focusing on BERT embeddings, as they outperformed the TF-IDF representation. The top metric values in each subject are highlighted in bold, and PTA_1 is highlighted by $PTA_0 + PTA_1$ accuracy.

The Students Model, when tested with FS_1 and FS_3, achieved lower MAE and higher PTA_0 and PTA_1 scores across all subjects compared to the Reports Model, as shown in Table 4.8. This emphasizes that student performance prediction is better achieved through a comprehensive model that consolidates information across all reports rather than treating them as isolated observations.

It is also interesting to highlight the PTA_1 results achieved by the Students Model where it significantly outperforms the Reports Model in both feature sets. Hence, it can be suggested that the Students Model is a more reliable one since it mostly either correctly identifies the letter grade or misses it by one tick. In the Students Model, $PTA_0 + PTA_1$ reaches an accuracy of 0.98 with FS_1. The Reports Model achieved improved results with FS_2, due to its instance-based approach, which treats individual reports as separate data points. This setup inadvertently acts like oversampling, potentially explaining the performance gains when using the second feature set.

Table 4.8: Performance comparison between the Students Model and Reports Model across the different subjects using FS_1 and FS_3.

		Reports Model			Students Model		
		MAE	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**
FS_1	**Japanese**	9.47	0.27	0.23	10.62	0.54	0.42
	Math	12.36	0.45	0.07	12.42	0.47	0.46
	Science	16.66	0.40	0.11	17.32	0.34	0.48
	Social Studies	13.92	**0.55**	0.02	14.35	0.36	0.53
	English	14.51	0.52	0.02	12.65	0.42	0.50
	Total	52.02	0.49	0.07	48.02	0.59	0.39
FS_3	**Japanese**	**9.32**	0.37	0.22	9.42	**0.55**	**0.43**
	Math	**10.12**	**0.52**	0.14	10.32	0.49	**0.48**
	Science	13.31	0.43	0.18	**12.15**	**0.47**	**0.48**
	Social Studies	12.00	0.53	0.10	**10.92**	0.54	**0.42**
	English	10.99	**0.62**	0.11	**10.10**	0.53	**0.42**
	Total	33.29	0.62	0.17	**31.67**	**0.68**	**0.32**

The findings from this study demonstrate that both the Reports Model and the Students Model offer valuable frameworks for predicting student performance using teacher reports and student data. The feature set FS_3 led to marked improvements in prediction accuracy, showcasing its utility across models. Compared to traditional scoring metrics represented by FS_2, a 32% increase of accuracy was achieved by integrating teacher reports using the Students Model. In the Reports Model, teacher comments represented with BERT embeddings showed a significant advantage over TF-IDF, emphasizing the role of contextual understanding when representing natural language data.

The Students Model, on the other hand, effectively condensed individual performance data by merging reports into a single vector. This holistic representation proved valuable in capturing broad trends and subject-level performance variations. Lower MAE values in the Students Model validate that an aggregated approach to student data strengthens prediction capabilities. The Students Model achieved a 68% PTA_0 in predicting total letter grades—a notable improvement over the Reports Model—further supporting the effectiveness of a unified data representation in accurately assessing student performance.

4.6 Summary

In conclusion, this study demonstrated the effectiveness of leveraging teacher reports and student data through two distinct models—the Reports Model and the Students Model—to enhance the accuracy of student performance predictions. The Reports Model treated each report as an individual instance capable of capturing unique insights, while the Students Model consolidated reports to provide a comprehensive view of student performance. The Reports Model showcased the benefits of using BERT embeddings over TF-IDF representations, emphasizing the importance of contextual understanding in natural language processing tasks. Both models demonstrated the value of FS_3 in improving predictive accuracy. However, the Students Model achieved a notable 68% accuracy in predicting letter grades, outperforming the Reports Model by 6%. Moving forward, further refinements and the integration of additional data sources will be essential to advance these predictive capabilities, ultimately contributing to more tailored educational insights and improved prediction.

5

TOPP: Linguistic, Psychological, and Sentiment (LPS) Features Model

5.1 Introduction

In this TOPP study, we aim to gain a deeper understanding of how teacher mindsets influence student success by presenting an innovative analytical framework. Instead of relying solely on traditional techniques to numerically analyze teacher reports, our approach explores the emotional and psychological dimensions found within these observations. This method allows us to move beyond surface-level data and capture both the content and the emotional tone of teacher feedback. By integrating sentiment analysis and topic modeling, our framework provides a nuanced perspective on how teachers' views and communication styles may relate to students' academic performance.

Teacher assessments are fundamental to education, serving not only as indicators of student learning but also reflecting educators' underlying mindsets and beliefs. Research has demonstrated that teachers' attitudes towards intelligence—whether they view it as fixed or malleable—have a significant impact on student outcomes. A teacher with a **fixed mindset** may perceive intelligence as an unchangeable characteristic, believing that there is little that can be done to change it [57]. Research in self-theories has shown that such beliefs can limit the support teachers offer to students and inadvertently hinder their progress and motivation [57] [58]. In contrast, teachers with a **growth mindset** foster a belief in students that their abilities can improve

through effort, correlating with positive educational results. Given this correlation between teachers' mindsets and students' outcomes, we explored the potential of using daily teacher observation reports to enhance performance prediction models. Specifically, we extracted topics, sentiment scores, and linguistic and psychological features from the reports to construct an interpretable input vector.

Our study is guided by two key research questions:

- **RQ1:** Can we create an interpretable input vector that synthesizes teachers' reports to predict student performance?
- **RQ2:** Can early observations predict future performance?

To address these questions, we conducted a series of experiments on different subsets of the data. In **EXP1**, we tested the model's ability to predict student performance based on teacher reports recorded four months prior to the performance outcome, aiming to evaluate the model's effectiveness with early data. In **EXP2** and **EXP3**, we tested the model's consistency and predictive accuracy over two different time frames. Our methodology combines supervised topic modeling with advanced sentiment analysis, enriched by linguistic and psychological feature extraction to yield a comprehensive and interpretable predictive model.

5.2 Related Work

In Chapter 3, we outlined the various features used in performance prediction models, including performance measures and employed methods. This chapter, along with the next, centers on the application of sentiment analysis (SA) in predicting performance outcomes.

Educational data mining has shown a rising interest in SA, as numerous studies have enhanced SA frameworks to provide visual feedback [59, 60], recommend learning interventions [61, 62], and more [63, 64]. SA is now commonly applied to understand learners' satisfaction levels, identify areas of concern, and improve instructional methods [65, 66].

Given the research on the significant relationship between students' emotional states and their academic achievements [67], SA has also become a critical instrument for predicting academic outcomes. Studies have used SA on student feedback, self-evaluations, and online discussions to predict academic performance. For example, Yu et al. developed a model to predict academic failure early by applying SA to students' self-evaluated comments [68]. Similarly, Crossley et al. combined SA with click-stream data in MOOCs to predict course completion [69]. Robinson, Navea, and Ickes extended sentiment analysis to assess the content of students' self-introductions, using SA to predict course performance based on the tone and sentiment of introductory statements. This analysis, which employed the Linguistic Inquiry and Word Count (LIWC) tool, illustrated how initial sentiment indicators correlate with final course grades [70]. Wen et al. also used SA on a forum posts in a massive open online classes (MOOC) to predict student performance, to identify trends in students' opinions and overall course success [71]. In another study, Wen et al. reported a significant correlation between sentiment scores and the number of students dropping from MOOc, demonstrating that sentiment analysis can be a valuable tool for predicting academic outcomes [72].

In conclusion, previous research shows that analyzing student feedback and discussions with sentiment analysis can help predict academic performance and identify students who may need support. Most studies have focused on students' own words to understand their emotions and motivation. However, to the best of our knowledge, there is no research on using teacher reports to predict student outcomes using sentiment analysis. Examining teachers' observations and sentiments could provide a new perspective on how teacher attitudes influence student success. This could help create a more complete view of the classroom environment and improve models for predicting student performance.

5.3 Experimental Settings

While the teacher reports are given in the five distinct core subjects: Japanese, Mathematics, Science, Social Studies, and English, the goal of this study was

Table 5.1: Overview of dataset information for each experiment, detailing the number of students and reports for regular scores and their availability.

	Date Range of Reports	Simulation Exam	Complete Dataset		Regular Score		Regular Score Provided	
			Students	Reports	Grade	Year	Students	Reports
EXP1	5/2020 - 5/2021	9/2021	321	24,477	8	2020	175	18,566
EXP2	5/2020 - 9/2020	9/2020	441	19,653	9	2020	377	17,545
EXP3	5/2021 - 9/2021	9/2021	433	19,866	9	2021	279	13,988

to develop a general and straightforward model for analyzing teacher reports. Thus, we chose to predict each student's total score without distinguishing between specific subjects.

Table 5.1 shows the number of students with regular test scores and the corresponding number of reports provided in each of the three experiments. The total number of reports per student varied based on the number of classes attended.

5.3.1 Feature Selection

In our experimental settings, we adopted the three feature sets for comparison. The first feature set, FS_1, consists of the input vectors derived from the teachers' comments. The method of extraction is explained in Section 5.4. The second feature set, FS_2, contains the students' regular test scores. In **EXP1** and **EXP2**, these regular test scores correspond to tests taken in 2020, while in **EXP3**, they correspond to tests taken in 2021. Finally, the third feature set, FS_3, is a combination of FS_1 and FS_2, allowing us to assess whether incorporating teachers' reports improves the prediction accuracy of the model.

5.4 Methodology

In our earlier research, [32], bag-of-words reprsentations and BERT embeddings of the teachers' comments were used as input vectors to predict student performance. In this study, we adopt a sentiment analysis approach, utilizing a pre-trained BERT model specialized in sentiment analysis [73] to

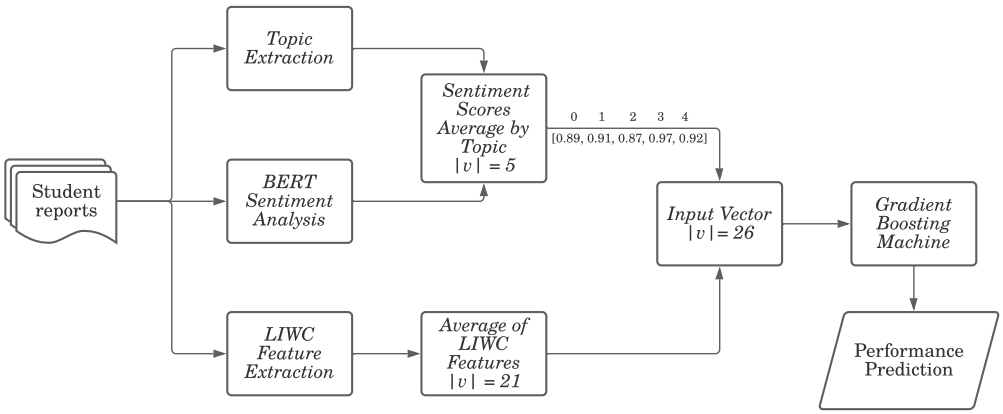


Figure 5.1: Overview of the methodology for sentiment analysis and linguistic feature extraction.

assign sentiment scores to various discussion topics. Additionally, we extract psychological and linguistic features using LIWC. The details of each step are thoroughly explained in the following subsections. Figure 5.1 illustrates the overall methodology employed in this analysis.

5.4.1 Seed-guided Topic Extraction

Teachers' reports often address multiple aspects of student development, such as in-class performance or effort in completing homework. To extract the topics discussed in these reports, we employed a seed-guided approach. We began by pre-defining five key topics that were deemed central to the teachers' feedback: (1) Understanding, (2) Classwork, (3) Tests, (4) Concentration, and (5) Homework. These topics were selected based on preliminary analysis of the report data, considering both their frequency of occurrence and pedagogical importance.

To identify the most frequently mentioned key phrases in teachers' comments, we employed the TopicRank algorithm from the pke extraction toolkit [74]. After extracting the key phrases, we assigned them to the predefined topics by calculating the cosine similarity between the BERT embeddings of the key phrases and the topic name embeddings. Cosine similarity was chosen due to its simplicity and widespread usage in seed-guided topic mining approaches [75, 76], though we acknowledge its limitations in capturing fine-grained

Table 5.2: Topics and their corresponding key phrases.

Topic	Key Phrases (Japanese)	Key Phrases (English)
Understanding	理解, 覚える, 分かる, やり直し, 使い方, 間違い, 難しかっ, 解ける, 問題なく, スムーズ, 書き換え, 解く, スピード, 見直し, 使い分け, 間違え, 正しく, 苦手	comprehension, to remember, to understand, redoing, usage, mistake, difficult, to be solved, no problem, smooth, rewriting, to work out, speed, review, proper use, mistake, correct, poor at
Classwork	授業, 本日, 学校進度, 説明, 内容, 学習, 演習, 範囲, 学力, 講習, 意識, 予習	class, today, school progress, explanation, contents, learning, drill, scope, academic, training, awareness, preparation for a lesson
Tests	試験, テスト, 単語テスト, 定期テスト, 学校テスト, 確認テスト, 考査, 計算ミス	exam, test, vocabulary test, regular test, school test, confirmation test, examination, calculation error
Concentration	集中, 途切れ, 詳しく	concentration, interruption, detail
Homework	ケアレスミス, 練習問題, 予想問題, 文章問題, 難しい内容, 宿題	careless mistakes, practice problems, anticipation problems, composition problems, difficult contents, homework

semantic distinctions. The identified topics and their corresponding key phrases are presented in Table 5.2.

Given that a single report may address multiple topics, each report was segmented into sentences as a pre-processing step. The sentences and topic phrases were then converted into BERT embeddings, and each sentence was assigned to the topic with the highest cosine similarity. We found that the topics 'Understanding' and 'Classwork' accounted for the majority of sentences (47.3 and 47%, respectively), while 'Homework,' 'Tests,' and 'Concentration' were less frequent, with 'Homework' averaging 4%. This distribution sug-

gests that teachers predominantly focus on students' understanding and daily classroom progress in their feedback reports.

5.4.2 BERT Sentiment Analysis

We implemented a contextualized sentiment analysis framework using a Japanese-specific BERT model fine-tuned on sentiment analysis [73]. This model produces both polarity scores and confidence metrics, which we normalized to a [-1, 1] scale.

The distribution of sentiment scores exhibited a pronounced negative skew, indicating that most of the scores were concentrated on the higher end of the scale, which corresponds to positive sentiment values. In a negatively skewed distribution, the tail on the left side is longer than the right side. This means that while there are some lower sentiment scores, the majority of scores cluster towards the right, reflecting a predominance of positive sentiments expressed in the teacher comments. Specifically, 96% of all sentences received positive sentiment scores, which suggests that teachers generally provide encouraging feedback.

Given that our input vector is organized by individual students, we computed the average sentiment for each topic by aggregating the sentiment scores associated with each student's comments. Considering the skewness in the data distribution, we chose to utilize the median as a more robust measure of central tendency. Our analysis indicated that the topics of 'Test' and 'Classwork' had the highest median sentiment scores, at 0.95 and 0.93, respectively, while the 'Understanding' and 'Homework' topics reflected lower averages of 0.87 and 0.82.

Figure 5.2 illustrates the distribution of sentiment scores across the five topics. All topics demonstrated significant negative skewness, with 'Classwork' showing the highest skew and 'Understanding' the least. This suggests that teachers adopt a more supportive tone when discussing 'Classwork,' while occasionally expressing concerns regarding students' grasp of the material in 'Understanding.'

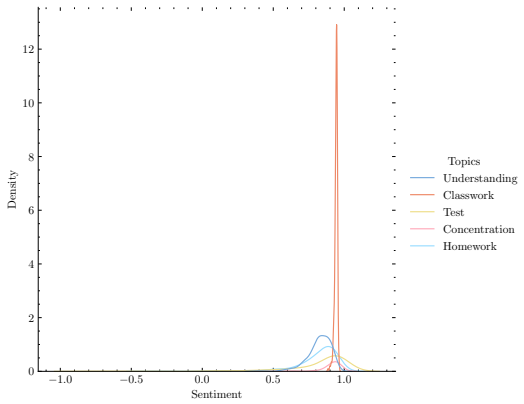


Figure 5.2: Sentiment score distribution in each of the predefined topics

Furthermore, we conducted a bivariate correlation analysis between the average sentiment scores for each topic and the final simulation scores. Among the three analyzed data subsets, the average sentiment scores for the 'Understanding' topic consistently demonstrated the strongest correlation with simulation scores, achieving a maximum R value of 0.35. This finding highlights the potential influence of teacher sentiment on student performance, particularly in areas related to understanding the material.

5.4.3 Linguistic and Psychological Analysis

Our analysis revealed that teachers' comments in the dataset are predominantly encouraging and supportive, resulting in a significant negative skew in the sentiment score distribution. To further explore the linguistic and psychological characteristics of these comments, we employed the LIWC software, which quantifies the percentage of words used across more than 70 linguistic and psychological categories. This analysis is particularly relevant as it provides insights into the underlying emotional and cognitive processes reflected in the teachers' comments.

For our study, we utilized the Japanese version of LIWC [77], which not only translates the original LIWC English dictionary but also includes words unique to the Japanese language, having undergone validation by psychologists. By applying this specialized dictionary, we were able to analyze each teacher's report, computing the average for each output category per student.

Next, we performed a bivariate correlation analysis, examining the LIWC categories alongside the average sentiment scores for each predefined topic in relation to the students' final scores. From an initial pool of 89 variables, we identified 21 that exhibited significant correlations with the final scores across the three experiments, using a threshold of $p < 0.01$ for selection. The key variables identified were 'Anxiety', 'Power', 'Affective or Emotional Processes', 'Risk', and 'Common Verbs'. Table 5.3 summarizes the correlations with students' final scores.

Next, a bivariate correlation analysis was conducted on each of the LIWC categories along with the average sentiment score in each of the pre-defined topics with the students' final scores. From a total of 89 variables, we selected 21 which had a significant correlation with the final score from the 3 experiments. Table 5.3 shows the 5 highest correlated variables with p-values < 0.01.

Table 5.3: Correlation of selected linguistic features with final simulation scores.

Variable	R
Anxiety	0.29
Power	-0.23
Affective or emotional processes	0.17
Risk	0.17
Common Verbs	-0.15

The 21 variables were "pronoun", "ipron", "quant", "verb", "adjverb", "affect", "posemo", "negemo", "anx", "health", "power", "risk", "tentat", "percept", "see", "bio", "drives", "reward", "Period", "AllPunc", and "Quote". Consequently, the final input vector for FS_1 incorporated the average counts of these selected variables along with the average sentiment scores for the five predefined topics. After constructing the input vectors for the three feature sets, we employed a Gradient Boosting Machine as our predictive model for the students' scores. Figure 5.1 presents a flowchart that illustrates the methodology employed in this analysis.

5.5 Experiments

5.5.1 Results

We compared our input construction approach with the BoW baseline of the Students Model explained in 4 by constructing similar input vectors. For FS_1, a BoW vector was created to represent all teachers' comments for each student. FS_2 consisted of the regular score, while FS_3 was a concatenation of FS_1 and FS_2. The average MAE across all ten folds was computed and served as the primary evaluation metric, alongside (PTA_0) and PTA_1. The results of the experiments are shown in Table 5.4. Values in bold indicate best MAE and PTA_0 results in both approaches and values in blue indicate best overall results.

Table 5.4: Evaluation metrics for score predictions using the three feature sets in the three experiments using our feature vector compared to the BoW baseline.

		Ours			BoW		
		MAE	**PTA_0**	**PTA_1**	**MAE**	**PTA_0**	**PTA_1**
FS_1	**EXP1**	66.29	0.36	0.45	39.68	**0.71**	0.29
	EXP2	56.65	0.56	0.30	65.5	0.45	0.36
	EXP3	65.82	0.40	0.40	70.57	0.27	0.50
FS_2	**EXP1**	42.39	0.55	0.37	42.39	0.55	0.37
	EXP2	38.61	0.63	0.29	38.61	0.63	0.29
	EXP3	**36.22**	**0.60**	0.35	36.22	**0.60**	0.35
FS_3	**EXP1**	**40.07**	**0.61**	0.33	**39.07**	0.61	0.31
	EXP2	**36.03**	**0.68**	0.26	**36.14**	**0.67**	0.27
	EXP3	37.97	0.59	0.35	**35.28**	0.59	0.35

5.5.2 Discussion

To address **RQ1**, we extracted sentiment and linguistic features from teacher comments and conducted **EXP2** and **EXP3** to evaluate the effectiveness of the feature sets. The results showed that incorporating these features improved the accuracy of performance predictions. Specifically, the addition of FS_1 (sentiment and linguistic features) to FS_2 (regular score) to create (FS_3) led

to an increase of 5% in PTA_0 in EXP2 with our feature vector compared to an increase of 4% in the BoW vector. In EXP3, the change was -1% for both feature vectors. In EXP1, both approaches showed an increase of 6% in PTA_0 when using FS_3 compared to FS_2. These results suggest that the sentiment and linguistic features enhance the predictive model, although the improvement is modest. FS_2, which includes the regular score, showed consistently lower MAE and outperformed FS_1 in all experiments using our approach due to the regular score's strong correlation with the final simulation score.

In addressing **RQ2** on the potential for early performance prediction, we conducted EXP1 using a subset of the reports ending five months before the exam. The MAE for EXP2 and EXP3 with all feature sets was lower than for EXP1, as expected due to the greater time interval between the reports and the exam. The gap in time likely allows for significant changes in student performance, creating a trade-off between early prediction and accuracy. In EXP1, BoW yielded significantly higher MAE and PTA_0 scores than both FS_2 and FS_3, suggesting that BoW might be more effective for early predictions. However, in EXP2 and EXP3, sentiment features in FS_1 were able to compete with BoW, indicating that these psychological and emotional features may gain relevance as the exam approaches.

In conclusion, extracting sentiment characteristics and topics from the teachers' comments provides for more in-depth comprehension and analysis of the reports while also giving a competitive capacity for performance prediction. The features we extracted from the teachers' comments reduce both size complexity compared to the BoW vector and time complexity compared to obtaining BERT embeddings. Despite the promising results, certain limitations should be acknowledged. For example, the model's performance might vary with a more extensive dataset and experiments were not held for each separate subject. Moreover, a comparison with other feature selection methods was not provided. These are suggested avenues for future research.

5.6 Summary

In this study, we explored the extraction of sentiments and psychological features from teacher observation reports by performing seed-guided topic modeling, sentiment analysis using a pretrained BERT model and LIWC feature extraction to inform a performance prediction model. We conducted experiments on three subsets of the teacher reports to assess the model's effectiveness. The extracted observational features provided valuable insights into the teachers' comments and demonstrated a significant correlation with final student performance. Our experimental results indicate that this approach can compete effectively with existing approaches. However, given the inherent limitations of teacher comments in capturing the full scope of student performance, we recommend augmenting these features with additional student-related data, such as prior academic scores. This combination could lead to a more accurate and robust performance prediction model.

6 TOPP: Sentiment-based Similarity Learning (SBSL) Model

6.1 Introduction

In this chapter, we present the final TOPP model building upon the work presented in Chapter 5 in which sentiment, topics, psychological and linguistic features were extracted to build an input vector. While these various features can give an in-depth understanding of the reports, it requires extensive data pre-processing and configuration. Topic modeling is also an unsupervised problem in nature which can lead to ambiguity in results. Therefore, we investigate the role of sentiment in teacher reports and how it can help to improve prediction. In this work, we fine-tune the siamese Sentence-BERT [14] network on the similarity of the reports according to sentiment. To the best of our knowledge, no research has been done utilizing teacher reports and enhancing the embeddings using similarity learning. Our research aims to answer two main questions.

- **RQ1**: Is there a correlation between sentiment in teacher reports and student performance?
- **RQ2**: Can similarity learning utilizing sentiment scores enhance teacher report embeddings for a more accurate performance prediction model?

We address these questions through our experiments and verify the results with 5 datasets covering reports on different subjects. We compare our results with previous research that has utilized teacher reports for performance prediction and show how our method can contribute to the field.

6.2 Experimental Settings

For the experiments, the teacher reports were divided into two main subsets, a fine-tuning dataset for similarity learning and a testing dataset. The number of reports, the date ranges and the number of students are shown in Table 6.1. The reports written in 2021 were used for pretraining and the reports from 2020 were used to make predictions about the scores on the September 2020 simulation exam. Since the reports used from 2020 belong to different students than the reports from 2021, we used the 2020 dataset for testing as in previous studies. The distribution of the September 2020 exam scores per subject are shown in Chapter 4 in Figure 4.1 in addition to the standard deviations in Table 4.2.

Table 6.1: Description of the experimental data subsets

	Finetuning Dataset	**Testing Dataset**
Reports	26,931	14,313
Range	March 2021 - August 2021	May 2020 - October 2020
Students	433	180

6.2.1 Feature Selection

In line with previous research, our experimental setup employs three main feature sets. The first set, FS_1, comprises the input vector derived from the teachers' comments, the method of which is outlined in Section 6.3. We add to the report embedding the students' average sentiment score and average understanding score. The quartiles of both scores are also appended to the vector. FS_2 is the second feature set, which comprises the students' regular scores. Lastly, we investigate the impact of combining the input vector from the teachers' comments with the regular scores on the accuracy of the prediction model. As such, FS_3 is a combination of FS_1 and FS_2.

6.3 Methodology

6.3.1 Preprocessing

Teachers at the cram school are required to write post-class reports for students to document progress and provide encouragement. However, this reporting process can be burdensome, leading to repetitive language, which can dilute the personal feedback and affect the accuracy of sentiment analysis. Upon analyzing these reports, we observed a significant frequency of repetitive sentences, which were often positive in sentiment and could skew the overall sentiment score of a report.

To address this, we segmented the reports into individual sentences and calculated the frequency of each across the corpus. The threshold for sentence removal was determined experimentally to enhance the correlation between the extracted sentiment scores and students' final scores. Sentences repeated more than 300 times were removed, balancing the need to retain valuable content while minimizing redundancy. Table 6.2 lists these most frequently removed sentences and their frequency by subject in the 2020 dataset.

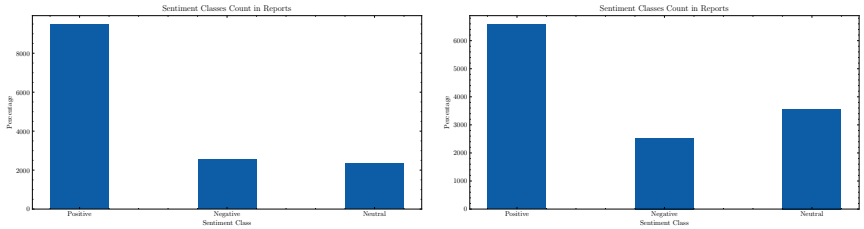
6.3.2 Sentiment Score Extraction

In the subsequent sentiment score extraction process, we used a pre-trained Twitter-XLM-RoBERTa model [78] for sentiment analysis. Each sentence was assigned a positive, neutral, or negative sentiment score along with a confidence level. To assign a sentiment score to each report, we applied a prioritization rule (NPN): if any sentence was labeled 'Negative', the entire report was marked 'Negative'; if no 'Negative' labels were present but a 'Positive' label appeared, the report was marked 'Positive'; otherwise, it was assigned a 'Neutral' label.

Figure 6.1 illustrates the sentiment distribution before and after eliminating redundant sentences in the 2020 reports. Many of the repeated sentences had positive sentiment, which impacted the overall report label. After removing these redundant sentences, the proportion of positive reports decreased from

Table 6.2: Top five repeated teacher comments across subjects with their counts.

Redundant Sentences	Japanese	Math	Science	Social	English
授業中、集中して問題に取り組んでくれました *They concentrated on the problems during the class.*	563	1247	983	858	1207
あとは宿題で復習と定着を行えば、今回の内容はばっちりです *Now all you need to do is review and consolidate the content with homework and you'll be all set for this time!*	555	1232	1016	866	1522
本日の学習内容について、理解できているようです *They seem to have a good understanding of what they learned today.*	506	1000	799	730	1259
本日の宿題を実施してくれていました *They did their homework today.*	318	758	694	626	921
前回のチェックテストについて、合格できました *Regarding the last check test, they were able to pass.*	154	250	340	312	368



(a) Before redundancy elimination. **(b)** After redundancy elimination.

Figure 6.1: Distribution of sentiment scores.

66% to 52%, while neutral reports increased from 16.2% to 28.2%. This process also reduced the number of total reports by 11.5%, indicating that 1,651 reports consisted exclusively of redundant sentences.

To address **RQ1**, we performed a bivariate correlation analysis on both the fine-tuning and testing datasets to investigate the relationship between students' ratio of positive-sentiment reports to total reports and their corresponding simulation exam scores. In the fine-tuning dataset, the sample correlation coefficient (r) was 0.2, with a p-value of less than 0.01, which allowed us to reject the null hypothesis, thereby indicating a statistically significant linear relationship between sentiment ratio and final scores.

Similarly, in the testing dataset, the average correlation coefficient across all subjects was 0.15, with a p-value below 0.05, supporting the presence of this relationship. Notably, when neutral-labeled reports were excluded and the sentiment ratio recalculated, the correlation coefficient in the testing dataset rose to 0.19. Based on these findings, neutral reports were excluded from the testing analysis.

While the correlation between sentiment ratio and final score is moderate, these results consistently demonstrate a meaningful relationship between sentiment in teachers' reports and student performance.

6.3.3 SBERT Finetuning

In previous work, various methods have been employed to generate embeddings from teachers' comments, such as using BERT embeddings based on the [CLS] token for each report or constructing a BoW model. In this study,

however, we aimed to generate more predictive embeddings by fine-tuning Sentence-BERT (SBERT) [14] using the base BERT model pretrained on Japanese text by Tohoku University [1] with a focus on similarity learning. SBERT is designed to bring sentence embeddings with similar semantic content closer together in the vector space while separating embeddings with different semantic meanings.

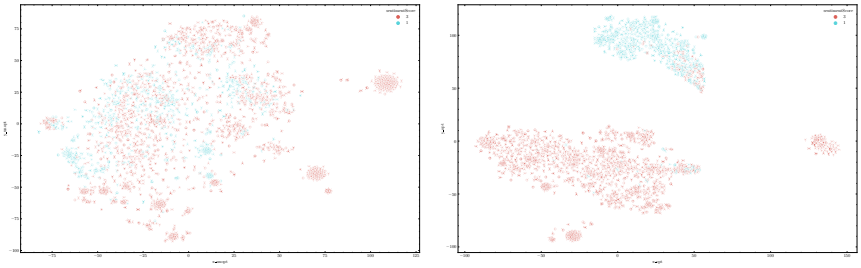
To fine-tune SBERT for sentiment similarity, we created pairs of reports (x, y), with similarity scores s_{xy} assigned based on the sentiment labels of the reports. For report pairs where both x and y had a positive sentiment label, we assigned a similarity score $s_{xy} = 1$. Conversely, for pairs where one report had a positive label and the other had a negative label, we assigned a similarity score $s_{xy} = 0$. The dataset comprised 5,659 negatively-labeled reports and 17,457 positively-labeled reports, from which we generated 34,914 positive-positive pairs and 17,457 positive-negative pairs, creating twice as many positive-positive pairs as positive-negative pairs to manage the distributional imbalance.

The siamese network was trained to minimize the mean squared error (MSE) loss between the cosine similarity of the report embeddings, $\mathbf{r}_x$ and $\mathbf{r}_y$, and the target similarity score $s_{xy} \in \{0, 1\}$, as shown in Equation 6.1.

$$\mathcal{L} = \frac{1}{n}\sum^{n}(s_{xy} - \cos(\mathbf{r}_x, \mathbf{r}_y))^2 \tag{6.1}$$

After fine-tuning SBERT, we generated optimized sentence embeddings for the testing dataset by averaging the word vectors within each report. To illustrate the sentiment clustering achieved with these optimized embeddings, we also extracted sentiment scores for the testing set. Using t-distributed Stochastic Neighbor Embedding (t-SNE), we visualized the Math report embeddings obtained from the fine-tuned model (Figure 6.2(b)) and compared them with embeddings from the original SBERT base model, 'tohoku-jp-bert-wwm,' which had not been further fine-tuned (Figure 6.2(a)). Each point in the figure represents a report, color-coded by its assigned sentiment score.

[1] `tohoku-jp-bert-wwm` from https://github.com/cl-tohoku/bert-japanese



(a) Unoptimized Embeddings **(b)** Optimized Embeddings

Figure 6.2: Math report embeddings labeled according to sentiment score

As shown in Figure 6.2, the fine-tuned model successfully clusters the report embeddings in alignment with sentiment labels, demonstrating clear separation between positive, neutral, and negative reports.

Building on the approach in Chapter 4 and 5, where a summarizing vector was created for each student, we calculated the sum of the report embeddings for each student. This summarized vector served as input for FS_1 and FS_3 in an XGBoost model, which was then used to predict the final subject score and score category.

6.3.4 Performance Prediction with Optimized Embeddings

Given the variation in report content across subjects and to ensure robust model evaluation, we trained five separate XGBoost models for regression and classification—one for each subject. Model performance in predicting final subject scores was assessed using MAE. For classification, we evaluated accuracy in predicting both letter grades and student pass/fail status, with pass/fail determined based on a 50% threshold. Letter grades were defined on a 4-class scale where: A (100-75), B (75-50), C (50-25), and D (25-0). In this current study, the switch from six to four classes for grade prediction was made to simplify the classification task and address class imbalances that often arise in educational datasets. The inclusion of pass/fail classification adds an additional practical layer for evaluating student performance to offer a straightforward indicator of success.

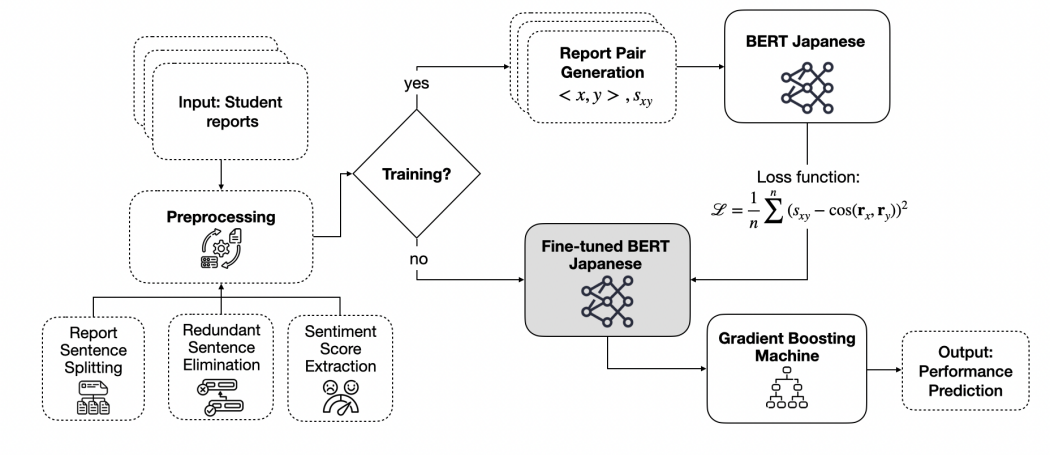


Figure 6.3: Flowchart describing proposed sentiment based similarity learning approach.

6.4 Experiments

6.4.1 Results

All experiments were evaluated using 10-fold cross-validation. For each fold, the MAE was calculated, along with the accuracy of pass/fail predictions and PTA_0 for 4-class letter grades. Given the class imbalance in the test datasets, model PTA_0 was also compared to a majority class (MC) baseline. Table 6.3 shows MAE and PTA_0 for 4-class and pass/fail (P/F) classifications with optimized and unoptimized SBERT embeddings, where bolded values indicate top performance. Overall, optimized embeddings outperformed unoptimized ones, and though FS_1 did not exceed the 4-class baseline, accuracy improved by 2% with optimized embeddings compared to a 3% drop with unoptimized embeddings. Notably, FS_3 performed better than FS_2 with optimized embeddings, highlighting the impact of our fine-tuning approach.

To benchmark against prior research, we also created vectors similar to [32] in Chapter 4, using embeddings obtained without further fine-tuning and without redundancy elimination. The average results for each subject are shown in Table 6.4. Although the "Students Model" approach achieved a lower MAE for FS_1, the Similarity Learning approach showed better performance in all other cases.

Table 6.3: Comparison of MAE, PTA_0, and pass/fail accuracy (P/F) for optimized and unoptimized embeddings across FS_1, FS_2, and FS_3 subject datasets, including a 4-class majority class baseline (MC).

		Optimized			Unoptimized			MC
		MAE	**PTA_0**	**P/F**	**MAE**	**PTA_0**	**P/F**	
FS1	Japanese	**9.55**	0.53	**0.59**	9.66	**0.55**	0.55	0.58
	Math	**12.08**	**0.46**	**0.61**	12.25	0.43	0.61	0.46
	Science	**17.32**	**0.42**	**0.59**	18.52	0.39	0.48	0.41
	Social Studies	**13.93**	0.41	**0.66**	14.13	0.43	0.59	**0.44**
	English	14.21	0.44	0.56	**13.92**	0.42	**0.62**	**0.45**
	Average	**13.42**	0.45	**0.60**	13.70	0.45	0.57	**0.47**
FS2	Japanese	9.58	0.56	0.66	9.58	0.56	0.66	0.61
	Math	9.50	0.46	0.67	9.50	0.46	0.67	0.46
	Science	12.28	0.56	0.78	12.28	0.56	0.78	0.41
	Social Studies	11.87	0.46	0.69	11.87	0.46	0.69	0.44
	English	11.44	0.56	0.76	11.44	0.56	0.76	0.42
	Average	10.94	0.52	0.71	10.93	0.52	0.71	0.47
FS3	Japanese	**9.17**	0.58	**0.62**	10.13	0.54	0.62	**0.61**
	Math	**9.82**	**0.58**	**0.70**	10.07	0.55	0.68	0.46
	Science	12.36	**0.52**	**0.78**	**11.65**	0.48	0.74	0.41
	Social Studies	11.28	**0.50**	**0.74**	**11.21**	0.43	0.64	0.44
	English	11.61	**0.50**	0.78	**11.01**	0.47	**0.79**	0.42
	Average	10.85	**0.54**	**0.73**	**10.81**	0.49	0.69	0.47

Table 6.4: Similarity learning method results compared with previous 'Students model' approach.

		Similarity Learning			Students Model		
		MAE	**4-Class PTA_0**	**P/F**	**MAE**	**4-Class PTA_0**	**P/F**
FS1	**Average**	**13.42**	**0.45**	**0.60**	14.09	0.43	0.56
FS3	**Average**	10.85	**0.54**	**0.73**	**10.78**	0.51	0.70

6.5 Discussion

Our study aimed to answer two main research questions: **RQ1**: Is there a correlation between sentiment in teacher reports and student performance? and **RQ2**: Can similarity learning utilizing sentiment scores enhance teacher report embeddings for a more accurate performance prediction model? As shown by the Pearson correlation coefficient in Section 6.3.2, we observed a moderate positive correlation between sentiment in teachers' reports and students' final performance. This finding aligns with literature that suggests teacher perceptions and attitudes toward students may contain predictive information about their academic success.

Utilizing this correlation, our method aimed to capture the nuanced information contained in these reports by learning sentiment-based similarity. Optimized embeddings using our method, on average, significantly outperformed unoptimized embeddings in the feature sets FS_1 and FS_3. Notably, optimized embeddings in FS_3 demonstrated the highest overall accuracy, particularly for pass/fail (P/F) predictions. This indicates that embedding optimization can enhance predictive power by capturing complex sentiment-related patterns that may not be as well represented in unoptimized embeddings.

6.6 Summary

In this study, we analyzed correlation between extracted sentiment scores from teacher reports and students' final performance. We then proposed a similarity learning approach to enhance academic performance prediction models by optmizing embeddings based on this correlation. We define a sentiment-based similarity metric and use it to finetune Sentence BERT embeddings. Our results showed that sentiment scores extracted from teacher reports can enhance academic performance prediction models. We demonstrated that incorporating similarity learning based on such correlation can further improve the performance of the prediction models. Our research encourages the study of other factors that could be correlated with performance and the usage of such attributes in similarity learning to gain more accurate and reliable results. Overall, this study contributes to the field of academic

performance prediction by providing an effective and straightforward approach that can help build a more reliable prediction model and in turn help teachers, students, and educational institutions reach their full potential.

7

ASAS: Comparative Study in Arabic Short Answer Scoring

7.1 Introduction

Automatic grading has been a prominent subject of discussion in the field of AI in educationfor over fifty years [79]. This technology offers numerous benefits, addressing key challenges in modern education. One major advantage is its ability to efficiently process large volumes of student work, which is particularly useful in massive open online courses (MOOCs) and large classrooms where manual grading would be impractical [80]. Automated systems ensure consistency and objectivity, reducing grading disparities and supporting educational equity by eliminating variations that may arise from human biases or fatigue [81]. Additionally, automatic grading provides students with immediate feedback, which is crucial for learning as it enables them to quickly identify mistakes and improve [82]. This frees up teachers to focus more on personalized instruction allowing for deeper student engagement. By decreasing the labor needed for grading, automatic grading offers a cost-effective solution for institutions, allowing resources to be directed toward other needs. As AI technology advances, automatic grading is set to have an even bigger impact on education.

Tests can employ various question formats and scoring methods to evaluate learning outcomes. These formats range from basic multiple-choice questions to more complex ones requiring written responses in natural language such as short answers or essays. Automatic Short Answer Scoring (ASAS) is a field that specifically addresses the automated assessment of short answer

responses [83]. To distinguish short answers from other written response types, five key criteria must be met:

1. **Length:** Responses fall between a phrase and a paragraph.
2. **Knowledge Type:** Responses require recall of external knowledge rather than recognition of information within the question.
3. **Response Format:** Responses must be expressed in natural language.
4. **Assessment Focus:** Evaluation emphasizes content rather than writing style.
5. **Question Design:** Questions must be structured objectively, limiting response scope while remaining open-ended.

Advances in NLP have significantly accelerated the development of ASAS, particularly with the advent of large deep learning models capable of handling complex learning representations. ASAS approaches generally fall into two main paradigms: instance-based and reference-based methods [84]. Instance-based methods, also known as response-based methods, train models on extensive datasets of individual student responses, enabling the system to learn from a wide variety of answers. In contrast, reference-based methods evaluate responses by comparing them to reference answers, making them less reliant on large datasets and potentially offering greater interpretability [85].

In this study, we contribute to ASAS research by providing an empirical comparison of two state-of-the-art distinct methods-an instance-based approach and a reference-based approach. In our first instance-based approach, we leverage a pre-trained language model (i.e. BERT) and train it on different questions with a shared type. For each instance, we create an input structure that provides contextual information for the model. In our reference-based approach, we train a score-based semantic similarity model using SentenceTransformers [14]. Our results suggest that while both models achieve comparable performance in evaluating UA in traditional settings, the reference-based model shows promising potential in handling UQ in zero-shot scenarios. Additionally, we present a question type classification model and enhance training data with GPT-3.5-generated synthetic examples to

improve performance in low-resource conditions. Finally, we make the code and models publicly[1] available to facilitate future research in this area.

7.2 Related Work

Research in ASAS can be categorized into two main paradigms, instance-based or similarity-based methods [84]. Most recent ASAS research follows the instance-based paradigm, where algorithms are trained primarily using a large set of student answers to learn about the features of correct and incorrect responses. With the rise of large language models and transfer learning, most studies typically involve fine-tuning BERT such as in the work by Lun et al. Another example is the work of Nael et al. where they fine-tune BERT and ELECTRA models on a machine-translated benchmark dataset. Condor et al. used SBERT embeddings to train a model with an instance-based approach rather than using it in a similarity-based approach. Fernandez et al. introduced a single shared scoring model for multiple questions using a specified input structure that provides contextual information for each item. Similarly, to score mathematical questions, Zhang et al. use an in-context learning approach that provides scoring examples as part of the input to a MathBERT model to promote generalization.

On the other hand, in the reference-based approach, student answers are evaluated by comparing them to one or more target answers. Judgments of correctness are thus determined based on their similarity to a reference solution. In early work, reference-based approaches mainly employed feature-engineering methods such as utilizing string-based or corpus-based similarity methods [91] and n-grams [92]. More recently, Meccawy et al. conducted a comparative study evaluating the efficiency of different word embedding approaches for conducting feature vectors. In their study, Wang et al. introduced innovative metrics for score-based similarity to construct a text representation space that is optimized for both inter and intra-level distinctions, leading to improved scoring efficiency. In our reference-based approach, we define score-based similarity in a manner similar to what they have presented in their research.

[1] Our code can be found at: `https://github.com/mennafateen/SSS-vs-InCML`

7.3 Dataset Description

In this study, we use the AR-ASAG dataset, which is the first publicly available dataset for automatic short-answer scoring in Arabic [95]. The dataset consists of 2133 short answers written by graduate students in response to 48 questions. The questions are taken from 3 different exams on cybercrime where each exam consists of 16 questions. The questions in the exams could be classified into 5 categories based on the type of answer they expect, namely: *define*, *explain*, *consequences*, *justify*, and *compare*.

The answers in this dataset were independently annotated by two human raters on a scale of 0 to 5 where 0 is completely incorrect and 5 is considered a perfect answer. The raters were instructed to assign a score based on the similarity of the student's answers to a reference answer given for each question. Determining the similarity between two answers not only is a subjective task but also requires a deep understanding of the topic. In cases like this, where no detailed scoring rubric is provided, the raters can find it especially difficult to determine the precise degree of similarity. This is reflected in the low inter-rate agreement of 35%. However, this is expected since the raters were also given the freedom to assign intermediate scores such as 4.5 or 3.25, etc.

In our study, we treat the scoring problem as a classification problem instead of a regression one. We discretize the scores into 6 categories, 0, 1, 2, 3, 4, and 5 by taking the rounded-down median after ceiling the scores to the nearest 0.5. This is done to increase the inter-rate agreement to 56% instead of 35%. The distribution of the scores in the dataset can be seen in Figure 7.1 where we can observe the majority of the scores being concentrated in the range of 3 to 4.

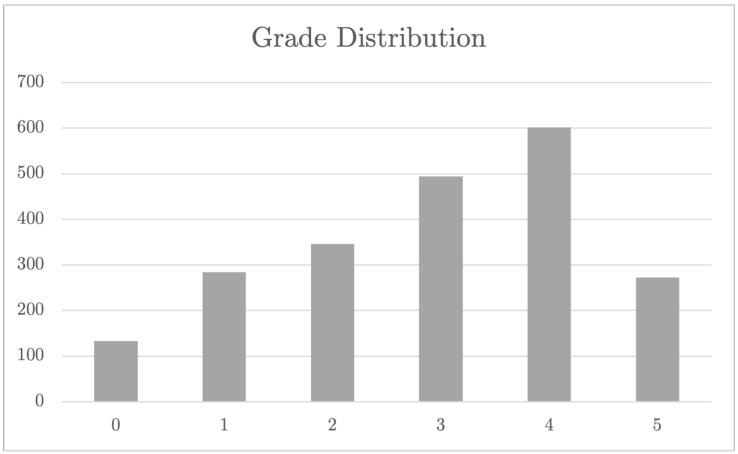


Figure 7.1: Distribution of the scores in the dataset.

7.4 Methodology

Problem Formulation

We formulate the problem of automatic short answer scoring as follows: Given a question q, a short answer a, and a reference answer a^*, the goal is to predict a score $s \in [0, 5]$ that represents the quality of the answer. Usually, each question is treated as a separate task where a separate model is trained for each task or question. However, this approach is not feasible in low-resource settings where there is a lack of annotated data. Hence, we propose a general in-context question-based scoring framework for automated scoring of short-answer questions where we divide the scoring problem into two sub-problems, question-category-based scoring and question category classifying.

The question-category-based scoring problem can be formally defined as follows: we have a set of tasks $T = \{t_i\}_{i=1}^{N}$, where each task t_i is defined by a question q_i, its reference answer a_i^*, and a set of instances $D_i = \{(a_{i,j}, s_{i,j})\}_{j=1}^{M_i}$, where M_i is the number of instances for the i-th task. The goal is to learn a function $f : (q, a^*, a) \rightarrow s$ that can generalize to both unseen answers and unseen questions with a small number of annotated examples. To solve the defined problem, we propose and compare two main approaches, namely, In-Context Meta-Learning Model (InCML) and Score-based Semantic Similarity (SSS). We describe each approach in detail in the following

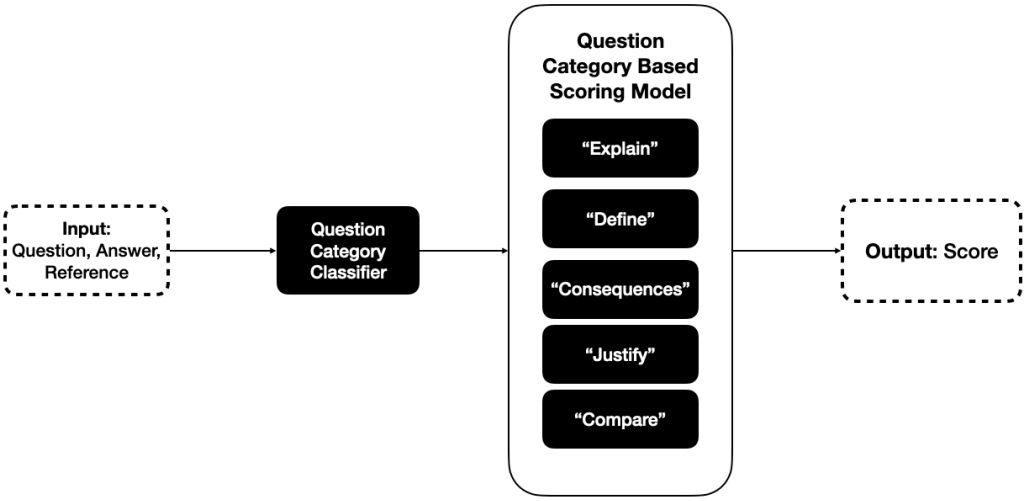


Figure 7.2: Overview of the in-context question-based scoring framework.

subsections. Within each approach, we train one model per question category, resulting in 5 models per approach.

To facilitate the selection of the most suitable question-category-based scoring model for a given question, an auxiliary question category classifier is trained to identify the question category of a given question. The output of this model should then serve as a guide for selecting the most suitable model for a given question. This is illustrated in Figure 7.2.

Question Category Classification

To construct a balanced training dataset, we train the model on the first 4 questions from each question category and set aside the remaining questions for testing. We use the SetFit framework [96] and use the pretrained AraBERT model [97] as the pretrained body. We then generate 50 pairs for contrastive learning and train the model for 5 epochs. The classification head is a logistic regression layer that takes the output of the last layer of the pretrained body as input. With this 4-shot training setup, the model achieves a mere accuracy of 0.357. To address this, we propose to augment the training data with synthetic examples generated by GPT-3.5. For each question category, we instruct the model to provide x more examples, for instance:

Provide five more examples that are similar to the following using the same "Define" prompt: list

- عرف مصطلح الجريمة الإلكترونية (Define the term cybercrime)
- عرف مصطلح أمن المعلومات (Define the term information security)
- عرف مصطلح الهندسة الاجتماعية النفسية (Define the term psychological social engineering)
- عرف مصطلح تبييض أو غسيل الأموال (Define the term money laundering).

We experiment with different values of x and report the results in Table 7.1 where in Aug_1, Aug_2, and Aug_3, we augment the training data so that the total number of examples are 45, 125, and 250 respectively. As shown in the table, the model's performance significantly improves as we increase the number of augmented examples, achieving an accuracy of 0.893 with Aug_3 and an F1-score of 0.821. With this question classification model experiment, we show that in low-resource settings, a potential solution that could be explored is to augment available samples using generative large language models such as GPT-3.5.

Table 7.1: Question category classification results on the testing set with different numbers of augmented examples

Metric	Original	Aug_1	Aug_2	Aug_3
Accuracy	0.357	0.607	0.714	**0.893**
F1-Score	0.443	0.645	0.699	**0.821**
Precision	0.511	0.711	0.733	**0.820**
Recall	0.724	0.806	0.841	**0.900**

Instance-based: In-Context Meta Learning Model

The i approach draws inspiration from the work of [89]. Building upon their foundational concepts, we apply this approach to the unique domain of cybersecurity short answer scoring in Arabic. To introduce context, we input the answers using a template that is constructed by concatenating the target answer to be scored a_j, question q_i, and its reference answer,

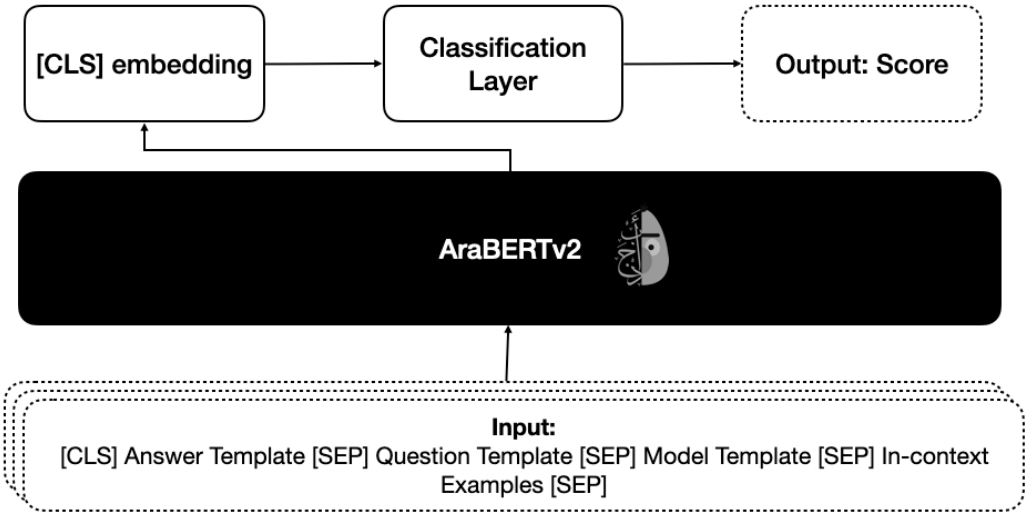


Figure 7.3: In-context meta-learning model.

a_i^*. We additionally include a set of K in-context examples E_i that are randomly sampled from the training set D_{train} for the i-th task or question. We build a template for each component by adding semantically meaningful task instructions as shown in Table 7.2. Moreover, we convert the numeric scores s_j in the in-context examples E_i to meaningful words such that: *0:* ضعيف جدا *(very poor), 1:* ضعيف *(poor), 2:* متوسط *(fair), 3:* جيد *(good), 4:* جيد جدا *(very good), 5:* ممتاز *excellent*.

We then concatenate the templates to form the final input to the model. During inference time, the same templates are created for the input components where the in-context examples are fetched from the training set for seen questions only.

We train our model on the union of training datasets for all items or questions per question category $\cup_{i=1}^{5} D_{train}^{i}$, instead of training a separate model for each item, thus reducing the number of model parameters and required storage space. Figure 7.3 illustrates the in-context meta-learning model.

Reference-based: Score-based Semantic Similarity

When human experts are asked to score answers to open-ended questions, they usually compare the answers to a reference answer and assign a score

Table 7.2: Input components and templates for the in-context meta-learning model.

Input Component	Template	Sample
Student Answer	قيم هذه الإجابة: x	قيم هذه الإجابة: العلم الذي يستخدم التحليل الاحصائي لصفات الانسان الحيوية وذلك للتأكد من هويته الشخصية (Grade this answer: The science that uses statistical analysis of a person's vital characteristics to confirm his identity)
Question	السؤال: q_i	السؤال: عرف مصطلح القياس الحيوي (Question: Define the term biometrics)
Reference	النموذج: a_i^*	النموذج: هو العلم الذي يستخدم التحليل الإحصائي لصفات الإنسان الحيوية وذلك للتأكد من هويته الشخصية بإستخدام صفاته الفريدة وهي صفات سلوكية وصفات فيزيائية (Reference: It is the science that uses statistical analysis of a person's vital characteristics to confirm his identity using his unique characteristics, which are behavioral characteristics and physical characteristics.)
Grades	تقييم : ...	تقييم : ضعيف جدا ضعيف متوسط جيد جيد جدا ممتاز (Grades: very poor, poor, fair, good, very good, excellent)
Examples	مثال: $x_{\neg j}$	مثال: هو علم يدرس حالة الإنسان الفريدة التي تميز شخصًا عن آخر تقييم : ضعيف (Example: It is a science that studies the unique human condition that distinguishes one person from another Grade: Weak)

based on the similarity between the two answers. In this approach, we propose to train a model that can mimic this process by learning to assign a score to a given answer based on its similarity to a reference answer.

To achieve this, we perform the following steps: First, we construct a simple in-context template for each answer to be graded a_j by prepending the question q_i to the answer. It has been shown that incorporating the question in the input can improve the performance of ASAS models [98]. Then, we define score-based similarity as follows: Given a pair of answers a_x and a_y with their scores s_x and s_y, the similarity between the two answers is defined as:

$$sim(a_x, a_y) = \frac{s_x}{s_y}; (s_x \leq s_y) \tag{7.1}$$

For each task/question i, we then construct a dataset $\mathcal{D}_i$ of answer pairs annotated together via the score-based metric indicating their similarity as shown in Equation 7.2, where X is the number of examples per score k and a_k is a student answer that was graded k.

$$\mathcal{D}_i = \{\{(a_k, a_{\neg k}, sim(a_k, a_{\neg k}))\}_{x=0}^{X}\}_{k=0}^{5} \tag{7.2}$$

We experiment with 3 different settings of X when constructing the dataset with $X = 30$, $X = 50$, and a final configuration where we account for the distribution of the scores in the training set so that the number of samples per score is $50 \times \frac{N_k}{\sum_{k=0}^{5} N_k}$

We then train one model on the union of training datasets for all items or questions per question category $\cup_{i=1}^{5} D_{train}^{i}$, instead of training a separate model for each item as described in the InCML approach. In this approach, using SBERT, we fine-tune the pretrained AraBERT model through a Siamese network structure where we train the model using a cosine-similarity loss function. For each answer pair a_x and a_y in the union dataset, we pass both answers through the model which generates an embedding u_x and u_y for each answer. The gold similarity score is then compared with the cosine similarity between the generated embeddings as shown in Equation 7.3. Figure 7.4 illustrates the score-based semantic similarity model.

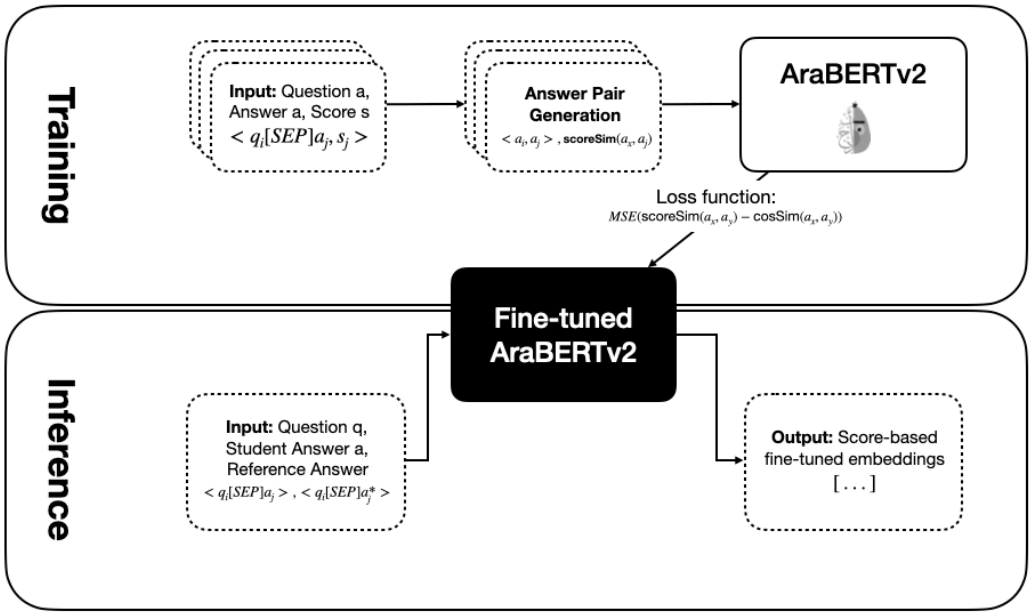


Figure 7.4: Score-based Semantic Similarity Model.

$$\mathcal{L} = \frac{1}{n}\sum^{n}(sim(a_x, a_y) - \cos(u_x, u_y))^2 \tag{7.3}$$

7.5 Experimental Results

Evaluation Metrics

To evaluate the performance of the proposed approaches, we use two metrics, namely, Quadratic Weighted Kappa (QWK) and PTA. QWK is a commonly used metric in ASAS that measures the agreement between raters. We use the Cohen's kappa implementation from scikit-learn, which is calculated as:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{7.4}$$

where p_o is the empirical probability of agreement on the label assigned to any sample (the observed agreement ratio), and p_e is the expected agreement when both annotators assign labels randomly. p_e is estimated using a per-annotator empirical prior over the class labels. (PTA$_x$) measures the

percentage of answers that are scored correctly or within x points of the gold score. PTA_0 would be equivalent to accuracy while PTA_1 also includes answers that are scored within 1 point of the gold score (e.g. 3 is considered correct if the gold score is 2 or 4) and so on.

Experimental Setup

We use AraBERT as the pretrained body for both approaches. In the InCML approach, we test 3 settings of K in-context examples, InCML_0, InCML_1 and InCML_3. In training, we use the Adam optimizer with a learning rate of 1e-5 and a batch size of 8 and train the models for 6 epochs.

Similarly, in SSS, we train the model for 6 epochs but use a batch size of 16 instead. We experiment with 3 different settings of X when constructing the dataset. Specifically, SSS_{30}, SSS_{50}, and SSS_{50W}, where $X = 30$, $X = 50$, in Equation 7.2 and a final configuration where we account for the distribution of the scores in the training set so that the number of samples per score is $50 \times \frac{N_k}{\sum_{k=0}^{5} N_k}$.

Results

We undertake two experiments to evaluate the performance of the proposed approaches. In the first experiment, we test the models' performance on unseen answers. We set aside 10% of the answers from each question category for testing. In the second experiment, we evaluate the performance of the models on unseen questions. We set aside the first question and its answers from each question category for testing. This setting is considered a zero-shot learning scenario since the models are not trained on any examples from the test set. The results of both experiments are shown in Table 7.3 and 7.4 respectively. As a baseline for comparison, we report the results of a majority class classifier and the QWK and PTA between the rounded-up grades of the two human raters.

Table 7.3: Experimental results for the UA split.

		Human	MV	InCML$_0$	InCML$_1$	InCML$_3$	SSS$_{30}$	SSS$_{50}$	SSS$_{50W}$
	QWK	*0.676*		0.611	0.656	**0.697**	0.591	0.593	0.632
	PTA$_0$	*0.357*	0.357	0.286	**0.500**	0.404	0.357	0.393	0.357
P1	**PTA$_1$**	*0.893*		0.857	0.843	0.889	0.857	0.857	**0.893**
	PTA$_2$	*0.929*		0.964	0.954	0.964	**1.000**	**1.000**	**1.000**
	PTA$_3$	*1.000*		**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	QWK	*0.788*		0.722	0.668	0.760	0.718	**0.763**	0.718
	PTA$_0$	*0.568*	0.239	0.432	0.443	**0.451**	0.375	0.443	0.364
P2	**PTA$_1$**	*0.920*		0.909	0.875	0.936	0.932	**0.955**	0.943
	PTA$_2$	*0.977*		0.989	0.963	**0.998**	0.989	0.989	0.977
	PTA$_3$	*1.000*		**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	QWK	*0.749*		**0.798**	0.774	0.727	0.557	0.581	0.660
	PTA$_0$	*0.385*	0.308	0.385	**0.542**	0.385	0.154	0.385	0.385
P3	**PTA$_1$**	*0.846*		**0.923**	0.869	0.919	0.846	0.846	0.885
	PTA$_2$	*0.962*		**1.000**	**1.000**	0.950	**1.000**	0.962	0.962
	PTA$_3$	*1.000*		**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	QWK	*0.666*		0.602	**0.649**	0.519	0.614	0.613	0.515
	PTA$_0$	*0.533*	0.311	0.400	0.464	0.351	**0.467**	**0.467**	0.444
P4	**PTA$_1$**	*0.822*		0.867	0.813	0.733	**0.911**	**0.911**	0.844
	PTA$_2$	*0.978*		**1.000**	0.989	0.936	**1.000**	**1.000**	**1.000**
	PTA$_3$	*1.000*		**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	QWK	*0.716*		**0.696**	0.000	0.070	0.529	0.606	0.405
	PTA$_0$	*0.526*	0.421	0.421	0.421	0.368	**0.474**	0.421	0.368
P5	**PTA$_1$**	*0.842*		**0.842**	0.684	0.737	0.789	**0.842**	0.789
	PTA$_2$	*0.947*		**1.000**	0.789	0.842	**1.000**	**1.000**	**1.000**
	PTA$_3$	*1.000*		**1.000**	0.947	0.947	**1.000**	**1.000**	**1.000**

Table 7.4: Experimental results for the UQ split.

		Human	MV	InCML$_0$	InCML$_1$	InCML$_3$	SSS$_{30}$	SSS$_{50}$	SSS$_{50W}$
	QWK	*0.743*		0.145	0.000	0.063	0.620	0.599	**0.627**
	PTA$_0$	*0.596*	0.383	0.213	0.000	0.064	**0.447**	0.404	**0.447**
P1	**PTA$_1$**	*0.957*		0.553	0.043	0.149	**1.000**	**1.000**	**1.000**
	PTA$_2$	*1.000*		0.936	0.085	0.404	**1.000**	**1.000**	**1.000**
	PTA$_3$	*1.000*		0.979	0.319	0.766	**1.000**	**1.000**	**1.000**
	QWK	*0.876*		0.000	-0.055	0.026	**0.645**	0.558	**0.645**
	PTA$_0$	*0.833*	0.416	0.167	0.083	0.167	**0.500**	**0.500**	**0.500**
P2	**PTA$_1$**	*0.916*		0.167	0.167	0.167	**0.917**	0.833	**0.917**
	PTA$_2$	*1.000*		0.417	0.417	0.333	**1.000**	**1.000**	**1.000**
	PTA$_3$	*1.000*		0.833	0.417	0.833	**1.000**	**1.000**	**1.000**
	QWK	*0.752*		0.000	-0.014	0.000	0.488	0.446	**0.495**
	PTA$_0$	*0.714*	**0.469**	0.082	0.082	0.082	0.204	0.224	0.204
P3	**PTA$_1$**	*0.918*		0.184	0.184	0.184	**0.918**	0.878	**0.918**
	PTA$_2$	*0.959*		0.347	0.347	0.347	**1.000**	**1.000**	**1.000**
	PTA$_3$	*0.980*		0.531	0.531	0.531	**1.000**	**1.000**	**1.000**
	QWK	*0.774*		0.101	-0.015	-0.161	0.254	0.284	0.365
	PTA$_0$	*0.395*	0.271	0.208	0.125	0.042	**0.333**	**0.333**	**0.333**
P4	**PTA$_1$**	*0.875*		0.500	0.292	0.375	0.688	**0.708**	0.688
	PTA$_2$	*0.979*		0.646	0.375	0.771	0.938	0.938	**1.000**
	PTA$_3$	*1.000*		0.896	0.646	0.958	**1.000**	**1.000**	**1.000**
	QWK	*0.358*		0.000	**0.069**	0.003	0.000	0.029	0.024
	PTA$_0$	*0.667*	**0.500**	0.000	0.146	0.000	0.042	0.042	0.021
P5	**PTA$_1$**	*0.979*		0.000	0.396	0.021	0.500	**0.563**	0.500
	PTA$_2$	*1.000*		0.000	0.563	0.188	**1.000**	**1.000**	**1.000**
	PTA$_3$	*1.000*		0.042	0.604	0.667	**1.000**	**1.000**	**1.000**

Table 7.5: Combined average experimental results for the UQ split and the UA split.

Metric	Type	Human	MV	InCML$_0$	InCML$_1$	InCML$_3$	SSS$_{30}$	SSS$_{50}$	SSS$_{50W}$
QWK	UQ	0.701	–	0.049	-0.003	-0.014	0.401	0.383	0.431
	UA	0.719	–	0.686	0.549	0.555	0.602	0.631	0.586
PTA$_0$	UQ	0.641	0.408	0.134	0.087	0.071	0.305	0.301	0.301
	UA	0.474	0.327	0.385	0.474	0.392	0.365	0.422	0.384
PTA$_1$	UQ	0.929	–	0.281	0.216	0.179	0.805	0.796	0.805
	UA	0.865	–	0.880	0.817	0.843	0.867	0.882	0.871
PTA$_2$	UQ	0.988	–	0.469	0.357	0.409	0.988	0.988	1.000
	UA	0.959	–	0.991	0.939	0.938	0.998	0.990	0.988
PTA$_3$	UQ	0.996	–	0.656	0.503	0.751	1.000	1.000	1.000
	UA	1.000	–	1.000	0.989	0.989	1.000	1.000	1.000

7.6 Discussion

7.6.1 Unseen Answers

As shown in Table 7.3, compared to the majority class classifier, both approaches with different configurations outperform the baseline in all question categories in the unseen answers experiments. Comparing the model's performance to human performance, we observe that with question category $P1$ and $P3$, InCML_0 and InCML_3 outperform the human raters in terms of QWK. In terms of PTA_0, InCML_1, InCML_3, and additionally SSS_{50} outperform the human raters with question types $P1$ while InCML_1 again outperforms the human raters with type $P3$ questions. In the remaining question categories, the performance of both approaches in terms of QWK is marginally below the QWK achieved between the human raters with the in-context meta-learning approach showing a tendency to outperform the score-based similarity approach.

In the in-context meta-learning approach, we observe that the performance of the model does not necessarily improve as we increase the number of in-context examples. In fact, in some cases, the performance decreases. We speculate that this might be attributed to potential overfitting on the in-context examples. It is also important to note that the performance of InCML fluctuates depending on the in-context examples that are extracted from the training set which introduces inherent instability. On the other hand, in the case of the semantic score-based similarity approach, an increase in the number of examples per score generally corresponds to improved model performance.

In the unseen answers experiment, with a few training examples, we observe that while both approaches have comparable performance to the human raters, the instance-based in-context meta-learning approach generally gives better performance compared to the reference-based approach.

7.6.2 Unseen Questions

In Table 7.4, we see that the overall performance of the models in the unseen questions experiment, or in zero-shot settings, is lower than the performance of unseen answers. However, we observe that the PTA_0 of the models is still higher than the majority class classifier in most question category models using our reference-based, SSS_{30}, SSS_{50} and SSS_{50W} methods.

Compared to the instance-based InCML approach, it is evident that the reference-based SSS approach proposed gives higher performance showcasing its reduced data hunger advantage and its ability to generalize to new questions.

The combined average results in Table 7.5 offer a comprehensive perspective on the models' performance across unseen answers (UA) and unseen questions (UQ), highlighting the strengths of each approach in different contexts. For unseen answers, the InCML methods, particularly InCML0, excelled, achieving the highest QWK of 0.686. This suggests that this method effectively adapts to new student answers when sufficient training data is available. In contrast, reference-based approaches such as SSS50 also performed competitively, showcasing their ability to provide consistent results even in less controlled settings.

However, the narrative shifts in the zero-shot scenario of unseen questions. Here, reference-based methods like SSS_{50W} emerged as the most reliable, achieving the highest QWK (0.431) and consistently outperforming instance-based methods across all metrics. This demonstrates their robust generalization capabilities when faced with entirely new question categories, a critical advantage in data-sparse or adaptive learning environments. These results demonstrate the complementary strengths of the two approaches. Instance-based methods excel in data-rich scenarios, adapting well to familiar answer patterns, whereas reference-based methods excel in zero-shot settings, leveraging their flexibility and robust representations. This dual capability suggests that future work could explore integrating these approaches to take advantage of their respective strengths, creating a hybrid framework that adapts dynamically to varying levels of data availability and task complexity.

7.7 Summary

In this paper, we propose a general in-context question-type-based scoring framework for automated scoring of short-answer questions. We divide the scoring problem into two sub-problems, namely, question category classification and question-category-based scoring. For question-category classification, we use a few-shot, prompt-free framework to train the model. We also show that with data augmentation using GPT3.5, the performance could be significantly increased. We then propose two main approaches for the question-category-based scoring problem, namely, instance-based in-context meta-learning and reference-based semantic similarity. Utilizing the only publicly available Arabic ASAG dataset, we evaluate both approaches in their ability to generalize to unseen answers and unseen questions. Experimental results show that both proposed approaches outperform the majority class classifier and are comparable to human raters when grading unseen answers. However, the performance is highly question-type-dependent and no particular approach is consistently better than the other. In zero-shot settings, when generalizing to unseen questions, we observe a tendency for the reference-based semantic similarity approach to outperform the instance-based in-context meta-learning approach. We thus believe that in classroom settings, the reference-based semantic similarity approach could be a more suitable solution due to its superiority in zero-shot settings.

Limitations

In this paper, we presented a comparison between a specific instance-based and reference-based approach, thus our findings are limited to these methods and cannot be generalized to different methods. This study was also limited to a question-category-based scoring framework and while preliminary experiments were conducted, we did not compare with specific question-category-based models or cross-question-category models for a more straightforward and comprehensible comparison. Due to the scarcity of resources, our comparison also relies on a specific dataset, which does not encompass

the full diversity of responses or topics encountered in a real-world educational setting. Furthermore, since we use an Arabic dataset, we adapted a BERT model pre-trained on Arabic data but have not presented a comparison with a language-agnostic model. Finally, while we briefly touched upon the potential of reference-based approaches in offering explainability, we have not delved into the topic of interpretability of the provided models. Understanding why a model assigns a specific score to an answer is essential for educational applications, as it can provide valuable feedback to students, however, it is beyond the scope of this paper and is left for future work.

8 ASAS-F: A Modular RAG-based System

8.1 Introduction

Feedback plays a critical role in student learning. Extensive research demonstrates that detailed and timely feedback significantly improves student performance by guiding learning and correcting misunderstandings [3, 99]. However, providing such elaborated feedback, especially for short answer questions, is a time-consuming and labor-intensive task for educators, particularly when managing large numbers of students. To address this, ASAS systems have been developed to quickly assess student responses [100, 101]. Despite their utility, ASAS systems primarily focus on scoring correctness, offering holistic scores without the detailed feedback students need to understand their mistakes [83].

Current ASAS systems are typically trained on classification or regression tasks, where the objective is to assign a score to a student's answer. While effective in evaluating answers, these systems lack the explainability needed for two key purposes: (1) helping students understand the reasoning behind their errors and the feedback given and (2) fostering trust in the scoring system [102, 103, 104]. Without this granularity, students miss opportunities to improve, and educators may be reluctant to adopt automated grading systems.

In response to these limitations, recent research has focused on developing ASAS systems with feedback **automatic short answer scoring with feedback (ASAS-F)** that not only score student answers but also generate detailed feedback [105, 106, 107]. These systems aim to provide students with actionable feedback that explains the reasoning behind their scores, helping

them identify and correct their mistakes. Existing ASAS-F systems rely on resource-intensive fine-tuning with limited datasets, specific tasks [108, 109] or complex prompt engineering, all of which may not generalize well across contexts. Moreover, evaluating the quality of the textual feedback is usually only done using traditional statistical metrics, which do not capture the main aspects of quality, such as accuracy and clarity.

To overcome these challenges, we introduce a novel ASAS-F system that leverages LLMs within a RAG [110] framework. Inspired by similarity-based scoring [85], our method retrieves the most similar answers from a short answer scoring feedback dataset using the ColBERT retrieval [27] model. These retrieved answers serve as few-shot examples, allowing LLMs to generate both accurate scores and detailed feedback.

Our study explores the following research questions:

- **RQ1:** How does the performance of our modular ASAS-F system compare to state-of-the-art models in automatic short answer scoring?
- **RQ2:** When labeled training data is available, how can we optimize prompts and few-shot examples to improve our ASAS-F performance in an efficient way?
- **RQ3:** How accurate and clear is the feedback generated by our ASAS-F system?

Our experiments on the research questions demonstrate that the proposed approach significantly minimizes the need for extensive fine-tuning, resulting in a computationally efficient solution that maintains high accuracy while delivering clear, accurate feedback. The system is designed in a modular fashion, allowing it to adapt easily to various educational tasks without the need for extensive prompt engineering. By reducing reliance on large curated datasets, our method generates meaningful feedback with minimal adaptation, offering a scalable and cost-effective solution for educational applications. We share the code, model outputs and their evaluations from our experiments on Github to promote transparency and reproducibility [1].

To summarize, the key contributions of our work are:

[1] https://github.com/mennafateen/modular-asas-f-rag

- We propose a novel ASAS-F system that scores short answers and generates detailed feedback using LLMs and RAG.
- We propose a modular approach that minimizes the need for prompt engineering, making the system adaptable to diverse tasks
- We propose a RAG-based approach that eliminates the need for computationally expensive fine-tuning and large datasets, making the solution scalable and efficient.

8.2 Related Work

8.2.1 Automatic Short Answer Scoring

Automatic scoring has been studied extensively in the field of NLP and educational technology [83, 111, 112, 113]. ASAS systems aim to grade students' short answers automatically, offering immediate feedback, reducing teachers' workload, and streamlining the grading process.

Traditional ASAS systems are often rule-based and rely on handcrafted features and heuristics to score short answers. While these systems are effective and have shown promising results on benchmark datasets, they require significant manual effort to develop [101, 114]. With the rise of deep learning, neural network-based and Transformer-based ASAS systems have been introduced, which automatically learn features from data and outperform traditional rule-based systems [100, 115].

Among the newer methods, instance-based approaches have gained traction. These approaches involve using pretrained models to map student responses directly to scores by pooling token representations across model layers [116]. These methods capture the context and semantics of input text, enabling them to outperform traditional rule-based systems. However, they often require large amounts of labeled data to achieve high performance and struggle in few-shot settings where data is limited. Comparisons between instance-based and similarity-based approaches have shown the latter to be more effective in zero-shot settings [85, 94, 117]. Additionally, recent advancements in dense

information retrieval have proven effective in automatic essay scoring tasks [118]. Building on these insights, our work introduces a similarity-based approach using retrieval-augmented generation to enhance the performance of ASAS-F systems.

8.2.2 ASAS Using Generative Models

Despite the success of neural network-based ASAS systems, many of the proposed models are computationally expensive and require large amounts of data to achieve good performance. In many real-world scenarios, labeled data for training ASAS systems is scarce, making it challenging to train these models.

Few studies have explored the use of LLMs for ASAS tasks. One study investigated the usage of LLMs for ASAS, focusing on zero-shot and few-shot settings across three diverse datasets [119]. The results showed that LLMs were able to achieve strong performance on tasks involving general knowledge questions but struggled with questions that required domain-specific knowledge or complicated reasoning.

Another study explored the use of proprietary LLMs, namely GPT3.5 and GPT4, for ASAS tasks [120], [121]. The findings indicated that while these models showed promise, their performance was negatively correlated with the length of student answers.

In a recent study, researchers examined the use of ChatGPT for holistic essay scoring and compared its performance to human raters [122]. The findings showed that ChatGPT's scoring was not statistically significantly different from human ratings and demonstrated substantial agreement. However, this study highlighted the need for further research particularly regarding improving scoring precision.

Additionally, research assessed the possibility of using ChatGPT (GPT-3.5) for automated grading [106]. The results showed a low correlation between the scores given by ChatGPT and human graders, with ChatGPT tending to give middle scores more frequently despite strong variation in student

answers. Moreover, the study found that minor changes in answers could lead to significant differences in the scores given by ChatGPT.

8.2.3 ASAS with Feedback

While scores provide a general overview of the quality of the answer, they lack the granularity and explainability that students need to improve their answers. Elaborated feedback would also increase the trustworthiness of the scoring system.

The Short Answer Feedback (SAF) dataset [107] includes short answer questions in communication network topics, each with a reference answer, given score, label, and content-focused elaborated feedback. The SAF dataset provides a benchmark for evaluating ASAS systems that generate feedback, enabling the development of more informative and actionable systems. It also established a baseline for ASAS systems using the T5 model fine-tuned on the SAF dataset, demonstrating the feasibility of generating feedback for short answers.

A framework was proposed [105] that distils the rationale generation capability of ChatGPT by designing several prompt templates to generate feedback from ChatGPT, then fine-tuning a smaller language model on the refined generated outputs. The study showed that the fine-tuned model outperformed the original ChatGPT model on QWK scores.

Further research explored four different approaches for generating counterfactual feedback for short answers [123]. While some of the modified feedback could be graded as correct by automatic scoring systems, a domain expert deemed them incorrect.

Finally, one study explored the usage of LLMs for automated essay scoring (AES) with rationale [124], though the primary focus in ASAS is on content quality rather than writing style and structure. To the best of our knowledge, there is limited research evaluating LLMs for ASAS-F tasks, particularly in zero-shot and few-shot settings. Our work aims to bridge this gap by exploring the potential of LLMs for scoring short answers and generating feedback.

8.3 Methodology

In this section, we present our approach for building the ASAS-F system using LLMs and outline the methodologies proposed for different scenarios, including zero-shot ASAS-F, few-shot ASAS-F with automatic optimization and few-shot ASAS-F using RAG.

8.3.1 Problem Formulation

Traditional ASAS systems typically use classification or regression models to score student answers, focusing solely on assigning a numerical score. While effective for grading, these systems lack the ability to provide detailed feedback or explain the rationale behind a score, limiting their usefulness for student learning. In addition to such numeric scores and labels, ASAS-F systems additionally generate detailed elaborated feedback [3] that explains the reasoning behind the score.

Formally, the ASAS-F system can be expressed as:

$$\text{ASAS-F}(q, a, s) \rightarrow (y, l, f) \tag{8.1}$$

where given a question q, a reference answer a, and a student answer s, the system assigns:

- A score $y \in [0, 1]$ (indicating correctness),
- A label $l \in \{\text{correct}, \text{incorrect}, \text{partially correct}\}$,
- Feedback f that explains the reasoning behind the score.

In a zero-shot ASAS-F setting, the system utilizes the knowledge embedded in pre-trained LLMs to provide scoring and feedback without requiring labeled training data. When labeled data $\mathcal{D}$ is available, we explore two few-shot ASAS-F approaches: one using automatic prompt optimization, and another using RAG for enhanced performance.

8.3.2 ASAS-F-Z: Zero-Shot ASAS-F

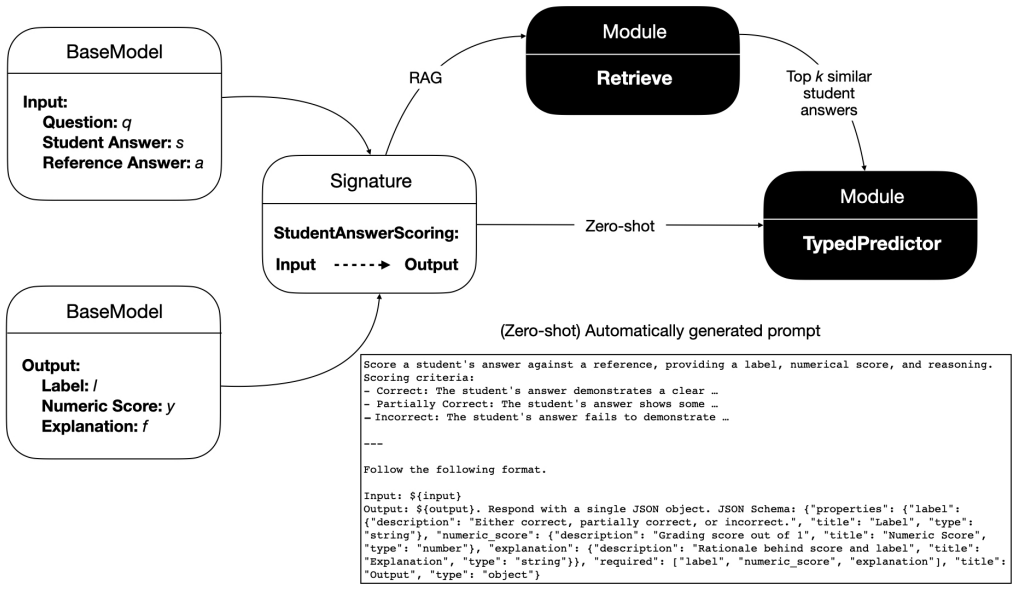


Figure 8.1: Overview of the implementation of the modular ASAS-F-Z and ASAS-F-RAG systems using DSPy

The modular zero-shot ASAS-F approach leverages the extensive domain knowledge embedded in pre-trained LLMs to score student answers and provide feedback without requiring any labeled training data. This approach takes advantage of the generalization capabilities of LLMs, which have been trained on vast amounts of text data and possess a broad understanding of language and context.

One of the significant challenges in implementing a zero-shot ASAS-F system is effective prompt engineering. Crafting prompts that elicit the desired responses from LLMs can be complex and often involves intricate string manipulation. This process is not only time-consuming but also prone to errors.

To address these challenges, we use DSPy [24], a framework designed to automate prompt generation and refinement to realize our ASAS-F-Z and ASAS-F-RAG systems as shown in Figure 8.1.

We first define our base inputs and outputs with their corresponding types and build a signature. A signature replaces a hand-written prompt by specifying what a function should do rather than how to do it. In its most basic form,

a signature consists of input and output fields, each with a type. To add more control, we add a basic description of what the signature does, i.e., score a student answer and generate feedback. We also add basic scoring criteria to the signature. Instead of manually crafting prompts with different prompt engineering techniques, we use predefined modules in DSPy such as the 'Predictor' module or the 'Chain-Of-Thought' module that can be easily replaced or modified to generate prompts. These modules allows us to automatically generate prompts by processing the input and output fields, generating instructions, and creating a template for the specified signature.

In the few-shot setting, which is discussed in Section 8.3.5, an additional 'Retrieve' module is added to retrieve examples from the training data. A code snippet that shows the basic implementation using DSPy to generate prompts for the zero-shot ASAS-F system is given in Appendix 8.7.

8.3.3 ASAS-F-Static: Few-Shot ASAS-F Using Random Static Examples

To further explore the impact of few-shot examples on ASAS-F performance, we introduce ASAS-F-Static, which uses 3 and 5 randomly selected static few-shot examples from the training data. Unlike ASAS-F-RAG, which dynamically retrieves the most similar examples for each student answer, ASAS-F-Static uses a fixed set of examples throughout the evaluation.

This approach allows us to assess whether the improvement in performance observed with ASAS-F-RAG is due to the dynamic retrieval of contextually relevant examples or simply the presence of additional examples in the prompt. By comparing ASAS-F-Static with ASAS-F-RAG, we can determine the effectiveness of using random static examples versus dynamically retrieved ones.

8.3.4 ASAS-F-Opt: Automatic Few-Shot Optimization with DSPy

In scenarios where labeled training data is available, optimizing both prompts and few-shot examples can significantly enhance the performance of ASAS systems. This section introduces ASAS-F-Opt, which leverages DSPy for the automatic optimization of prompts and examples within the ASAS-F framework. Unlike ASAS-F-Z, where DSPy was primarily used to establish a modular system and generate prompts, ASAS-F-Opt extends this functionality. It employs a systematic optimization approach to evaluate and refine combinations of prompts and examples based on defined performance metrics.

We start by defining a performance metric for model evaluation. Then, we construct a Bayesian surrogate model that samples and assesses various combinations of prompts and examples. Our approach uses the MIPROv2 prompt optimization algorithm [25], ensuring that the most effective prompts and examples are identified to enhance scoring and feedback generation.

For simplicity, we use the accuracy of the generated labels as our optimization metric. We use the Llama3:70b [23] as the prompt generation model, and the smaller Mistral:7b [125] functions as the task executor. The prompt proposal model generates initial prompts based on the source code and analyzes the dataset to produce example traces. The optimizer then systematically selects and refines these examples based on their performance according to our defined metric. A code snippet of the implementation can be found in Appendix 8.7.

8.3.5 ASAS-F-RAG: Few-Shot ASAS-F Using RAG

In this section, we first introduce a basic similarity-based majority-vote classifier using ColBERT for ASAS-F. We then extend this approach to incorporate RAG, which enhances the generative process by providing the LLM with contextually rich examples.

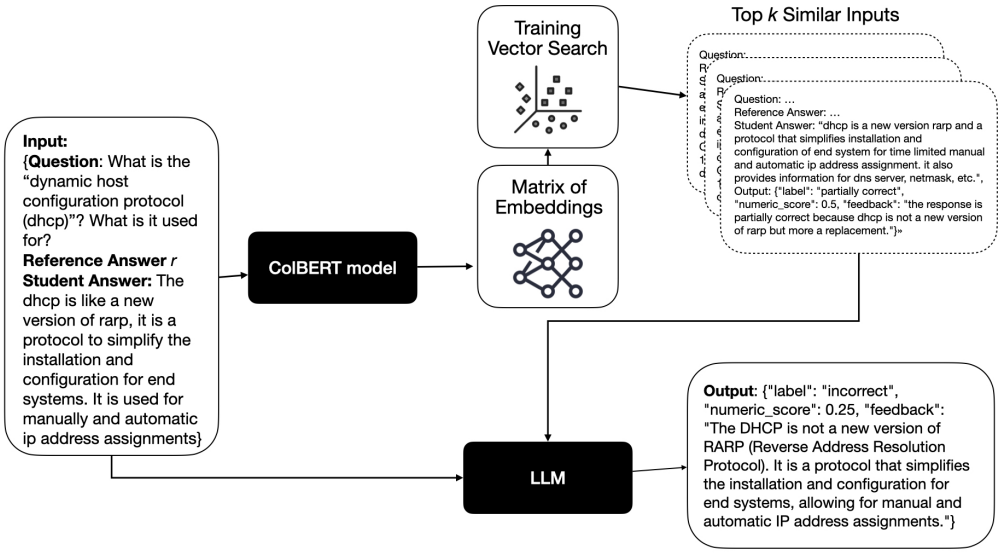


Figure 8.2: Overview of the ASAS-F system using LLMs and ColBERT-driven RAG.

Similarity-Based Majority-Vote with ColBERT

Many existing ASAS systems use similarity-based methods to assign scores to student answers [85]. These systems typically compare a student's response to reference answers or high-scoring examples, relying on similarity metrics to determine the score.

In our few-shot ASAS-F system, we build upon this approach by not limiting the comparison to a single reference answer. Instead, we retrieve the most similar examples from the training data to enhance the prompt. We hypothesize that this method will guide the LLM to produce feedback that more closely aligns with human evaluations, resulting in more accurate feedback.

To implement this, we utilize the ColBERT model [27], which efficiently retrieves contextually relevant examples from the training data. Unlike traditional models that encode the entire input into a single vector, ColBERT encodes the input into a matrix of contextual token-level embeddings. This approach allows us to capture fine-grained similarities between the student's answer and the training examples. Each word in the training example is represented by a BERT-based embedding, and these embeddings can be pre-stored for efficiency.

The relevance score between a student's answer s and an example d is calculated using a sum of maximum similarity (MaxSim) operators between the tokens in s and d. For each token in s, ColBERT identifies the most contextually similar token in d and sums these similarities to compute the overall relevance score. An example is considered more relevant if it has tokens that are highly contextually similar to those in the student's answer.

Formally, a student's answer s is tokenized, prepended with [CLS] and [Q] (query) tokens, and passed through BERT to produce vectors $[s_1, \ldots, s_N]$. The example d undergoes a similar process with [CLS] and [D] (document) tokens. A linear layer then adjusts the output size, and the student answer vectors and example vectors are normalized to unit length, yielding E_s and E_d, the final vector sequences. The ColBERT score is computed as:

$$score(s, d) = \sum_{i=1}^{N} \max_{j=1}^{m} \left(E_{s_i} \cdot E_{d_j} \right) \tag{8.2}$$

As a preliminary experiment, we assess the effectiveness of using the ColBERT retriever by employing a similarity-based majority-vote approach to classify student answers. This approach involves two key steps: retrieval and majority-vote classification. Initially, ColBERT encodes the student's answer and the training examples into embeddings and retrieves the top k most similar examples based on these embeddings. Then, for each student answer, the classifier aggregates the scores of the k retrieved examples and assigns the most frequent score as the final classification and the mean score as the final numeric score.

ASAS-F-RAG

Since we focus on generating feedback, we use LLMs in combination with ColBERT to retrieve the most similar examples from the training data and ground the feedback in real examples. We define this approach as ASAS-F-RAG. For each student answer, we retrieve the top $k = 3$ or $k = 5$ most similar examples from the training data. Each example consists of the question, the reference answer and the student answer. These examples are then fed into the LLM to generate feedback. Figure 8.2 shows an overview of the

ASAS-F-RAG system. The ColBERT retriever is used to retrieve the most similar examples, which are then passed to the LLM to generate feedback. The feedback is then used to assign a score to the student answer. A code snippet of the implementation of the system can be found in Appendix 8.7

8.4 Experimental Setup

8.4.1 Dataset

We evaluate our system on the Short Answer Feedback (SAF) Dataset [107]. The SAF dataset consists of short answer questions in communication network topics, each with a reference answer, given score, label and conent-focused elaborated feedback. There are no other datasets, based on our review, that provide such feedback for short answers. The dataset is split into training (70%), unseen answers (UA) (12%) and unseen questions (UQ) (18%). The test split of UA includes new answers to the existing training questions, while the UQ split introduces novel questions. However, it is important to note that in our approach, both splits are considered novel or unseen questions, as we do not fine-tune or train the model on the training split. Even in the few-shot setting, we use less than 0.5% of the training data as the provided examples.

8.4.2 Evaluation Metrics

Scoring Metrics

To evaluate the performance of our ASAS-F system in terms of scoring, we used two key metrics for the classification of the generated label: accuracy and macro-averaged F1 score. These metrics assess how well the system assigns correct labels (e.g., correct, incorrect, partially correct) to student answers. For the generated numeric score, we employed Root Mean Squared Error (RMSE) to measure the difference between the predicted scores and

the actual scores. These metrics provide a solid foundation for assessing the system's ability to deliver accurate and reliable scores.

Feedback Evaluation

Evaluating the quality of generated feedback is more complex due to its subjective nature and the need to verify accuracy. We approached this evaluation using both automated metrics and human assessments.

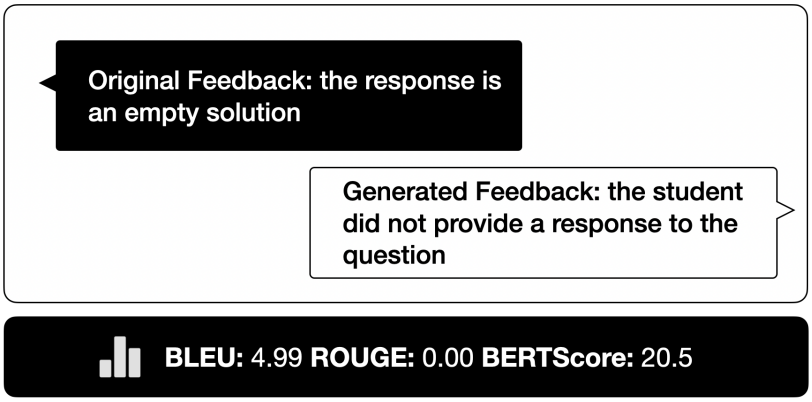


Figure 8.3: Example of feedback generated by the ASAS-F system compared to the reference feedback. Traditional metrics may not capture the nuances of feedback quality.

For automatic evaluation, we used traditional metrics such as Bilingual Evaluation Understudy Score (BLEU) (which measures n-gram overlap between generated and reference text), Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (which calculates the overlap of consecutive words focusing on recall), and the more recent BERTScore (which computes similarity using contextual embeddings). While these metrics provide a baseline for comparison, they fail to fully capture the nuances of feedback quality. For example, in Figure 8.3, we observe a case where the generated feedback may be semantically accurate but have low BLEU or ROUGE scores due to a lack of overlapping n-grams. To address these limitations, we focused on two critical human-evaluated aspects of feedback to ensure a comprehensive and reliable analysis.

- **Accuracy:** How well the facts in the feedback align with both the reference answer and the student's answer.

- **Clarity:** How specific, clear, and coherent the feedback is.

We conducted a human evaluation using five experience teachers from the Information Technology and Networks Field employed through the Prolific platform. The annotation was conducted on a total of 108 randomly chosen samples, which included six unseen answer samples for two questions and three unseen question samples for one question per model, across 12 different models. This evaluation allowed us to ensure that the feedback provided was not only factually correct but also clear and actionable for students. The raters spent an average of seven hours on this task and were compensated at the recommended rate of £9 per hour.

8.4.3 Model Selection

We employ various large language models, namely Mistral:7b, Mixtral:8x22b [125], Llama3:8b, and Llama3:70b [23] selected for their advanced performance and availability. We benchmark our ASAS-F system against several baselines: a majority class classifier, the T5 model [22] fine-tuned on the SAF dataset [107], and the Llama2:7b model fine-tuned on the SAF dataset [126]. The Llama2:7b finetune baseline is only available for the regression task, and has not been evaluated on the label classification task.

8.5 Results

8.5.1 Scoring Performance Analysis

Similarity-Based Majority-Vote with ColBERT

We evaluate the performance of the retrieval-based majority-vote classification approach on accuracy, F1 and RMSE. This serves as a preliminary experiment. Table 8.1 shows the performance of the ColBERT-based majority-vote classifier on the SAF dataset. The results indicate that the ColBERT-based majority-vote classifier outperforms the majority class baseline across all metrics showing the efficiency of using our refined approach.

Table 8.1: Performance of the similarity-based majority-vote classifier using ColBERT.

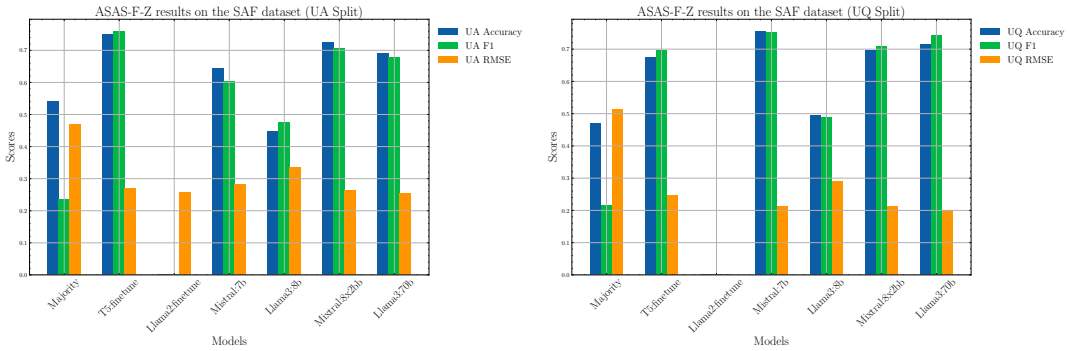
	UA Split			UQ Split		
	Acc	**F1**	**RMSE**	**Acc**	**F1**	**RMSE**
Majority	0.54	0.234	0.470	0.471	0.214	0.512
k = 3	0.694	0.674	0.266	0.602	0.646	0.277
k = 5	0.739	0.725	0.259	0.607	0.260	0.653

ASAS-F-Z

Table 8.2: ASAS-F-Z results on the SAF dataset. Bold values indicate the best performance overall for each metric.

	UA Split			UQ Split		
Model	**Accuracy**	**F1 Score**	**RMSE**	**Accuracy**	**F1 Score**	**RMSE**
Majority	0.540	0.234	0.470	0.471	0.214	0.512
T5: finetune	**0.750**	**0.759**	0.269	0.674	0.697	0.248
Llama2: finetune	-	-	0.257	-	-	-
Mistral: 7b	0.643	0.602	0.283	**0.755**	**0.751**	0.212
Llama3: 8b	0.448	0.476	0.335	0.495	0.489	0.290
Mixtral: 8x22b	0.726	0.706	0.263	0.695	0.708	0.213
Llama3: 70b	0.691	0.678	**0.253**	0.716	0.742	**0.199**

Table 8.2 and Figures 8.4(a) and 8.4(b) show the results of the zero-shot ASAS-F system on both test splits of the SAF dataset compared to the baselines. In the UA split, the fine-tuned baselines outperform the zero-shot system in terms of accuracy and F1 score. However, in the UQ split, excluding the Llama3:8b model, the ASAS-F-Z system is able to outperform all baselines in all metrics. The highest performing model is the smallest Mistral:7b model, which achieves an accuracy of 0.755 and an F1 score of 0.751 in the UQ split. Llama3:70b achieves the lowest RMSE. This suggests that while finetuning can perform similarly to human raters in answers that it has seen during training, it tends to underperform when evaluating on new questions. The zero-shot ASAS-F system, on the other hand, is able to generalize better to new questions, indicating the potential of this approach for educational applications.



(a) Unseen Answers **(b)** Unseen Questions

Figure 8.4: Performance of the ASAS-F-Z system on the SAF dataset. Higher is better for accuracy and F1 score, lower is better for RMSE.

ASAS-F-Static

To evaluate the impact of adding few-shot examples without dynamic retrieval, we introduced ASAS-F-Static, which utilizes randomly selected static examples from the training data. Table 8.3 presents the results of this approach.

When comparing ASAS-F-Static with the zero-shot ASAS-F-Z system across various models, we found that ASAS-F-Static generally did not outperform ASAS-F-Z in both the unseen answers (UA) and unseen questions (UQ) splits. Across models like Mistral:7B, Mixtral:8x22B, and Llama3:70B, the zero-shot approach often matched or exceeded the performance of the static few-shot method. While ASAS-F-Static showed some improvement with the Llama 3:8B model—where adding static examples led to higher accuracy in the UA split—these gains were inconsistent and not observed across other models or splits. This indicates that simply adding randomly selected static examples does not significantly enhance model performance and may sometimes impede it. The findings highlight the crucial role of example relevance over quantity in few-shot learning, suggesting that without contextually relevant examples, models may not benefit from additional data and could potentially be misled by noisy irrelevant information.

Table 8.3: ASAS-F-Static and ASAS-F-Opt results on the SAF dataset. Bold values indicate the best performance overall for each metric.

	UA Split			UQ Split		
Model	**Accuracy**	**F1 Score**	**RMSE**	**Accuracy**	**F1 Score**	**RMSE**
Majority	0.540	0.234	0.470	0.471	0.214	0.512
T5: finetune	**0.750**	**0.759**	0.269	0.674	**0.697**	0.248
Llama2: finetune	-	-	0.257	-	-	-
k = 3, Mistral: 7b (static)	0.575	0.425	0.394	0.573	0.520	0.383
k = 5, Mistral: 7b (static)	0.615	0.478	0.370	0.576	0.509	0.404
k = 3, Llama3: 8b (static)	0.619	0.615	0.353	0.510	0.546	0.289
k = 5, Llama3: 8b (static)	0.635	0.642	0.330	0.630	0.662	0.282
k = 3, Mixtral: 8x22b (static)	0.635	0.607	0.299	0.672	0.689	**0.198**
k = 5, Mixtral: 8x22b (static)	0.671	0.630	0.264	**0.695**	0.675	0.244
k = 3, Llama3: 70b (static)	0.663	0.682	0.255	0.641	0.682	0.225
k = 5, Llama3: 70b (static)	0.643	0.654	0.285	0.622	0.667	0.246
k = 3, Mistral: 7b (optimized)	0.639	0.561	0.305	0.646	0.672	0.317
k = 3, Llama3: 8b (optimized)	0.671	0.590	0.305	0.654	0.686	0.245
k = 3, Mixtral: 8x22b (optimized)	0.694	0.667	0.321	0.607	0.465	0.312
k = 3, Llama3: 70b (optimized)	0.679	0.719	**0.245**	0.539	0.589	0.217

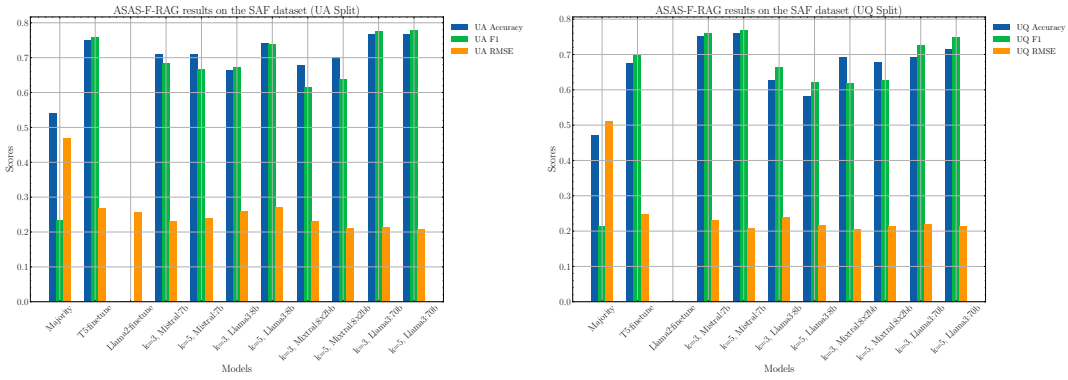
ASAS-F-Opt

In ASAS-F-Opt, we enhance performance by optimizing the selection of few-shot examples. Unlike static selection or dynamic retrieval, this approach refines examples to maximize their effectiveness for specific metrics.

The results in Table 8.3 show promising improvements in the UA split, particularly with Llama3:70b achieving the best overall accuracy (0.679), F1 score (0.719), and RMSE (0.245).

However, for the same model, in the UQ split, the static selection of few-shot examples outperformed. The smaller models demonstrated gains with the optimized approach in the UQ split, but with the larger models, performance varied across splits, suggesting that optimization may not generalize consistently.

These results highlight the potential of optimized few-shot selection but also underscore the need for more robust strategies to ensure consistent improvements across different models and splits. The complexity of optimization strategies and the need for nuanced approaches to align with ASAS-F requirements are evident, emphasizing the importance of further research in this area.



(a) Unseen Answers **(b)** Unseen Questions

Figure 8.5: Performance of the ASAS-F-RAG system on the SAF dataset. Higher is better for accuracy and F1 score, lower is better for RMSE.

ASAS-F-RAG

Table 8.4: ASAS-F-RAG results on the SAF dataset. Bold values indicate the best performance overall for each metric.

	UA Split			UQ Split		
Model	**Accuracy**	**F1 Score**	**RMSE**	**Accuracy**	**F1 Score**	**RMSE**
Majority	0.540	0.234	0.470	0.471	0.214	0.512
T5: finetune	0.750	0.759	0.269	0.674	0.697	0.248
Llama2: finetune	-	-	0.257	-	-	-
k = 3, Mistral: 7b	0.710	0.684	0.232	0.753	0.761	0.230
k = 5, Mistral: 7b	0.710	0.667	0.240	**0.760**	**0.768**	0.209
k = 3, Llama3: 8b	0.663	0.672	0.259	0.628	0.664	0.238
k = 5, Llama3: 8b	0.742	0.739	0.271	0.581	0.621	0.217
k = 3, Mixtral: 8x22b	0.679	0.616	0.230	0.693	0.619	**0.204**
k = 5, Mixtral: 8x22b	0.702	0.638	0.212	0.677	0.626	0.215
k = 3, Llama3: 70b	**0.766**	0.775	0.213	0.693	0.726	0.220
k = 5, Llama3: 70b	**0.766**	**0.778**	**0.208**	0.714	0.749	0.213

To address the limitations shown by the ASAS-F-Z system especially in the UA split and the low performance shown by ASAS-F-Opt, we propose the ASAS-F-RAG system. This system automatically retrieves the most similar answers from existing data to augment the prompts used to improve model performance. Table 8.4 shows the results of the ASAS-F-RAG system on both test splits of the SAF dataset compared to the baselines. Figures 8.5(a) and 8.5(b) illustrate the performance of the ASAS-F-RAG system on the UA and UA splits, respectively.

In the UA split, the results highlight that the ASAS-F-RAG system consistently outperforms both the ASAS-F-Z and ASAS-F-Static systems across various models. For instance, the Llama3:70b model with five examples achieved the highest performance across all metrics, demonstrating superior accuracy, F1 score, and RMSE compared to other models and approaches. This indicates that larger models with more examples can leverage their extensive training to provide more accurate predictions in the few-shot setting. The other models also exhibited a similar trend, with ASAS-F-RAG outperforming ASAS-F-Static, which suggests that the relevance of retrieved examples is crucial for enhancing model performance.

Conversely, in the UQ split, the performance improvements with additional examples are less consistent. Notably, the Llama3:8b and Mixtral:8x22b models did not follow the general trend of improved performance with more examples. The Llama3:70b model, while still competitive, did not maintain its leading position as it did in the UA split. Here, the zero-shot Llama3:70b model actually outperformed the few-shot models, suggesting that the examples used in the few-shot setting might not be as effective when dealing with questions that differ significantly.

The discrepancies in performance between the two splits can be attributed to the nature of the retrieved examples. In the unseen questions split, the examples retrieved were from different questions, which may have introduced variability that impacted the model's ability to generalize effectively. This variability might explain why the performance with more examples did not consistently improve and why the zero-shot model performed comparably or better in some cases.

These findings highlight the importance of example relevance in few-shot learning. The ASAS-F-RAG system's ability to dynamically retrieve contextually similar examples allows it to provide more relevant information to the model, enhancing performance over both the zero-shot and static few-shot approaches. In contrast, ASAS-F-Static's use of random examples may not provide the necessary context for the model to improve its predictions. This suggests that the effectiveness of few-shot examples depends not only on their presence but also on their contextual relevance to the task at hand.

Table 8.5: Statistical feedback quality evaluation on the SAF dataset. Bold values indicate the best performance overall for each metric. Underline values indicate the second-best performance.

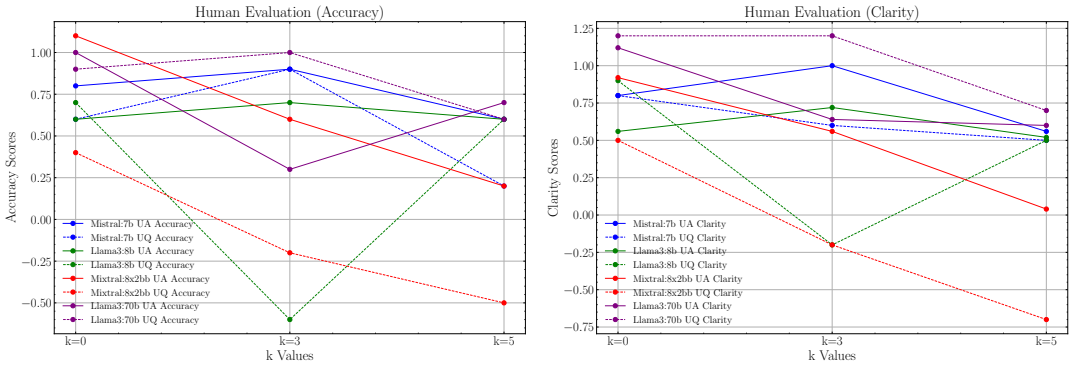
	UA Split			UQ Split		
Model	**BLEU**	**ROUGE**	**BERTScore**	**BLEU**	**ROUGE**	**BERTScore**
Majority	2.2	20.2	42.2	0.2	11.5	38.1
T5: finetune	33.7	**52.8**	**65.0**	**10.7**	**31.1**	**52.2**
Llama2: finetune	**39.6**	13.7	-	-	-	-
k = 0, Mistral: 7b	2.42	5.11	23.6	2.15	6.93	25.9
k = 3, Mistral: 7b	7.90	23.7	40.2	6.38	<u>28.8</u>	<u>43.9</u>
k = 5, Mistral: 7b	6.75	21.7	37.7	3.82	19.2	33.6
k = 0, Llama3: 8b	1.76	3.97	17.5	1.45	4.06	18.6
k = 3, Llama3: 8b	9.86	28.5	43.3	4.97	24.7	39.3
k = 5, Llama3: 8b	11.9	32.5	45.6	5.36	24.7	39.7
k = 0, Mixtral: 8x22b	2.42	26.7	25.6	2.72	6.20	28.0
k = 3, Mixtral: 8x22b	13.4	29.2	42.4	<u>8.73</u>	23.6	35.0
k = 5, Mixtral: 8x22b	11.6	26.7	41.0	3.78	17.6	30.4
k = 0, Llama3: 70b	1.81	4.11	20.5	2.10	5.35	21.6
k = 3, Llama3: 70b	13.2	34.7	49.5	3.67	19.3	35.7
k = 5, Llama3: 70b	<u>15.0</u>	<u>36.3</u>	<u>50.6</u>	3.86	20.1	36.0

8.5.2 Feedback Quality Analysis

Statistical Analysis

Table 8.5 presents the evaluation of feedback quality on the SAF dataset using SacreBLEU [127], ROUGE-2 [128], and BERTScore [129] metrics for the ASAS-F-Z and ASAS-F-RAG models. It is evident that the baseline models were able to outperform both the zero-shot and few-shot ASAS-F systems in all metrics. However, the baseline models due to finetuning have been reported to often copy common phrases from the training data. Looking at the second-best performing models, we can see that the ASAS-F-RAG outperforms the ASAS-F-Z in both splits. This suggests that incorporating a small number of labeled examples can improve the quality of the generated feedback in terms of consistency with the reference feedback. However, increasing the number of examples beyond a certain threshold does not necessarily lead to higher similarity to the reference feedback. This can be seen in the UQ split where the second-best performing models used 3 examples.

Human Evaluation



(a) Accuracy **(b)** Clarity

Figure 8.6: Human feedback evaluation results on samples with existing student answers. Higher is better.

To ensure the quality of the feedback generated by our system, we conducted a human evaluation with five expert teachers in Information Technology and Networks. The raters assessed 108 randomly selected samples from 12 models, focusing on accuracy and clarity. Feedback was rated on a 5-point Likert scale ranging from -2 to 2, where -2 indicated strongly inaccurate or unclear feedback, 0 indicated neutral feedback, and 2 indicated strongly accurate or clear feedback. Figures 8.6(a) and 8.6(b) illustrate the human evaluation results for samples with existing student answers. Table 8.6 presents the overall results, excluding no-response samples.

For the baseline models, human evaluation was not conducted. Table 8.7 provides a sample of the feedback generated by the baseline model and samples from our ASAS-F system. The feedback generated by the baseline model is repetitive and lacks the depth needed for comprehensive understanding. A high degree of similarity could artificially inflate the statistical evaluation metrics like BLEU scores. The high overlap between training inputs and generated outputs signals that the model may not truly be learning to adapt to novel inputs but is instead reproducing familiar patterns. Thus, while the baseline system might achieve high scores on automated metrics due to repetition of training data, its semantic contribution may be minimal compared to the more informative and contextually appropriate feedback generated by our system. More examples of comparisons between the feedback generated

Table 8.6: Human feedback quality evaluation results on the samples with existing student answers from the SAF dataset. Bold values indicate the best performance overall for each metric while underline values indicate the second-best performance.

	UA Split		UQ Split	
	Accuracy	**Clarity**	**Accuracy**	**Clarity**
k = 0, Mistral:7b	0.84	0.80	0.60	0.80
k = 3, Mistral:7b	0.92	<u>1.00</u>	<u>0.90</u>	0.60
k = 5, Mistral:7b	0.60	0.56	0.20	0.50
Average	0.79	0.79	0.57	0.63
k = 0, Llama3:8b	0.64	0.56	0.70	<u>0.90</u>
k = 3, Llama3:8b	0.68	0.72	-0.60	-0.20
k = 5, Llama3:8b	0.40	0.52	0.60	0.50
Average	0.57	0.60	0.23	0.40
k = 0, Mixtral:8x22b	**1.12**	0.92	0.40	0.50
k = 3, Mixtral:8x22b	0.56	0.56	-0.20	-0.20
k = 5, Mixtral:8x22b	0.16	0.04	-0.50	-0.70
Average	0.61	0.51	-0.10	-0.13
k = 0, Llama3:70b	<u>1.04</u>	**1.12**	<u>0.90</u>	**1.20**
k = 3, Llama3:70b	0.32	0.64	**1.00**	**1.20**
k = 5, Llama3:70b	0.68	0.60	0.60	0.70
Average	0.68	0.79	0.83	1.03
k = 0, Average	0.91	0.85	0.65	0.85
k = 3, Average	0.62	0.73	0.28	0.35
k = 5, Average	0.46	0.43	0.23	0.25
Overall Average	0.66	0.67	0.35	0.48

by the baseline models and our ASAS-F system can be found in Appendix 8.8.

In the analysis, we find that inconsistencies across different models occasionally confused raters, leading to lower accuracy scores. Notably, for no-response answers, which comprised 22% of all samples, the employed LLMs were generally able to generate feedback that correctly identified the lack of an answer. In 23 out of 24 cases, the feedback was considered strongly accurate. In the remaining case, the feedback failed to explicitly state the absence of an answer. However, disagreement among the raters was observed in all cases regarding the accuracy and clarity. Despite this,

the inter-rater agreement for these no-response samples was relatively high, with a Krippendorff's alpha of 0.77 for accuracy and 0.71 for clarity. [2]

In contrast, for the remaining 78% of samples—where student answers were present—Krippendorff's alpha dropped significantly to 0.19 for accuracy and 0.20 for clarity, indicating low agreement. This discrepancy may stem from the granularity of the 5-point scale and the inherent subjectivity of evaluating nuanced feedback.

Given the confusion that occured with the no-responses, we focus on the samples with existing student answers for the human evaluation. Among the models evaluated, the Llama3:70b model achieved the highest average accuracy and clarity scores in both the UA and UQ splits. The Mistral:7b model followed closely, showing competitive performance. However, a general trend of decreasing accuracy and clarity with an increasing number of examples was observed across all models.

As a post hoc analysis, we merged the strongly accurate and accurate categories and the strongly inaccurate and inaccurate categories to create a 3-point scale. This adjustment led to an increase in the Krippendorff's alpha to 0.91 for accuracy and 0.89 for clarity for the no-response answers. For the remaining answers, the Krippendorff's alpha increased to 0.60 for accuracy and 0.64 for clarity, indicating a moderate level of agreement among raters. This suggests that the original scale was too granular and that a simpler scale may have resulted in higher agreement scores.

In the few-shot approach, LLMs often mimicked the style of the provided reference answers or examples as shown in the first example in Figure 8.7. Many raters commented the feedback that was too brief, lacking sufficient context or explanation. Although the feedback was factually correct, it did not provide the depth needed for comprehensive understanding, which could be critical in educational settings. Conversely, in the zero-shot approach, the generated feedback was often longer and appeared more detailed. However, as the second example in Figure 8.7 demonstrates, while some of this feedback was labeled as accurate, it was actually incorrect when compared to the reference answers. This highlights the challenge of calibrating LLMs, as they

[2] Krippendorff's alpha is a statistical measure of agreement among raters, accounting for the possibility of chance agreement. It ranges from 0 to 1, with values closer to 1 indicating strong agreement.

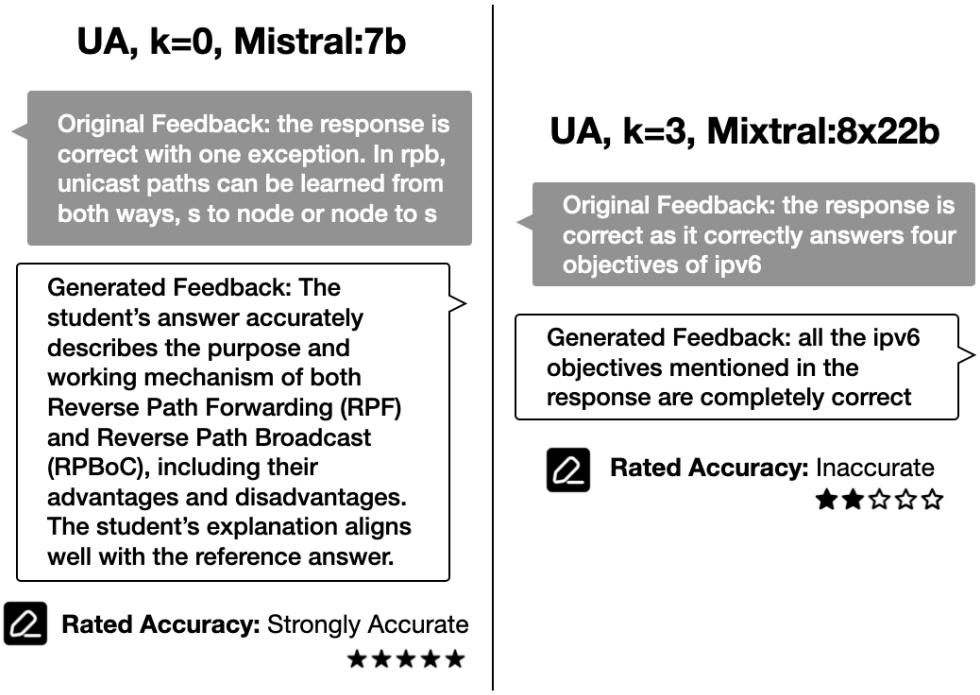


Figure 8.7: Samples of human evaluation accuracy ratings for generated feedback vs actual feedback.

may present incorrect feedback with high confidence [130], making it difficult to detect when they are hallucinating [131], especially when evaluating the feedback in isolation.

8.6 Discussion

In this study, we aimed to answer key research questions regarding the performance of an ASAS-F system in zero-shot and few-shot settings. Rather than relying on intricate and labor-intensive prompt engineering to optimize performance for specific datasets, we adopted a modular design for our system using DSPy. We explored two main methods for selecting few-shot examples ASAS-F-RAG and ASAS-F-Opt in comparison with the simpler ASAS-F-Z and fine-tuned baselines. We also evaluated an approach using randomly selected static few-shot examples (ASAS-F-Static) to assess the impact of example relevance. Our evaluation focused on the system's ability to score

student answers that align with the reference answers and generate feedback that is accurate and clear.

To address **RQ1** (How does the performance of our modular ASAS-F system compare to state-of-the-art models in automatic short answer scoring?), our findings revealed that LLMs can compete with fine-tuned baselines in scoring accuracy. While fine-tuning on unseen answers showed better performance compared to the zero-shot approach, three out of four LLMs surpassed the fine-tuned baselines when tested on unseen questions, with the smallest model, Mistral:7b, delivering the best results. Our few-shot approach consistently outperformed fine-tuned baselines in both evaluation splits, underscoring the effectiveness of RAG for improving scoring performance in ASAS tasks. Additionally, we compared our RAG-based method with an approach using randomly selected static few-shot examples (ASAS-F-Static). The results demonstrated that the dynamic retrieval of contextually relevant examples in the RAG-based method significantly outperforms the static few-shot approach, highlighting the importance of example relevance in few-shot learning scenarios. Our modular system and the LLMs generally performed well in terms of the evaluation metrics. However, when enforcing type constraints on the outputs, some LLMs failed to abide by them. For these output errors, we implemented a fallback mechanism to ensure comprehensive evaluation (see Appendix 8.9).

For **RQ2** (When labeled training data is available, how can we optimize prompts and few-shot examples to improve our ASAS-F performance in an efficient way?), we experimented with a Bayesian optimization approach to automatically optimize the selection of few-shot examples and prompts. However, we found that our dynamic few-shot approach using RAG significantly outperformed the automatic optimization method. In tasks like ASAS, where the subjectivity of scoring plays a critical role, dynamic selection of similarly scored examples proves more effective than fully automated methods. This indicates that aligning retrieved examples closely with the scoring task's subjectivity is crucial for optimizing ASAS performance. Moreover, using randomly selected static few-shot examples (ASAS-F-Static) did not yield significant performance improvements over the zero-shot approach, emphasizing the importance of relevant examples.

Regarding **RQ3** (How accurate and clear is the feedback generated by our ASAS-F system?), traditional evaluation metrics like BLEU and ROUGE are often insufficient, as they do not account for the nuanced nature of educational feedback. These metrics can fail to capture the quality of feedback when generated sentences are factually correct but differ significantly from the reference text. In our human evaluation, we observed considerable disagreement among raters on the accuracy and clarity of the feedback generated by the LLMs, reflecting the inherent subjectivity of the task. Although the few-shot approach generated feedback more aligned with reference feedback, its accuracy ratings were lower than those from the zero-shot approach. This highlights the challenges in evaluating feedback quality, particularly in subjective tasks. Future work should explore more robust evaluation frameworks for feedback generation, focusing on both subjective and objective measures.

8.7 DSPy Code Snippets

In this section, we provide code snippets of the DSPy implementation of the ASAS-F system. The code snippets for ASAS-F-Z, ASAS-F-Opt and ASAS-F-RAG are shown in Listings 8.1, 8.2, and 8.3, respectively.

```
class Input(BaseModel):
    question: str = Field(
        description="The question posed to the student"
    )
    student_answer: str = Field(
        description="The student's written answer"
    )
    reference_answer: str = Field(
        description="The reference material for the question"
    )

class Output(BaseModel):
    label: str = Field(
        description="Either correct, partially correct, or incorrect."
    )
    numeric_score: float = Field(
        description="Grading score out of 1"
    )
```

```
    explanation: str = Field(
        description="Rationale behind score and label"
    )

class StudentAnswerScoring(dspy.Signature):
    """Score a student's answer against a reference, providing a label,
    numerical score, and reasoning.

    Scoring criteria:
    - Correct: The student's answer demonstrates a clear understanding
      of the core concept, with key points accurately addressed.
      Minor errors are acceptable.
    - Partially Correct: The student's answer shows some understanding
      of the core concept but misses key points or contains notable
      inaccuracies.
    - Incorrect: The student's answer fails to demonstrate an
      understanding of the core concept or is largely inaccurate.
    """
    input: Input = dspy.InputField()
    output: Output = dspy.OutputField()

scorer = dspy.TypedPredictor(StudentAnswerScoring)
```

Listing 8.1: DSPy code snippet of the ASAS-F-Z system.

```
from pydantic import BaseModel, Field
import dspy

class Answer(BaseModel):
    question: str = Field(
        description="The question posed to the student"
    )
    student_answer: str = Field(
        description="The student's written answer"
    )
    reference_answer: str = Field(
        description="The reference material for the question"
    )

class Feedback(BaseModel):
    label: str = Field(
        description="Either correct, partially correct, or incorrect."
    )
    numeric_score: float = Field(
        description="Grading score out of 1"
```

```
    )
    explanation: str = Field(
        description="Rationale behind score and label"
    )

class StudentAnswerScoring(dspy.Signature):
    """Score a student's answer against a reference, providing a label,
    numerical score, and reasoning.

    Scoring criteria:
    - Correct: The student's answer demonstrates a clear understanding
      of the core concept, with key points accurately addressed.
      Minor errors are acceptable.
    - Partially Correct: The student's answer shows some understanding
      of the core concept but misses key points or contains notable
      inaccuracies.
    - Incorrect: The student's answer fails to demonstrate an
      understanding of the core concept or is largely inaccurate.
    """
    answer: Answer = dspy.InputField()
    feedback: Feedback = dspy.OutputField()

scorer = dspy.TypedPredictor(
    StudentAnswerScoring,
    max_retries=8
)

from dspy.teleprompt import MIPROv2

teleprompter = MIPROv2(
    prompt_model=llm,
    task_model=task_llm,
    metric=validate_label,
    num_candidates=3,
    init_temperature=0.0,
    verbose=True
)

kwargs = dict(
    display_progress=True,
    display_table=0,
    num_threads=16
)

```

```
compiled_program = teleprompter.compile(
    scorer,
    trainset=trainset,
    valset=valset,
    num_batches=20,
    max_bootstrapped_demos=1,
    max_labeled_demos=3,
    eval_kwargs=kwargs
)
```

Listing 8.2: Code snippet of the ASAS-F-Opt system.

```

from dsp import passages2text
import random

NUM_EXAMPLES = 5

class Input(BaseModel):
    question: str = Field(
        description="The question posed to the student"
    )
    reference_answer: str = Field(
        description="The reference material for the question"
    )
    student_answer: str = Field(
        description="The student's written answer"
    )

class Output(BaseModel):
    label: str = Field(
        description="Either correct, partially correct, or incorrect."
    )
    numeric_score: float = Field(
        description="Grading score out of 1"
    )
    feedback: str = Field(
        description="Rationale behind score and label"
    )

class FeedbackGeneratorRAG(dspy.FunctionalModule):
    def __init__(self, num_passages=NUM_EXAMPLES):
        super().__init__()
        self.retrieve = dspy.Retrieve(k=num_passages)

```

```
    @dspy.predictor
    def output(self, examples, input: Input) -> Output:
        """Score a student's answer against a reference, providing a
            label,
        numerical score, and feedback.

        Scoring criteria:
        - Correct: The student's answer demonstrates a clear understanding
          of the core concept, with key points accurately addressed.
          Minor errors are acceptable.
        - Partially Correct: The student's answer shows some understanding
          of the core concept but misses key points or contains notable
          inaccuracies.
        - Incorrect: The student's answer fails to demonstrate an
          understanding of the core concept or is largely inaccurate.
        """
        pass

    def forward(self, question, reference_answer, student_answer):
        retrieved = self.retrieve(student_answer).passages
        similar_scored_examples = passages2text(retrieved)

        input_data = Input(
            question=question,
            reference_answer=reference_answer,
            student_answer=student_answer
        )

        scoring = self.output(
            examples=similar_scored_examples,
            input=input_data
        )

        return dspy.Prediction(
            label=scoring.label,
            numeric_score=scoring.numeric_score,
            feedback=scoring.feedback
        )
```

Listing 8.3: DSPy code snippet of the ASAS-F-RAG system.

8.8 Output Generation Examples

This section provides a comparison of feedback samples generated by the baseline models and our system. Tables 8.8, 8.9, and 8.10 present feedback generated by the baseline models and samples from our system for the same answers. These samples show clear instances of phrase reuse from the fine-tuning process from the baseline, highlighting limited generalization capabilities compared to the more informative and contextually appropriate feedback generated by our system.

8.9 Typed Predictor Error Analysis

Our ASAS-F system uses DSPy's typed predictors, which enforce input/output type constraints to ensure structured predictions. During evaluation, we encountered a 4.13% average error rate across models due to formatting issues that prevented output generation. To mitigate this, we implemented a fallback mechanism using an untyped predictor when the typed predictor failed. Table 8.11 shows the error rates by model. The Llama models performed better at maintaining type constraints, with Llama3:8b achieving the lowest error rate (2.04%) and Llama3:70b at 3.0%. In comparison, Mistral:7b and Mixtral:8x22b had higher error rates at 4.16% and 7.32% respectively.

8.10 Summary

This study provides a comprehensive evaluation of an ASAS-F system in both zero-shot and few-shot settings. We introduced a novel RAG-based few-shot selection method with ColBERT which achieved significant improvements in scoring accuracy, surpassing fine-tuned baselines, particularly on unseen questions. These results show the robustness and adaptability of our approach for ASAS, reducing the reliance on computationally expensive fine-tuning. Additionally, we compared our RAG-based method with an approach using randomly selected static few-shot examples (ASAS-F-Static). The results

demonstrated that the dynamic retrieval of contextually relevant examples in the RAG-based method significantly outperforms the static few-shot approach. This highlights the importance of example relevance in few-shot learning scenarios.

While our scoring method showed strong scoring performance, the evaluation of generated feedback highlighted challenges. Statistical metrics such as BLEU and ROUGE were lower for our approach compared to fine-tuned baselines, primarily due to the lack of overlapping n-grams between the reference and generated feedback, even when the feedback was factually accurate. This highlights the limitations of traditional metrics in assessing feedback quality. Human evaluation revealed moderate agreement among raters, reflecting the inherent subjectivity of feedback assessment. Additionally, we observed an inverse relationship between the number of few-shot examples and the quality of feedback, suggesting that more examples may not always improve feedback quality. This finding also points to difficulties in calibrating LLMs and detecting hallucinations, particularly given the subjective nature of educational feedback.

Despite these challenges, our approach shows significant potential for improving the accuracy and efficiency of ASAS systems. Future work will focus on developing more reliable evaluation methods for feedback quality, exploring model calibration techniques to mitigate hallucinations, and refining prompt design strategies to enhance performance.

Table 8.7: Comparison of feedback generated by the baseline models and samples from the ASAS-F system. Underlined text indicates repetitions or redundancies.

Description	Feedback
Student Answer 1	The extension headers are placed between fixed header and payload. The main advantage is that they are optional. This means that the use of space is more efficient because there won't be any redundant information placed there in order to keep the expected length.
Gold Feedback	The response answers only the location of extension headers correctly. The definition is missing and the advantage is incorrect when compared to IPv4 as the option field in IPv4 is also optional. Thus, there is no efficiency benefit.
Finetuned T5	The advantage is incorrect as the option field in the IPv4 header is optional and can be placed between payload and fixed header. The advantage is incorrect as the option field in the IPv4 header is optional and can be placed between payload and fixed header.
k = 5, Llama3:70b	The response correctly states the location of extension headers between the fixed header and payload. However, the main advantage stated is not entirely accurate. While it's true that IPv6 extension headers are optional, this doesn't directly contribute to more efficient use of space. A correct advantage would be that they allow for adding new options without changing the fixed header size or that IPv6 packets with optional headers are typically processed faster/simpler by intermediate devices.
k = 3, Mistral:7b	The response correctly identifies the location of extension headers and mentions that they are optional. However, it does not fully address the main advantage as stated in the reference answer, which includes both the ability to append new options without changing the header and the simplified processing by intermediate devices.

Table 8.8: Comparison of feedback generated by the baseline models and samples from the ASAS-F system for Student Answer 2. Underlined text indicates repetitions or redundancies.

Description	Feedback
Student Answer 2	1. IPv6 extension headers contain supplementary information used by network devices (such as routers, switches, and endpoint hosts) to decide how to direct or process an IPv6 packet. 2. They are located between the fixed header and payload. 3. IPv6 extension headers can be of arbitrary length. The number of options that a packet carries is not limited.
Gold Feedback	The response answers the description, location of extension headers, and advantage correctly.
Finetuned T5	The advantage is partially correct as it does not state what extension headers are, how they are used in IPv4, and how they are used in IPv6. The advantage is incorrect as the option field in the IPv4 header is optional and can be of arbitrary length.
k = 0, Llama3:8b	The student's answer demonstrates a good understanding of the concept of IPv6 extension headers, correctly stating that they contain supplementary information and are located between the fixed header and payload. However, their explanation is not entirely accurate, as they mention arbitrary length without specifying the limitations on the number of options carried by a packet. Additionally, while they touch upon the main advantage of extension headers compared to IPv4, their answer lacks clarity and does not fully address the reference's correct points.
k = 5, Mixtral8:22xb	The response answers the description and location of the extension headers correctly but does not mention any advantage compared to IPv4.

Table 8.9: Comparison of feedback generated by the baseline and samples from the ASAS-F system for Student Answer 3.

Description	Feedback
Student Answer 3	Spanning trees are appealing to broad- and multicasting scenarios because they allow the packets to only travel one path (except traveling backwards). This removes the need for looking up specific tables as in RPF/RPB.
Gold Feedback	It is true that there is a unique path between nodes, but that not only does away with the need to look at routing tables in RPF/RPB, but reduces duplicates by removing loops (unnecessary links). No explanation was provided for modifying the link-state algorithm to construct a multicast spanning tree for nodes.
Finetuned T5	The explanation behind using a spanning tree for multicast and broadcast is partially correct as it does not mention how the link-state algorithm is modified to construct a multicast spanning tree. The explanation for modifying the link-state algorithm to construct a multicast spanning tree is partially correct as it does not mention how each node shares its multicast information with others by adding it to the link-state packet.
k = 5, Llama3:70b	The response correctly answers why a spanning-tree usage is ideal in multicast and broadcast scenarios, mentioning that it removes the need for looking up specific tables as in RPF/RPB. However, the description lacks detail on how link-state routing can be modified to construct a spanning tree for multicasting.
k = 0, Mistral:7b	The student's answer correctly identifies the property of spanning trees that makes them appealing for broad- and multicasting (single unique path between every pair of nodes). The student also provides a reasonable explanation on how to modify link state routing to construct a spanning tree for multicasting, although the reference answer provides more detailed information about the process.

Table 8.10: Comparison of feedback generated by the baseline and samples from the ASAS-F system for Student Answer 4.

Description	Feedback
Student Answer 4	PROPERTY: The spanning tree does not have any loops.
Gold Feedback	The response correctly answers why a spanning-tree usage is ideal in multicast and broadcast. However, the explanation for modifying the link-state algorithm to construct a multicast spanning tree for nodes is not provided.
Finetuned T5	The response correctly identifies the spanning-tree property that makes it appealing for broadcast and multicast and explains how the link-state algorithm is modified to construct a multicast spanning tree. However, the explanation for modifying the link-state algorithm is partially correct as it does not state how this modification is achieved.
k = 0, Llama3:8b	The student correctly identifies that spanning trees do not have loops, which is a key property. However, their explanation is incomplete and does not fully capture the essence of why spanning trees are appealing for multicasting. The reference answer provides a more comprehensive explanation, including the concept of unique paths between every pair of nodes in the tree. Additionally, the student's answer does not address how to modify link-state routing to construct a spanning tree for multicasting, which is an important aspect of the question.
k = 5, Mixtral8:22xb	The response only states one property of a spanning tree but fails to explain how link-state routing can be modified for constructing a multicast spanning tree.

Table 8.11: Error rates (%) in typed predictors across the LLMs.

Model	UA Split	UQ Split
k = 0, Mistral:7b	5.16	2.86
k = 3, Mistral:7b	3.57	2.86
k = 5, Mistral:7b	6.35	4.17
Average	5.03	3.30
Overall Average	**4.16**	
k = 0, Llama3:8b	3.97	1.82
k = 3, Llama3:8b	2.78	0.26
k = 5, Llama3:8b	1.59	1.83
Average	2.78	1.30
Overall Average	**2.04**	
k = 0, Mixtral:8x22b	8.33	6.25
k = 3, Mixtral:8x22b	3.57	11.20
k = 5, Mixtral:8x22b	5.95	8.59
Average	5.95	8.68
Overall Average	**7.32**	
k = 0, Llama3:70b	1.98	0.00
k = 3, Llama3:70b	3.97	3.39
k = 5, Llama3:70b	3.17	5.47
Average	3.04	2.95
Overall Average	**3.0**	

9

Conclusion and Future Work

In the first part of this thesis, we presented the TOPP framework, which leverages the rich qualitative insights contained in teacher observation reports. To answer the question **TQ1: How can representation learning techniques effectively integrate qualitative teacher observations with traditional academic metrics to improve student performance prediction accuracy?**, we developed four methods that integrate teacher observations with students' previous scores to predict academic outcomes.

The first method, the Reports model, generates report-level predictions by embedding individual reports and performing post-prediction aggregation. The Students model performs pre-prediction aggregation by combining all reports for each student before prediction. The Linguistic, Psychological, and Sentiment (LPS) model extracts linguistic, psychological, and sentiment features from teacher comments to enhance prediction accuracy. Finally, the Sentiment-based Similarity Learning (SBSL) model uses a deep Siamese neural network to learn sentiment-based representations from teacher comments, further improving prediction performance. Our experiments demonstrated that incorporating teacher observations with students' previous scores significantly improves performance prediction accuracy. This work highlights the importance of combining human expertise with AI models to enhance predictive accuracy and provide actionable insights for early intervention.

In the second part, we made two contributions to automatic short answer assessment. First, our comparative analysis of instance-based and reference-based scoring approaches revealed that instance-based methods excel with larger datasets, while reference-based methods show superior performance in zero-shot settings. These findings provide guidance for implementing ASAS systems across different educational contexts. Second, we introduced a novel

retrieval-augmented generation system for ASAS with feedback (ASAS-F) that minimizes fine-tuning requirements while maintaining high accuracy. Our system dynamically incorporates relevant human-scored examples to generate scores and feedback that align with human judgment, demonstrating superior scoring performance on unseen questions compared to fine-tuned baselines.

9.1 Contributions

This thesis makes several key contributions to the field of AI in education.

C1: We developed the TOPP framework for integrating teacher observations into student performance prediction models. Specifically, we crafted advanced feature extraction methodologies that combine topic modeling, sentiment analysis, and psychological feature extraction to capture the rich qualitative data present in teacher observation reports. We introduced a sentiment-based similarity learning method utilizing Sentence-BERT to generate optimized embeddings, enhancing the representation of teacher reports. By integrating qualitative teacher observation reports into our predictive models, we demonstrated significant improvements in prediction accuracy over traditional methods that rely solely on quantitative data. Our work includes a comprehensive empirical analysis of the impact of teacher observations on performance prediction across different feature sets and representation methods, highlighting the value of qualitative insights in educational data analytics.

C2: We advanced automatic short answer assessment through two significant studies. In **C2.1**, we conducted a comparative analysis of instance-based and reference-based scoring methods. We evaluated the strengths and limitations of each approach across various scenarios, identifying the optimal methods for both traditional settings and zero-shot learning settings where data is scarce. This analysis provided insights into the generalization capabilities of both scoring paradigms, offering guidance for the implementation of ASAS systems in different educational contexts.

In **C2.2**, we developed a scalable RAG-based system for combined scoring and feedback generation in ASAS. This system dynamically incorporates relevant human-scored examples to enhance adaptability and reduce the need for extensive fine-tuning. We demonstrated that our system achieves improved scoring accuracy that aligns closely with human judgment while requiring fewer resources compared to traditional fine-tuned models. This approach offers a practical and efficient solution for generating both scores and formative feedback, facilitating scalable and personalized assessment in educational settings.

9.2 Limitations and Future Work

Despite the contributions made, this thesis has certain limitations that present opportunities for future research.

Regarding **C1**, while the TOPP framework effectively integrates teacher observations into performance prediction models, the qualitative nature of these observations introduces inherent subjectivity. Teacher observations are prone to biases such as prejudgments, selective perception, and stereotyping, which can undermine the accuracy and consistency of assessments, especially without standardized tools or protocols. Future work could focus on developing automated or semi-automated systems to assist educators in enhancing the objectivity and consistency of observational data. Additionally, the sentiment-based similarity learning approach relied on binary sentiment labels, which showed only a moderate correlation with final scores. This reliance limits the model's ability to capture nuanced emotional or contextual variations in teacher comments. Exploring more complex sentiment analysis models, including multi-class sentiment categorization or continuous sentiment scales, could enhance predictive performance by modeling these subtleties more effectively.

For **C2.1**, the comparative analysis of instance-based and reference-based scoring methods was limited to one state-of-the-art approach for each method and was conducted on a single specific scarce dataset. This narrow scope may not fully capture the strengths and weaknesses of these methods across different contexts. Future studies should broaden the range of techniques

analyzed and include diverse datasets to provide a more comprehensive understanding of the applicability and generalization capabilities of these scoring paradigms.

In the case of **C2.2**, the ASAS-F system's evaluation of feedback quality may not fully align with its pedagogical effectiveness or human evaluators' judgments. Traditional evaluation metrics may miss key aspects and semantics of the feedback, and our human evaluation was limited to a small sample size and focused solely on our system's generated feedback. Future work could explore more comprehensive evaluation methodologies including a larger-scale human evaluation. Moreover, the ASAS-F system heavily depends on the quality and relevance of retrieved examples, making it vulnerable to failures in the retrieval system. In situations involving unseen questions, inadequate examples can compromise both scoring accuracy and feedback quality. Future research should focus on developing robust fallback strategies to maintain performance even when ideal examples are unavailable. Additionally, integrating teacher-in-the-loop approaches, where educators refine and validate system outputs, could ensure alignment with pedagogical goals and enhance the overall effectiveness of AI-driven assessment tools.

Bibliography

[1]Robert A Bloodgood, Jerry G Short, John M Jackson, and James R Martindale. "A change to pass/fail grading in the first two years at one medical school results in improved psychological well-being". In: *Academic Medicine* 84.5 (2009), pp. 655–662 (cit. on p. 1).

[2]Francis Akubuilo. "Holistic assessment of student's learning outcome". In: *Journal of Education and Practice* 3.12 (2012) (cit. on p. 1).

[3]Valerie J Shute. "Focus on formative feedback". In: *Review of educational research* 78.1 (2008), pp. 153–189 (cit. on pp. 2, 93, 98).

[4]ZS Harris. *Distributional structure*. 1954 (cit. on p. 7).

[5]Karen Sparck Jones. "A statistical interpretation of term specificity and its application in retrieval". In: *Journal of documentation* 28.1 (1972), pp. 11–21 (cit. on p. 9).

[6]F. Pedregosa, G. Varoquaux, A. Gramfort, et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on pp. 10, 34, 41).

[7]Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems* 26 (2013) (cit. on p. 10).

[8]Jeffrey Pennington, Richard Socher, and Christopher D Manning. "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543 (cit. on p. 10).

[9]Matthew E. Peters, Mark Neumann, Mohit Iyyer, et al. "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Ed. by Marilyn Walker, Heng Ji, and Amanda Stent. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 2227–2237 (cit. on p. 10).

[10]A Vaswani. “Attention is all you need”. In: *Advances in Neural Information Processing Systems* (2017) (cit. on pp. 11, 15).

[11]Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv:1810.04805* (2018) (cit. on p. 12).

[12]Yinhan Liu. “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692* 364 (2019) (cit. on p. 13).

[13]K Clark. “Electra: Pre-training text encoders as discriminators rather than generators”. In: *arXiv preprint arXiv:2003.10555* (2020) (cit. on p. 13).

[14]Nils Reimers and Iryna Gurevych. “Sentence-bert: Sentence embeddings using siamese bert-networks”. In: *arXiv preprint arXiv:1908.10084* (2019) (cit. on pp. 13, 63, 68, 76).

[15]Alan M Turing. *Computing machinery and intelligence*. Springer, 2009 (cit. on p. 14).

[16]Joseph Weizenbaum. “ELIZA—a computer program for the study of natural language communication between man and machine”. In: *Communications of the ACM* 9.1 (1966), pp. 36–45 (cit. on p. 14).

[17]Andrew McCallum, Kamal Nigam, et al. “A comparison of event models for naive bayes text classification”. In: *AAAI-98 workshop on learning for text categorization*. Vol. 752. 1. Madison, WI. 1998, pp. 41–48 (cit. on p. 14).

[18]Lawrence R Rabiner. “A tutorial on hidden Markov models and selected applications in speech recognition”. In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286 (cit. on p. 14).

[19]S Hochreiter. “Long Short-term Memory”. In: *Neural Computation MIT-Press* (1997) (cit. on p. 14).

[20]Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014) (cit. on p. 15).

[21]Jason Wei, Yi Tay, Rishi Bommasani, et al. “Emergent abilities of large language models”. In: *arXiv preprint arXiv:2206.07682* (2022) (cit. on p. 17).

[22]Colin Raffel, Noam Shazeer, Adam Roberts, et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *Journal of machine learning research* 21.140 (2020), pp. 1–67 (cit. on pp. 17, 106).

[23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. “Llama: Open and efficient foundation language models”. In: *arXiv preprint arXiv:2302.13971* (2023) (cit. on pp. 17, 101, 106).

[24] Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, et al. “Dspy: Compiling declarative language model calls into self-improving pipelines”. In: *arXiv preprint arXiv:2310.03714* (2023) (cit. on pp. 19, 99).

[25] Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, et al. “Optimizing Instructions and Demonstrations for Multi-Stage Language Model Programs”. In: *arXiv preprint arXiv:2406.11695* (2024) (cit. on pp. 19, 101).

[26] Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. “Chain-of-thought prompting elicits reasoning in large language models”. In: *Advances in neural information processing systems* 35 (2022), pp. 24824–24837 (cit. on p. 19).

[27] Omar Khattab and Matei Zaharia. “Colbert: Efficient and effective passage search via contextualized late interaction over bert”. In: *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2020, pp. 39–48 (cit. on pp. 20, 94, 102).

[28] Arto Hellas, Petri Ihantola, Andrew Petersen, et al. “Predicting academic performance: a systematic literature review”. In: *Proceedings companion of the 23rd annual ACM conference on innovation and technology in computer science education*. 2018, pp. 175–199 (cit. on pp. 23, 27, 28).

[29] Cheng Lei and Kin Fun Li. “Academic performance predictors”. In: *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*. IEEE. 2015, pp. 577–581 (cit. on pp. 23, 26).

[30] Graham Samuel Maxwell. *Teacher observation in student assessment*. Queensland School Curriculum Council, 2001 (cit. on p. 23).

[31] Menna Fateen and Tsunenori Mine. “Predicting Student Performance Using Teacher Observation Reports.” In: *the Fourteenth International Conference on Educational Data Mining, pp. 481-486, 2021* (2021) (cit. on p. 25).

[32] Menna Fateen, Kyouhei Ueno, and Tsunenori Mine. “An Improved Model to Predict Student Performance using Teacher Observation Reports”. In: *ICCE 2021, pp. 31-40, (2021)* () (cit. on pp. 25, 54, 70).

[33] Menna Fateen and Tsunenori Mine. “Extraction of Useful Observational Features from Teacher Reports for Student Performance Prediction”. In: *International Conference on Artificial Intelligence in Education*. Springer. 2022, pp. 620–625 (cit. on p. 25).

[34] Menna Fateen and Tsunenori Mine. “Using Similarity Learning with SBERT to Optimize Teacher Report Embeddings for Academic Performance Prediction”. In: *International Conference on Artificial Intelligence in Education*. Springer. 2023, pp. 720–726 (cit. on p. 25).

[35] Hermine Baum, Miriam Litchfield, and MF Washburn. “The results of certain standard mental tests as related to the academic records of college seniors”. In: *The American Journal of Psychology* (1919), pp. 307–310 (cit. on p. 25).

[36] Sophie D White, Sybil May, and MF Washburn. “A Study of Freshmen”. In: *The American Journal of Psychology* 28.1 (1917), pp. 151–155 (cit. on p. 25).

[37] Simon P Davies. “The role of notation and knowledge representation in the determination of programming strategy: a framework for integrating models of programming behavior”. In: *Cognitive Science* 15.4 (1991), pp. 547–572 (cit. on p. 25).

[38] Elliot Soloway. “Learning to program= learning to construct mechanisms and explanations”. In: *Communications of the ACM* 29.9 (1986), pp. 850–858 (cit. on p. 25).

[39] Petri Ihantola, Arto Vihavainen, Alireza Ahadi, et al. “Educational data mining and learning analytics in programming: Literature review and case studies”. In: *Proceedings of the 2015 ITiCSE on working group reports* (2015), pp. 41–63 (cit. on p. 25).

[40] Nguyen Thai Nghe, Paul Janecek, and Peter Haddawy. “A comparative analysis of techniques for predicting academic performance”. In: *2007 37th annual frontiers in education conference-global engineering: knowledge without borders, opportunities without passports*. IEEE. 2007, T2G–7 (cit. on p. 26).

[41] M Ramaswami and R Rathinasabapathy. “Student performance prediction”. In: *International Journal of Computational Intelligence and Informatics* 1.4 (2012), pp. 231–235 (cit. on p. 26).

[42] K Mthimunye and FM Daniels. “Predictors of academic performance, success and retention amongst undergraduate nursing students: A systematic review”. In: *South African Journal of Higher Education* 33.1 (2019), pp. 200–220 (cit. on p. 26).

[43] Ernesto Pathros Ibarra García and Pablo Medina Mora. “Model prediction of academic performance for first year students”. In: *2011 10th Mexican international conference on artificial intelligence*. IEEE. 2011, pp. 169–174 (cit. on p. 26).

[44] Rahaf Alamri and Basma Alharbi. “Explainable Student Performance Prediction Models: A Systematic Review”. In: *IEEE Access* (2021) (cit. on p. 26).

[45]Wei Zhang, Yilin Zhou, and Baolin Yi. “An Interpretable Online Learner’s Performance Prediction Model Based on Learning Analytics”. In: *Proceedings of the 2019 11th International Conference on Education Technology and Computers*. 2019, pp. 148–154 (cit. on p. 26).

[46]Jingyi Luo, Shaymaa E Sorour, Kazumasa Goda, and Tsunenori Mine. “Predicting Student Grade Based on Free-Style Comments Using Word2Vec and ANN by Considering Prediction Results Obtained in Consecutive Lessons.” In: *International Educational Data Mining Society* (2015) (cit. on p. 26).

[47]Shaymaa E Sorour, Kazumasa Goda, and Tsunenori Mine. “Comment data mining to estimate student performance considering consecutive lessons”. In: *Journal of Educational Technology & Society* 20.1 (2017), pp. 73–86 (cit. on p. 26).

[48]J Jayaraman. “Predicting student dropout by mining advisor notes”. In: *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*. 2020, pp. 629–632 (cit. on p. 26).

[49]Juan L Rastrollo-Guerrero, Juan A Gomez-Pulido, and Arturo Durán-Domínguez. “Analyzing and predicting students’ performance by means of machine learning: A review”. In: *Applied sciences* 10.3 (2020), p. 1042 (cit. on p. 27).

[50]Robert J Lowe. “Cram schools in Japan: The need for research”. In: *The Language Teacher* 39.1 (2015), pp. 26–31 (cit. on p. 28).

[51]Tianqi Chen and Carlos Guestrin. “Xgboost: A scalable tree boosting system”. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794 (cit. on p. 34).

[52]Clyde Vroman. *Japan: a Study of the Educational System of Japan and Guide to the Academic Placement of Students from Japan in United States Educational Institutions: Placement Recommendations by the Council on Evaluation of Foreign Student Credentials, Meeting July 29-30, 1965*. American Association of Collegiate Registrars and Admissions Officers, 1966 (cit. on p. 34).

[53]Agoritsa Polyzou and George Karypis. “Grade prediction with models specific to students and courses”. In: *International Journal of Data Science and Analytics* 2.3 (2016), pp. 159–171 (cit. on pp. 34, 35).

[54]Zhiyun Ren, Xia Ning, Andrew S Lan, and Huzefa Rangwala. “Grade Prediction Based on Cumulative Knowledge and Co-taken Courses.” In: *International Educational Data Mining Society* (2019) (cit. on p. 35).

[55]Paul McCann. “fugashi, a Tool for Tokenizing Japanese in Python”. In: *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 44–51 (cit. on p. 40).

[56] Taku Kudo. “Mecab: Yet another part-of-speech and morphological analyzer”. In: *http://mecab. sourceforge. jp* (2006) (cit. on pp. 40, 41).

[57] Laurie Murphy and Lynda Thomas. “Dangers of a fixed mindset: implications of self-theories research for computer science education”. In: *Proceedings of the 13th annual conference on Innovation and technology in computer science education*. 2008, pp. 271–275 (cit. on p. 51).

[58] Mantz Yorke* and Peter Knight. “Self-theories: some implications for teaching and learning in higher education”. In: *Studies in Higher Education* 29.1 (2004), pp. 25–37 (cit. on p. 51).

[59] Samuel Cunningham-Nelson, Mahsa Baktashmotlagh, and Wageeh Boles. “Visualizing student opinion through text analysis”. In: *IEEE Transactions on Education* 62.4 (2019), pp. 305–311 (cit. on p. 52).

[60] Johanna Watkins, Marcos Fabielli, and Mufti Mahmud. “Sense: a student performance quantifier using sentiment analysis”. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–6 (cit. on p. 52).

[61] Alvaro Ortigosa, José M Martín, and Rosa M Carro. “Sentiment analysis in Facebook and its application to e-learning”. In: *Computers in human behavior* 31 (2014), pp. 527–541 (cit. on p. 52).

[62] Sannyuya Liu, Xian Peng, Hercy NH Cheng, et al. “Unfolding sentimental and behavioral tendencies of learners’ concerned topics from course reviews in a MOOC”. In: *Journal of Educational Computing Research* 57.3 (2019), pp. 670–696 (cit. on p. 52).

[63] Gianluca Elia, Gianluca Solazzo, Gianluca Lorenzo, and Giuseppina Passiante. “Assessing learners’ satisfaction in collaborative online courses through a big data approach”. In: *Computers in Human Behavior* 92 (2019), pp. 589–599 (cit. on p. 52).

[64] Sujata Rani and Parteek Kumar. “A sentiment analysis system to improve teaching and learning”. In: *Computer* 50.5 (2017), pp. 36–43 (cit. on p. 52).

[65] Zhi Liu, Chongyang Yang, Sylvio Rüdian, et al. “Temporal emotion-aspect modeling for discovering what students are concerned about in online course forums”. In: *Interactive Learning Environments* 27.5-6 (2019), pp. 598–627 (cit. on p. 52).

[66] Chee Kian Leong, Yew Haur Lee, and Wai Keong Mak. “Mining sentiments in SMS texts for teaching evaluation”. In: *Expert Systems with applications* 39.3 (2012), pp. 2584–2589 (cit. on p. 52).

[67]Reinhard Pekrun, Thomas Goetz, Wolfram Titz, and Raymond P Perry. “Academic emotions in students’ self-regulated learning and achievement: A program of qualitative and quantitative research”. In: *Educational psychologist* 37.2 (2002), pp. 91–105 (cit. on p. 53).

[68]Liang-Chih Yu, Cheng-Wei Lee, HI Pan, et al. “Improving early prediction of academic failure using sentiment analysis on self-evaluated comments”. In: *Journal of Computer Assisted Learning* 34.4 (2018), pp. 358–365 (cit. on p. 53).

[69]Scott Crossley, Luc Paquette, Mihai Dascalu, Danielle S McNamara, and Ryan S Baker. “Combining click-stream data with NLP tools to better understand MOOC completion”. In: *Proceedings of the sixth international conference on learning analytics & knowledge*. 2016, pp. 6–14 (cit. on p. 53).

[70]Rebecca L Robinson, Reanelle Navea, and William Ickes. “Predicting final course performance from students’ written self-introductions: A LIWC analysis”. In: *Journal of Language and Social Psychology* 32.4 (2013), pp. 469–479 (cit. on p. 53).

[71]Miaomiao Wen, Diyi Yang, and Carolyn Rosé. “Linguistic reflections of student engagement in massive open online courses”. In: *Proceedings of the international AAAI conference on web and social media*. Vol. 8. 1. 2014, pp. 525–534 (cit. on p. 53).

[72]Miaomiao Wen, Diyi Yang, and Carolyn Rose. “Sentiment Analysis in MOOC Discussion Forums: What does it tell us?” In: *Educational data mining 2014*. Citeseer. 2014 (cit. on p. 53).

[73]*Daigo: BERT-base Japanese Sentiment*. `https://huggingface.co/daigo/bert-base-japanese-sentiment`. Accessed: 2022-01-29 (cit. on pp. 54, 57).

[74]Florian Boudin. “pke: an open source python-based keyphrase extraction toolkit”. In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*. Osaka, Japan, 2016, pp. 69–73 (cit. on p. 55).

[75]Mazar Moradi Fard, Thibaut Thonet, and Eric Gaussier. “Seed-Guided Deep Document Clustering”. In: *Advances in Information Retrieval* 12035 (2020), p. 3 (cit. on p. 55).

[76]Yu Meng, Jiaxin Huang, Guangyuan Wang, et al. “Discriminative topic mining via category-name guided text embedding”. In: *Proceedings of The Web Conference 2020*. 2020, pp. 2121–2132 (cit. on p. 55).

[77] Tasuku Igarashi and Kazutoshi Sasahara. “Development of the Japanese Version of the Linguistic Inquiry and Word Count Dictionary 2015 (J-LIWC2015)”. In: (2021) (cit. on p. 58).

[78] Francesco Barbieri, Luis Espinosa Anke, and Jose Camacho-Collados. “XLM-T: Multilingual Language Models in Twitter for Sentiment Analysis and Beyond”. In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, June 2022, pp. 258–266 (cit. on p. 65).

[79] Ellis B Page. “The imminence of... grading essays by computer”. In: *The Phi Delta Kappan* 47.5 (1966), pp. 238–243 (cit. on p. 75).

[80] Amit Chauhan. “Massive open online courses (MOOCS): Emerging trends in assessment and accreditation”. In: *Digital Education Review* 25 (2014), pp. 7–17 (cit. on p. 75).

[81] Debra Haley, Pete Thomas, Anne De Roeck, and Marian Petre. “Measuring improvement in latent semantic analysis-based marking systems: using a computer to mark questions about HTML”. In: (2007) (cit. on p. 75).

[82] Lynette Hirschman. “Automated grading of short-answer tests”. In: *IEEE Intelligent Systems, Trends and Controversies section* 15.5 (2000), pp. 22–37 (cit. on p. 75).

[83] Steven Burrows, Iryna Gurevych, and Benno Stein. “The eras and trends of automatic short answer grading”. In: *International journal of artificial intelligence in education* 25 (2015), pp. 60–117 (cit. on pp. 76, 93, 95).

[84] Andrea Horbach and Torsten Zesch. “The influence of variance in learner answers on automatic content scoring”. In: *Frontiers in Education*. Vol. 4. Frontiers Media SA. 2019, p. 28 (cit. on pp. 76, 77).

[85] Marie Bexte, Andrea Horbach, and Torsten Zesch. “Similarity-Based Content Scoring-A more Classroom-Suitable Alternative to Instance-Based Scoring?” In: *Findings of the Association for Computational Linguistics: ACL 2023*. 2023, pp. 1892–1903 (cit. on pp. 76, 94, 95, 102).

[86] Jiaqi Lun, Jia Zhu, Yong Tang, and Min Yang. “Multiple data augmentation strategies for improving performance on automatic short answer scoring”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 09. 2020, pp. 13389–13396 (cit. on p. 77).

[87] Omar Nael, Youssef ELmanyalawy, and Nada Sharaf. “AraScore: a deep learning-based system for Arabic short answer scoring”. In: *Array* 13 (2022), p. 100109 (cit. on p. 77).

[88] Aubrey Condor, Max Litster, and Zachary Pardos. “Automatic Short Answer Grading with SBERT on Out-of-Sample Questions.” In: *International Educational Data Mining Society* (2021) (cit. on p. 77).

[89] Nigel Fernandez, Aritra Ghosh, Naiming Liu, et al. “Automated scoring for reading comprehension via in-context bert tuning”. In: *International Conference on Artificial Intelligence in Education*. Springer. 2022, pp. 691–697 (cit. on pp. 77, 81).

[90] Mengxue Zhang, Sami Baral, Neil Heffernan, and Andrew Lan. “Automatic short math answer grading via in-context meta-learning”. In: *arXiv preprint arXiv:2205.15219* (2022) (cit. on p. 77).

[91] Wael Hassan Gomaa and Aly Aly Fahmy. “Automatic scoring for answers to Arabic test questions”. In: *Computer Speech & Language* 28.4 (2014), pp. 833–857 (cit. on p. 77).

[92] Abdulaziz Shehab, Mahmoud Faroun, and Magdi Rashad. “An automatic Arabic essay grading system based on text similarity Algorithms”. In: *International Journal of Advanced Computer Science and Applications* 9.3 (2018) (cit. on p. 77).

[93] Maram Meccawy, Afnan Ali Bayazed, Bashayer Al-Abdullah, and Hind Algamdi. “Automatic Essay Scoring for Arabic Short Answer Questions using Text Mining Techniques”. In: *International Journal of Advanced Computer Science and Applications* 14.6 (2023) (cit. on p. 77).

[94] Bo Wang, Billy Dawton, Tsunenori Ishioka, and Tsunenori Mine. “Optimizing Answer Representation using Metric Learning for Efficient Short Answer Scoring”. In: *The 20th Pacific Rim International Conference on Artificial Intelligence (PRICAI)* (2023) (cit. on pp. 77, 95).

[95] Leila Ouahrani and Djamal Bennouar. “AR-ASAG an Arabic dataset for automatic short answer grading evaluation”. In: *Proceedings of the Twelfth Language Resources and Evaluation Conference*. 2020, pp. 2634–2643 (cit. on p. 78).

[96] Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, et al. “Efficient few-shot learning without prompts”. In: *arXiv preprint arXiv:2209.11055* (2022) (cit. on p. 80).

[97] Wissam Antoun, Fady Baly, and Hazem Hajj. “AraBERT: Transformer-based Model for Arabic Language Understanding”. In: *LREC 2020 Workshop Language Resources and Evaluation Conference 11–16 May 2020*, p. 9 (cit. on p. 80).

[98] Gaoyan Lv, Wei Song, Miaomiao Cheng, and Lizhen Liu. “Exploring the effectiveness of question for neural short answer scoring system”. In: *2021 IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC) 2021 IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC)*. IEEE. 2021, pp. 1–4 (cit. on p. 84).

[99] John Hattie and Helen Timperley. “The power of feedback”. In: *Review of educational research* 77.1 (2007), pp. 81–112 (cit. on p. 93).

[100] Brian Riordan, Andrea Horbach, Aoife Cahill, Torsten Zesch, and Chungmin Lee. “Investigating neural architectures for short answer scoring”. In: *Proceedings of the 12th workshop on innovative use of NLP for building educational applications*. 2017, pp. 159–168 (cit. on pp. 93, 95).

[101] Lakshmi Ramachandran, Jian Cheng, and Peter Foltz. “Identifying patterns for short answer scoring using graph-based lexico-semantic text matching”. In: *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*. 2015, pp. 97–106 (cit. on pp. 93, 95).

[102] Naomi E Winstone, Robert A Nash, Michael Parker, and James Rowntree. “Supporting learners’ agentic engagement with feedback: A systematic review and a taxonomy of recipience processes”. In: *Educational psychologist* 52.1 (2017), pp. 17–37 (cit. on p. 93).

[103] Hui Liu, Qingyu Yin, and William Yang Wang. “Towards explainable NLP: A generative explanation framework for text classification”. In: *arXiv preprint arXiv:1811.00196* (2018) (cit. on p. 93).

[104] Hassan Khosravi, Simon Buckingham Shum, Guanliang Chen, et al. “Explainable artificial intelligence in education”. In: *Computers and Education: Artificial Intelligence* 3 (2022), p. 100074 (cit. on p. 93).

[105] Jiazheng Li, Lin Gui, Yuxiang Zhou, et al. “Distilling ChatGPT for Explainable Automated Student Answer Assessment”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. 2023, pp. 6007–6026 (cit. on pp. 93, 97).

[106] Johannes Schneider, Bernd Schenk, Christina Niklaus, and Michaelis Vlachos. “Towards llm-based autograding for short textual answers”. In: *arXiv preprint arXiv:2309.11508* (2023) (cit. on pp. 93, 96).

[107] Anna Filighera, Siddharth Parihar, Tim Steuer, Tobias Meuser, and Sebastian Ochs. "Your answer is incorrect... would you like to know why? introducing a bilingual short answer feedback dataset". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022, pp. 8577–8591 (cit. on pp. 93, 97, 104, 106).

[108] Myroslava Dzikovska, Natalie Steinhauser, Elaine Farrow, Johanna Moore, and Gwendolyn Campbell. "BEETLE II: Deep natural language understanding and automatic feedback generation for intelligent tutoring in basic electricity and electronics". In: *International Journal of Artificial Intelligence in Education* 24 (2014), pp. 284–332 (cit. on p. 94).

[109] Hieke Keuning, Johan Jeuring, and Bastiaan Heeren. "A systematic literature review of automated feedback generation for programming exercises". In: *ACM Transactions on Computing Education (TOCE)* 19.1 (2018), pp. 1–43 (cit. on p. 94).

[110] Patrick Lewis, Ethan Perez, Aleksandra Piktus, et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9459–9474 (cit. on p. 94).

[111] Xiaoming Xi. *Automated scoring and feedback systems: Where are we and where are we heading?* 2010 (cit. on p. 95).

[112] Stephen P Balfour. "Assessing Writing in MOOCs: Automated Essay Scoring and Calibrated Peer Review™." In: *Research & Practice in Assessment* 8 (2013), pp. 40–48 (cit. on p. 95).

[113] Zixuan Ke and Vincent Ng. "Automated Essay Scoring: A Survey of the State of the Art." In: *IJCAI*. Vol. 19. 2019, pp. 6300–6308 (cit. on p. 95).

[114] Yaman Kumar, Swati Aggarwal, Debanjan Mahata, et al. "Get it scored using autosas—an automated system for scoring short answers". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 9662–9669 (cit. on p. 95).

[115] Chul Sung, Tejas Indulal Dhamecha, and Nirmal Mukhi. "Improving short answer grading using transformer-based pre-training". In: *Artificial Intelligence in Education: 20th International Conference, AIED 2019, Chicago, IL, USA, June 25-29, 2019, Proceedings, Part I 20*. Springer. 2019, pp. 469–481 (cit. on p. 95).

[116] Kenneth Steimel and Brian Riordan. "Towards instance-based content scoring with pre-trained transformer models". In: *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*. Vol. 34. 2020 (cit. on p. 95).

[117] Menna Fateen and Tsunenori Mine. “In-Context Meta-Learning vs. Semantic Score-Based Similarity: A Comparative Study in Arabic Short Answer Grading”. In: *Proceedings of ArabicNLP 2023*. Ed. by Hassan Sawaf, Samhaa El-Beltagy, Wajdi Zaghouani, et al. Singapore (Hybrid): Association for Computational Linguistics, Dec. 2023, pp. 350–358 (cit. on p. 95).

[118] Salam Albatarni, Sohaila Eltanbouly, and Tamer Elsayed. “Graded Relevance Scoring of Written Essays with Dense Retrieval”. In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2024, pp. 1329–1338 (cit. on p. 96).

[119] Imran Chamieh, Torsten Zesch, and Klaus Giebermann. “LLMs in Short Answer Scoring: Limitations and Promise of Zero-Shot and Few-Shot Approaches”. In: *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*. 2024, pp. 309–315 (cit. on p. 96).

[120] Li-Hsin Chang and Filip Ginter. “Automatic Short Answer Grading for Finnish with ChatGPT”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 21. 2024, pp. 23173–23181 (cit. on p. 96).

[121] Josh Achiam, Steven Adler, Sandhini Agarwal, et al. “Gpt-4 technical report”. In: *arXiv preprint arXiv:2303.08774* (2023) (cit. on p. 96).

[122] Tamara P Tate, Jacob Steiss, Drew Bailey, et al. “Can AI provide useful holistic essay scoring?” In: *Computers and Education: Artificial Intelligence* 7 (2024), p. 100255 (cit. on p. 96).

[123] Anna Filighera, Joel Tschesche, Tim Steuer, Thomas Tregel, and Lisa Wernet. “Towards generating counterfactual examples as automatic short answer feedback”. In: *International Conference on Artificial Intelligence in Education*. Springer. 2022, pp. 206–217 (cit. on p. 97).

[124] Kevin P Yancey, Geoffrey Laflair, Anthony Verardi, and Jill Burstein. “Rating short l2 essays on the cefr scale with gpt-4”. In: *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*. 2023, pp. 576–584 (cit. on p. 97).

[125] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, et al. “Mixtral of experts”. In: *arXiv preprint arXiv:2401.04088* (2024) (cit. on pp. 101, 106).

[126] Gloria Ashiya Katuka, Alexander Gain, and Yen-Yun Yu. “Investigating Automatic Scoring and Feedback using Large Language Models”. In: *arXiv preprint arXiv:2405.00602* (2024) (cit. on p. 106).

[127] Matt Post. “A call for clarity in reporting BLEU scores”. In: *arXiv preprint arXiv:1804.08771* (2018) (cit. on p. 112).

[128]Chin-Yew Lin. “ROUGE: A Package for Automatic Evaluation of Summaries”. In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 74–81 (cit. on p. 112).

[129]Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. “Bertscore: Evaluating text generation with bert”. In: *arXiv preprint arXiv:1904.09675* (2019) (cit. on p. 112).

[130]Kaitlyn Zhou, Jena D Hwang, Xiang Ren, and Maarten Sap. “Relying on the Unreliable: The Impact of Language Models’ Reluctance to Express Uncertainty”. In: *arXiv preprint arXiv:2401.06730* (2024) (cit. on p. 116).

[131]Ziwei Ji, Nayeon Lee, Rita Frieske, et al. “Survey of hallucination in natural language generation”. In: *ACM Computing Surveys* 55.12 (2023), pp. 1–38 (cit. on p. 116).

Appendix

Publications

Journal

1. Fateen, Menna, Bo Wang, and Tsunenori Mine. "Beyond Scores: A Modular RAG-Based System for Automatic Short Answer Scoring with Feedback." IEEE Access (2024).

Conference Proceedings

1. Fateen, Menna, and Tsunenori Mine. "Predicting Student Performance Using Teacher Observation Reports." 14th International Conference on Educational Data Mining, EDM 2021. International Educational Data Mining Society, 2021.
2. Fateen, Menna, Kyouhei Ueno, and Tsunenori Mine. "An Improved Model to Predict Student Performance using Teacher Observation Reports." 29th International Conference on Computers in Education Conference, ICCE 2021. Asia-Pacific Society for Computers in Education, 2021.
3. Fateen, Menna, and Tsunenori Mine. "Extraction of useful observational features from teacher reports for student performance prediction." In International Conference on Artificial Intelligence in Education, pp. 620-625. Cham: Springer International Publishing, 2022.

4. Fateen, Menna, and Tsunenori Mine. "Using Similarity Learning with SBERT to Optimize Teacher Report Embeddings for Academic Performance Prediction." In International Conference on Artificial Intelligence in Education, pp. 720-726. Cham: Springer Nature Switzerland, 2023.

5. Fateen, Menna, and Tsunenori Mine. "In-Context Meta-Learning vs. Semantic Score-Based Similarity: A Comparative Study in Arabic Short Answer Grading." In Proceedings of ArabicNLP 2023, pp. 350-358. 2023.