Semi-Automated Grading of Free-Text Computer Science Answers

Mohammad Zaid Moonsamy, Richard Klein
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

***Abstract*—Manual grading of free-text responses in Data Structures and Algorithms (DSA) courses presents a significant challenge for educators, requiring substantial time and resources. This paper introduces a semi-automated grading system that combines Large Language Models (LLMs) with clustering techniques to reduce grading workload while maintaining assessment quality. Our key contribution is a novel Greedy Monte Carlo (GMC) sampling strategy that intelligently selects which submissions to mark manually, while providing confidence metrics for automated grading decisions. We evaluate our system using a dataset of anonymized student responses from a university-level DSA course. Our experiments reveal that BERT embeddings consistently outperform GPT-2 for representing student responses, and k-means clustering provides more reliable cluster distributions compared to hierarchical clustering approaches. The GMC sampling strategy demonstrates superior performance over random sampling, achieving Pearson correlation scores of up to 0.7 with human-assigned grades while requiring significantly fewer manual markings. Importantly, the system provides educators with confidence measures for automated grades, enabling informed decisions about when additional manual grading is necessary. These results suggest that our approach can substantially reduce grading effort while maintaining assessment reliability, offering a practical solution for scaling assessment in computer science education.**

I. Introduction

Automated grading systems have been a subject of research and interest for decades, driven by the desire to develop computer-based assessment solutions. While objective questions like multiple-choice, fill-in-the-blank, and matching exercises are easy to grade automatically, they provide only a limited view of student knowledge [1]. According to Bloom's Taxonomy, evaluating higher-order thinking skills requires open-ended formats like paragraphs and essays. Despite the complexity of grading these free-form responses with their unlimited answer space, educators prefer them for assessing deeper conceptual understanding [1], [2]. This manual grading process is particularly burdensome for those with large classes, with studies showing educators spend up to 30% of their time grading – time that could be devoted to teaching and research [3].

Researchers have developed automated grading systems ranging from basic tools to advanced transformer-based approaches, which outperform traditional CNN and RNN models for grading essays and short answers [4], [5]. These systems differ in automation levels and in how they handle various response types, such as content-based or style-based answers [1], [6]. While much research emphasizes fully automated deep learning methods, the "black box" nature of Large Language Models (LLMs) complicates interpretability, limiting transparency and trust in the assigned grades [1], [7], [8]. Additionally, training these models often demands large datasets, which can be impractical. Few studies explore semi-automated systems that combine human input with automated grading, particularly for domain-specific areas like Data Structures and Algorithms [8]–[10]. This gap is further evident in the lack of research on integrating transformer models in semi-automated grading systems with confidence metrics to indicate reliability in auto-graded submissions.

To address these gaps, we propose a semi-automated grading system that combines transformer models and unsupervised learning techniques to support human graders in the assessment process. The system leverages LLMs like BERT and GPT-2 for rich semantic representation, along with clustering and sampling techniques, to enable both efficient grading and meaningful human intervention. Our approach enhances grading efficiency while ensuring interpretability and providing a confidence metric—a measure indicating the reliability of auto-graded scores. We evaluate the system on a dataset from the IDSA (Introduction to Data Structures and Algorithms) course, comprising student responses from the University of the Witwatersrand, and compare its performance to a baseline semi-automated system without confidence measures. Results demonstrate that our system not only correlates more strongly with human-assigned marks but also offers a dependable confidence metric for the auto-graded responses.

The paper is organized as follows. Section 2 provides background on Large Language Models (LLMs) and the unsupervised techniques used in our system. In Section 3, we situate our work within the context of existing research. Section 4 details the methodology behind the development of our semi-automated grading system, while Section 5 presents and analyzes our results. Finally, Section 6 offers conclusions and suggestions for future research.

II. Background

In this section, we present the foundational concepts and research underlying our study. We begin with an overview of Large Language Models (LLMs), specifically BERT and GPT-2, which serve as the backbone of our semi-automated grading system. Next, we explore the clustering algorithms

and Monte Carlo Sampling techniques integral to the system's functionality. Finally, we outline the evaluation metrics used to assess grading systems and the statistical tests applied in our analysis.

A. Large Language Models

1) BERT: Bidirectional Encoder Representations from Transformers (BERT), introduced by Devlin et al. [11] in 2018, is a transformer-based model designed to capture deep bidirectional context by simultaneously considering both left and right contexts of words across all model layers. This bidirectional understanding is critical for precise language interpretation where context often dictates meaning. BERT's training relies on two objectives: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). In MLM, randomly selected tokens within sentences are masked, and the model learns to predict these tokens using surrounding words. This objective forces BERT to develop a bidirectional understanding of language, as it must consider both preceding and succeeding words to predict the masked token. NSP allows BERT to learn sentence pair relationships, which is valuable for tasks that demand cross-sentence comprehension [11]. Together, MLM and NSP equip BERT with an advanced capacity to interpret complex language.

2) GPT-2: Generative Pre-trained Transformer (GPT), developed by Radford et al. [12] in 2018, is another prominent LLM that demonstrates the versatility of the transformer architecture for a range of NLP tasks. Unlike BERT's bidirectional approach, GPT applies unidirectional context, predicting each word based only on previous words in the sequence. GPT-2 is pre-trained on a large corpus using a standard language modeling objective, enabling it to generate coherent text and understand various language tasks [12]. This pre-training is followed by fine-tuning on specific tasks, where GPT-2 adapts its linguistic knowledge for different applications.

The advanced language processing capabilities of BERT and GPT-2 excel at capturing semantic meaning in text which makes them suitable for accurately interpreting student answers.

B. Clustering Algorithms

1) K-Means: K-means clustering iteratively assigns data points to the nearest of k pre-specified centroids, then recalculates cluster means as new centroids until convergence [9], [13], [14]. While it offers linear complexity and scales well with large datasets, k-means requires knowing the desired number of clusters beforehand.

2) Agglomerative Hierarchical Clustering: Agglomerative Hierarchical Clustering (AHC) is a bottom-up approach where each observation initially forms its own cluster [15]. The algorithm iteratively merges the most similar clusters based on distance metrics like Euclidean distance and linkage criteria (single, complete, or average-linkage) until all points form a single cluster. Average and complete linkage merge clusters by minimizing the average distance (average linkage) or the maximum distance (complete linkage) between points in the clusters. While AHC doesn't require pre-specifying cluster numbers, it is sensitive to noise and outliers. Once merged, clusters cannot be readjusted [16].

C. Monte Carlo Sampling

Monte Carlo sampling is a method that uses repeated random sampling to generate numerical solutions, particularly helpful when deterministic methods are infeasible, such as in systems with numerous variables or uncertainties [17]. The typical process involves defining the problem domain and relevant probability distributions, generating random samples, performing computations on these samples, and analyzing aggregate properties like mean, variance, or confidence intervals.

Often combined with techniques like Markov Chain Monte Carlo (MCMC) to enhance sampling efficiency and convergence, Monte Carlo methods are well-suited for complex assessments. In our grading system, Monte Carlo sampling allows for not only a grade assignment but also an estimate of grading uncertainty, supporting a more transparent and comprehensive evaluation of student performance.

D. Common evaluation metrics

1) Quadratic Weighted Kappa: Quadratic Weighted Kappa (QWK) is a statistical measure used to assess the agreement between raters for expected and predicted scores. It quantifies the level of agreement beyond chance. The value of the parameter k is determined by the formula:

$$k = \frac{p_0 - p_e}{1 - p_e},$$

where p_0 represents the observed agreement and p_e represents the expected agreement. QWK values range from 0 to 1, where 1 represents perfect agreement, indicating that the expected and predicted scores match exactly, and 0 indicates complete disagreement, suggesting no agreement between the raters [4]. Notably, most automated grading research predominantly uses QWK as the primary evaluation metric [5], [7].

2) Pearson's r: Generating values ranging from -1 to 1, this method assesses the correlation between two continuous variables. A score of 1 indicates a positive correlation, 0 implies no correlation, and -1 indicates a negative correlation. It is calculated using

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}},$$

where X and Y are the distributions, the variables x_i and y_i represent the ith values of these distributions, while $\bar{x}$ and $\bar{y}$ denote the respective mean values [4]. In the domain of automated answer grading, Pearson's correlation emerges as one of the most widely utilised metrics for comparing marks assigned by instructors with predicted marks [1], [7], [8].

E. Statistical Tests

Statistical tests are essential tools in research for determining whether observed differences between groups or conditions are meaningful rather than due to chance. Commonly used in model evaluation, the t-test and Wilcoxon signed-rank test help determine if differences in model performance reflect genuine improvements. Details on the formulas can be found in Appendix A.

1) t-Test: The t-test is a parametric statistical test used to assess whether there is a significant difference between the means of two groups. In the context of model evaluation, it helps researchers determine if the performance difference between two models or two versions of the same model represents a genuine improvement or if it could have occurred by chance. The t-test assumes that the data follows a normal distribution and is particularly useful for comparing continuous measurements or scores. A higher $|t|$ value indicates a greater likelihood that the observed difference between the groups is statistically significant [18], [19].

2) Wilcoxon Signed-Rank Test: The Wilcoxon signed-rank test serves as a non-parametric alternative to the t-test. Unlike the t-test, it does not assume a normal distribution of the data, making it more robust for cases where this assumption may not hold. The Wilcoxon test works by comparing the relative rankings of paired differences, making it particularly useful when working with scores that may not be normally distributed or when dealing with outliers. A higher or lower W value, depending on the distribution of ranks, indicates a statistically significant difference between the paired samples [20], [21].

3) Effect Size: Cohen's d: To quantify the magnitude of differences, researchers often complement these statistical tests with effect size measurements such as Cohen's d. Effect size metrics help interpret the practical significance of a difference, rather than just its statistical significance. To assess the practical significance of these differences, effect size measures like Cohen's d are often used alongside statistical tests, providing context beyond statistical significance alone [22].

III. Related Work

The field of automated grading systems has witnessed significant advancements, with research focusing on key divisions such as essay versus short answer grading, and formative versus summative assessment. Short answers, often a few lines long, are primarily evaluated for content, while essays are assessed mainly on writing style, with content also playing a role [1], [6], [7]. Similarly, formative assessments provide guidance during learning, while summative assessments evaluate knowledge at the end of a learning activity [1], [6].

Early research on automated grading relied on traditional feature extraction and classical machine learning algorithms [1], [6], [10]. However, these methods had limitations in automatically learning complex data representations from raw text. The advent of deep learning techniques, such as CNN-LSTM architectures with attention mechanisms, have shown promise in achieving high scoring accuracy for automated grading [23], [24]. With the emergence of transformer models such as BERT and GPT, further advancements have been made, including pre-training on domain-specific data and leveraging transfer learning [5], [25].

While fully automated grading systems have achieved state-of-the-art results, they require large amounts of labeled data for training, which can be challenging to obtain, especially in domains with limited available data [4]. The use of deep learning models in these systems raises ongoing concerns regarding their explainability and interpretability. Additionally, there are concerns regarding the reliability and trust of grades in fully automated systems, as they can be vulnerable to adversarial student texts representing potential cheating attempts [7], [8].

In contrast, semi-automated grading systems have not been explored as extensively. These systems assume that semantically similar submissions reside close to each other in embedding space and should receive similar marks [1], [9], [10]. The core approach involves using machine learning or natural language processing techniques to generate representations of student answers, then employing clustering algorithms to group similar responses. A sampling strategy is then used to select a subset of submissions from each cluster for manual marking, with the human-assigned marks extrapolated to the remaining submissions in the cluster [1], [9], [10].

Previous research in semi-automated grading has evolved from the use of traditional feature extraction and unsupervised methods, such as Latent Semantic Analysis (LSA) and k-means clustering [1], [10], to more sophisticated techniques involving transformer model embeddings and advanced clustering algorithms [9]. However, the literature on semi-automated grading systems is limited, and surveys have not heavily emphasized their importance [4], [7], [8].

To address the limitations of existing research, this study explores fine-tuned transformer embeddings on domain-specific data, such as Data Structures and Algorithms (DSA), combined with clustering and sampling techniques to develop a semi-automated short-answer grading system. Additionally, the system incorporates a confidence measure to address the challenge of trust in automated grading.

IV. Methodology

To address the challenge of automated grading for free-text responses in Data Structures and Algorithms (DSA), we developed a semi-automated grading system inspired by the approach of Klein et al. [1]. The architecture of this system is shown in Fig. 1. Our approach comprises three main components: Embeddings, Clustering, and Sampling. For each component, we conducted experiments to determine the configurations that most effectively optimize the system's performance.

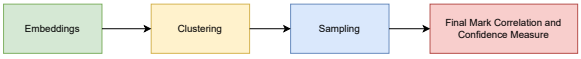


Fig. 1. The Semi-Automated Grading System pipeline

We hypothesize that self-supervised fine-tuning of transformer models, specifically BERT and GPT-2, on a large DSA corpus will improve the quality of text embeddings for student responses. These enhanced embeddings, combined with unsupervised techniques for clustering, sampling, and confidence measurement, will aim to reduce grading workload, increase grading accuracy (in correlation with human scores), and provide reliable confidence estimates for auto-assigned marks. The following subsections discuss each component and the experiments conducted to validate this hypothesis in detail.

A. Datasets and Preprocessing

1) DSA corpus: The study required an extensive Data Structures and Algorithms (DSA) corpus to effectively fine-tune the Large Language Models (LLMs) - BERT and GPT-2. To meet these substantial data requirements, we collected data from multiple authoritative sources. The corpus was assembled through web scraping techniques, incorporating content from university lecture materials, academic textbooks, and established online educational platforms including GeeksforGeeks, W3Schools, MIT OpenCourseWare, and Wikipedia. The dataset was further enriched with transcribed content from DSA-focused YouTube videos. All data collection adhered to copyright regulations, utilising only materials explicitly permitted for research purposes. Table I provides a detailed list of data sources and their licensing information, with PG indicating Permission Granted, NC for Non-Commercial use, and CC for Creative Commons attribution. For more details on the data ethics waiver, please refer to Appendix E.

TABLE I
DSA CORPORA - LIST OF SOURCES

Source Name	Resource Type	License
Wits DSA courses	Textbooks, Lecture Notes and Videos	PG
An open guide to Data Structures and Algorithms [26]	Textbook	CC
Virginia Tech DSAA [27]	Textbook	NC
James Madison University [28]	Notes	CC
Open Data Structures [29]	Textbook	CC
Problem-solving - Algorithms [30]	Textbook	CC
Think DSA [31]	Textbook	CC
Algorithms [32]	Textbook	NC
Wikipedia [33]	Webpages	CC
GeeksforGeeks [34]	Webpages	PG
MIT OpenCourseWare [35], [36]	Notes, Transcripts	CC
W3Schools [37]	Webpages	NC
CS50 [38]	Transcripts	CC
freeCodeCamp [39]	Webpages	CC

The preprocessing involved several critical steps to ensure data quality such as the removal of extraneous elements (whitespace, headers, metadata, and table of contents), preservation of programming-specific punctuation (e.g., semicolons in code segments) and concatenation of processed text from all sources. The final consolidated dataset comprised 189,394 samples. For tokenization, we employed the LLM-specific tokenizer implementing Byte Pair Encoding (BPE), ensuring compatibility with the model architectures.

2) IDSA dataset: For model evaluation, we used anonymized student response data from the Introduction to Data Structures and Algorithms (IDSA) course at the University of the Witwatersrand. This dataset comprised student answers from various assessment types—quizzes, tests, and examinations—with corresponding marks provided as labels. The corpus included responses to 16 distinct questions collected between 2020 and 2022, with approximately 250–400 answers per question. Although the original questions were excluded from the dataset, a sample of dataset entries is presented in Table II.

Pre-processing the IDSA dataset involved eliminating empty submissions to reduce redundancy from zero-mark samples, tokenizing with BERT- and GPT-2-specific tokenizers, normalizing marks to a 0-100 percentage scale (independent of original question weightings), and segregating responses by question. For experimental validation and system evaluation, the dataset was divided into validation and test sets. The validation set comprised 12 out of 16 questions for parameter tuning and model selection, while the test set included the remaining 4 questions for final performance evaluation.

B. Embeddings

The quality of text embeddings was vital to the performance of our semi-automated grading system, as they represented the semantic meaning of student responses. Given the nuanced and contextual nature of free-text answers, generating high-quality embeddings was challenging. To address this, we used Large Language Models (LLMs), specifically BERT (Bidirectional Encoder Representations from Transformers) and GPT-2 (Generative Pre-trained Transformer 2), both of which have demonstrated state-of-the-art performance in generating text embeddings.

We chose BERT for its bidirectional encoding, which captures contextual meaning more effectively than unidirectional models. BERT embeddings encode text semantically, outperforming other techniques in semantic similarity tasks [11], [40]. We used the base BERT model (110 million parameters) for its balance between complexity and performance, fitting for our grading system.

Additionally, we employed GPT-2, known for its strong semantic understanding [12], [41]. Unlike BERT's bidirectional approach, GPT-2's autoregressive architecture may capture broader semantic features. We used the GPT-2 small model (124 million parameters) to explore its embedding effectiveness. Comparing the embeddings from both models provided insights into the bidirectional and autoregressive approaches.

We addressed two research questions: (1) Which LLM—BERT or GPT-2—better captured the semantic meaning of student responses? (2) Would fine-tuning these models on our DSA corpus improve clustering by yielding higher-quality embeddings?

To explore these, we fine-tuned BERT and GPT-2 using Masked Language Modeling (MLM) and Next Token Prediction (NTP) on the DSA corpus. We compared embeddings from both base and fine-tuned models, retrieving student response embeddings from four sets per question in the validation dataset, which included 12 questions. Fine-tuning occurred over 50 epochs with early stopping. Details of the training and evaluation are in Appendix B.

To assess fine-tuning's impact, we performed statistical tests and analyzed variance in ground truth marks across clusters, evaluating whether fine-tuned embeddings improved clustering and the grading system's overall performance (see Fig. 2).

C. Clustering

Clustering plays a crucial role in our pipeline, as shown in Figures 1 and 2, where homogeneous clusters are desired to group student responses with similar marks. To optimize this

TABLE II
SAMPLE ENTRIES FROM THE IDSA DATASET

Assessment Type	Answer	Mark
Quiz3 Q1 (2021)	Complexity- O(n). Adding items to the top of the stack without removing items from the stack	0
Test Q1 (2022)	I would set a temp pointer to point to the head. Then, I would traverse through the list, checking if the next−>next node (the second node from temp) is not equal to the tail pointer. Once temp−>next equals the tail pointer, I would return temp. Since I only used one while loop, the time complexity would be O(n).	60
Exam Q2 (2022)	1) Linear search 2) When the needle integer is at the end of the haystack vector. O(n) 3) When the needle integer is at the front of the haystack vector. O(1)	100

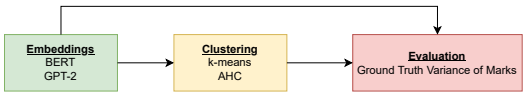


Fig. 2. Pipeline from Embedding Generation to Clustering and Ground Truth Evaluation

aspect, we employed two clustering techniques: k-means and Agglomerative Hierarchical Clustering (AHC).

K-means was chosen for its computational efficiency and simplicity, making it ideal for clustering the large volume of embeddings generated by LLMs. It quickly partitioned responses into clusters, revealing distinct patterns in student understanding. The centroid of each cluster provided a clear and interpretable prototype of the responses within that group. In contrast, AHC was used to capture semantic similarities between responses, grouping them based on underlying concepts. Unlike k-means, AHC does not require the number of clusters to be predetermined, offering greater flexibility. Its hierarchical structure also facilitated targeted manual sampling, allowing us to select representative responses from each cluster for further evaluation. For both clustering techniques, we utilized the cosine distance as our distance measure, which is effective for high-dimensional embeddings [1], [9]. For AHC, we also tested the performance of single and complete linkages to assess the best configuration.

To evaluate the performance of the embeddings and the clustering algorithms, we used the average ground truth variance across clusters. We ran both clustering methods for different numbers of clusters and computed the variance of ground truth marks in terms of percentage (0 to 100) within each cluster. The variance was then normalized to a scale of 0 to 100 by taking the standard deviation of the values, providing a clear metric for comparing the performance of the algorithms. This approach allowed us to assess both the quality of the embeddings and the effectiveness of the clustering in improving the grading process.

D. Sampling, Marking and Confidence

The main contribution of our work lies in the sampling and confidence strategy within the semi-automated grading pipeline. After embedding generation and clustering, we aimed for homogeneous clusters, yet manual marking remained necessary. Several critical questions arose: how to select which submissions to mark, how to assign marks to unmarked submissions, and how to trust the auto-graded marks. The following subsections address these concerns.

1) Mean of Cluster (MoC): To assign marks to unmarked submissions, we introduced the Mean of Cluster (MoC) strategy. In this approach, auto-assigned marks were based on the mean marks of the manually marked submissions within each cluster. If a cluster had no marked submissions, we used the k-nearest neighbors algorithm to assign marks, based on the assumption that k=3 provides a balanced approach and does not overfit for small datasets [42]. The sampling strategies were then compared based on this MoC mark allocation approach.

2) Random Sampling: Random sampling served as the baseline for our study. This trivial approach involved selecting submissions at random from any cluster and marking them. We iteratively assessed the accuracy of the auto-graded marks after each manual marking to measure how mark correlation accuracy evolved over time.

3) Greedy Monte Carlo Sampling: Greedy Monte Carlo (GMC) Sampling aims to improve upon the baseline (random) by incorporating Monte Carlo methods. It works as follows. Initially, a subset of submissions M from total submissions T is marked from each cluster by taking the x closest points from the centroids of each cluster. Then, Monte Carlo sampling is applied to this subset M to assign marks to unmarked submissions U in each cluster. The Monte Carlo approach runs for n iterations. In each iteration, we use a sampling rate controlling the percentage of the initial subset m from M used in each iteration. In this way, the initial subset varies each time, and we use different marked submissions in each iteration to allocate marks to the unmarked set of submissions U. Once the marks are allocated to unmarked submissions in U, we evaluate the Pearson's r and QWK correlation metrics. We allocate the marks using the MoC strategy discussed. After n Monte Carlo iterations, the marks for unmarked answers U vary as each submission in U will have n different marks, and so we can measure the mean and variance of each submission based on the n different marks. We measure the variance of marks v and take the standard deviation of it to get a variance value between 0 and 100 for each submission. Using this approach, We define confidence as $c = 100 - v$, with higher variance indicating lower confidence.

Once the confidence scores are calculated, we employ a greedy strategy to mark additional submissions which allow a user to mark more submissions if they are unhappy with the confidence in the auto marked submissions. We iteratively select the next submission y from U that has the lowest confidence c for marking and mark it, which will gives us M'. We run the Monte Carlo approach again on M' , thereby reducing

variance and increasing confidence in the overall grading incrementally by iterating the process until the marker is happy with c (confidence value) . In this way, GMC sampling can be employed to estimate the uncertainty or confidence in the assigned grades. By repeatedly sampling from the underlying probability distributions of student responses, the system can generate multiple possible grade assignments and analyze the variability or spread of these results. The effectiveness of this approach is illustrated in Fig. 3, and in addition to correlation metrics, we tracked the lowest confident answer after each marking step. This metric c for confidence is crucial for assessing the quality of the semi-automated system when ground truth labels are unavailable, enhancing interpretability.

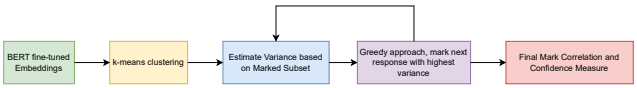


Fig. 3. Illustration of the semi-automated grading system incorporating Greedy Monte Carlo Sampling for iterative marking and confidence estimation

V. Discussion of Results

This section presents an analysis of the experiments conducted and the outcomes observed, highlighting key findings and their implications.

A. Evaluation of Embeddings with Ground Truth Variance of Marks

As shown in Fig. 2, we utilized the ground truth mark labels to evaluate the first two components of our semi-automated grading system: Embeddings and Clustering. We assessed the average ground truth variance across different cluster numbers in the validation set of questions. Fig. 4 displays the average ground truth variance (in percentage) for the question Exam2020-Q1, using k-means clustering. The average variance patterns were generally consistent across most questions in the validation set. However, some exceptions were observed, as the technical complexity of the questions influenced the ground truth variance differently. The complete set of graphs can be found in Appendix C.

Figure 4 and the full set of questions in Appendix C show that BERT embeddings consistently outperform GPT-2 embeddings, regardless of whether the models are fine-tuned. This trend is clearly evident across most of the question graphs. We attribute this to BERT's bidirectional encoding, which likely allows it to capture semantic relationships more effectively than GPT-2.

To assess the impact of fine-tuning on both models, we observed that GPT-2 embeddings generally performed better after fine-tuning on most questions. However, the effect of fine-tuning on BERT was less consistent. In some cases, BERT fine-tuned embeddings showed no significant improvement over the base BERT model. Interestingly, GPT-2 fine-tuned embeddings exhibited more erratic behavior—sometimes outperforming the base GPT-2 embeddings, while other times performing worse, as illustrated in Figure 5. This unpredictable behavior was observed on a small subset of questions, and

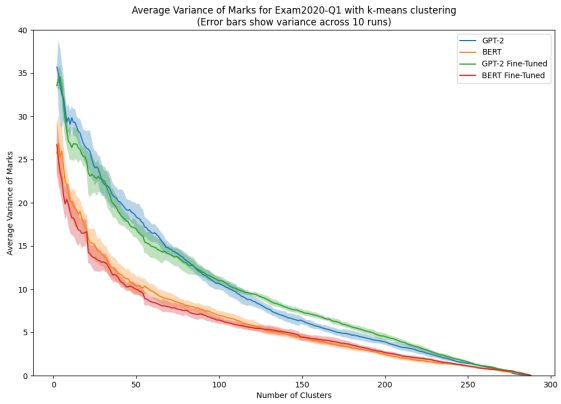


Fig. 4. Figure showing the average ground truth variance of marks for Exam 2020 Question 1 using k-means clustering, with comparisons across different sets of embeddings

we recommend that future work further investigate the causes of these inconsistencies. Overall, the most common pattern was that GPT-2 performed better than its fine-tuned version, and BERT fine-tuned embeddings either outperformed or performed similarly to the base BERT model. To evaluate the statistical significance of these observations, we conducted statistical tests on the performance differences between the models

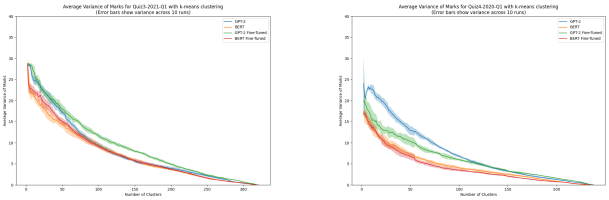


Fig. 5. Comparison of GPT-2 performance with k-means: GPT-2 outperforming GPT-2 Fine-tuned (Left) and GPT-2 Fine-tuned outperforming GPT-2 (Right)

Our statistical analysis reveals notable differences in the impact of fine-tuning between BERT and GPT-2 models. The results in Table III demonstrate that fine-tuning GPT-2 produces a statistically significant improvement in performance ($t = 10.931$, $p < 0.0001$), with a moderate negative effect size (Cohen's $d = -0.269$). This finding is further corroborated by the Wilcoxon signed-rank test ($S = 236709.0$, $p < 0.0001$), suggesting a robust and consistent pattern across both parametric and non-parametric analyses. In contrast, while BERT's Fine-Tuned (FT) version showed some improvement over its Base (B) model, the difference was marginally significant in the t-test ($t = -1.695$, $p = 0.0904$) with a minimal effect size ($d = 0.034$), and not significant in the Wilcoxon test ($S = 322460.0$, $p = 0.2718$).

These findings suggest that while both models benefit from fine-tuning, GPT-2 demonstrates a more pronounced and statistically reliable improvement compared to BERT. The smaller effect size for BERT ($d = 0.034$) versus GPT-2 ($d = -0.269$) quantifies this differential impact, indicating that

TABLE III
STATISTICAL TEST RESULTS FOR BERT AND GPT-2

Test	Comparison	Statistic	p-value	Cohen's d
t-test	BERT Base v FT	$t = -1.695$	$p = 0.0904$	$d = 0.034$
	GPT-2 Base v FT	$t = 10.931$	$p < 0.0001$	$d = -0.269$
Wilcoxon	BERT	S = 322460	$p = 0.2718$	
	GPT-2	S = 236709	$p < 0.0001$	

fine-tuning strategies may be particularly effective for GPT-2's architecture. However, it's worth noting that BERT's marginal improvement still suggests potential benefits from fine-tuning, albeit to a lesser extent than GPT-2. This pattern of results aligns with recent literature suggesting that different transformer architectures may respond differently to fine-tuning procedures, with some architectures showing more drastic improvements than others. Consequently, we concluded that, despite the marginal improvement from fine-tuning, BERT embeddings consistently outperformed GPT-2 embeddings in terms of average ground truth variance, whether fine-tuned or not. While fine-tuning BERT did not lead to significant gains, it did not negatively impact performance either, and given the availability of compute resources, we selected the fine-tuned BERT model for the embeddings in the pipeline.

B. Clustering Performance

We conducted experiments with two clustering techniques—k-means and Agglomerative Hierarchical Clustering (AHC)—to assess their performance in clustering student submissions based on ground truth variance in marks. The embeddings used for clustering were generated from four different sets of embeddings for each question: BERT, GPT-2, BERT fine-tuned and GPT-2 fine-tuned. For AHC, we tested two different linkage types: complete and average.

As depicted in Figure 6, our initial analysis revealed that AHC, particularly with the average linkage, seemed to outperform k-means clustering. However, this result was somewhat misleading. Upon closer inspection, we found that the AHC with average linkage tended to create many sparse clusters with very few submissions, typically containing only one to three answers. These small clusters had lower variance in marks, leading to an overall reduction in the average ground truth variance. Conversely, fewer larger clusters contained most of the submissions, further skewing the variance estimates.

To address this, we experimented with increasing the number of clusters, but the issue persisted. The sparse clusters continued to have a disproportionately low variance, and the larger clusters still contained many submissions, which led to an overestimation of the clustering ability of the algorithm.

We then tested AHC with complete linkage, which generates more balanced and evenly distributed clusters. This modification improved the clustering results, yielding a more realistic estimate of ground truth variance, as seen in Figure 6. Despite the improvements, the issue of sparse clusters persisted, and the results began to resemble those of the k-means algorithm. The k-means algorithm, known for its ability to generate evenly distributed clusters, provided a more reliable measure of the ground truth variance than AHC with either linkage type.

Based on these observations, we selected the k-means algorithm as our primary clustering method. It produced well-distributed clusters and provided a more accurate representation of the ground truth variance of marks. While we identified that hierarchical clustering could potentially be improved—especially by further decomposing large clusters into smaller, more meaningful subclusters—this approach falls outside the scope of our current study. Therefore, we focused on the k-means algorithm, which proved to be a robust and effective method for clustering student submissions.

C. Impact of Question Variability on Model and Clustering Performance

An interesting observation from our analysis is that the performance of the LLMs and clustering techniques was significantly influenced by the specific characteristics of the questions. In particular, the diversity in the Data Structures and Algorithms (DSA) content, such as the inclusion of specialized terminology, code blocks, and varying levels of complexity, appeared to affect both the embeddings generated by the models and the resulting clusters. This, in turn, influenced the ground truth variance of marks.

For example, as shown in Figure 7, GPT-2 fine-tuned performed better than BERT fine-tuned in a particular instance. However, this was an exception, as the performance differences between the two models varied across questions. The structure of the questions—including the use of technical terms, the presence of code snippets, and the depth of explanation required—likely played a role in this variability. Questions with more DSA jargon, or those that demanded more complex or specific answers, could lead to differences in how the models interpret and cluster the submissions. This suggests that the underlying structure and content of the questions themselves may affect the efficacy of the semi-automated grading system, and warrants further investigation to better understand how these factors contribute to model performance.

To further explore this variability, we visualized the clustering results for different questions, as shown in Figure 8, using the BERT fine-tuned model with k-means clustering. It is evident that the ground truth variance varies across questions, which presents a challenge for our semi-automated grading system's ability to generalize effectively. For example, Question 3 from Exam 2022 exhibited a higher average ground truth variance compared to Question 1 from Quiz 4 in 2020. Upon further analysis, we identified that part of this difference could be attributed to the mathematical notation used in the answers. However, this explanation is not definitive, and further investigation is needed. We hypothesize that this variability may stem from the specific training data the LLMs were pre-trained on. For instance, GPT-2 might be better suited to interpret the technical language present in some questions, leading to more precise embeddings and a lower average ground truth variance for those questions. Similarly, BERT's

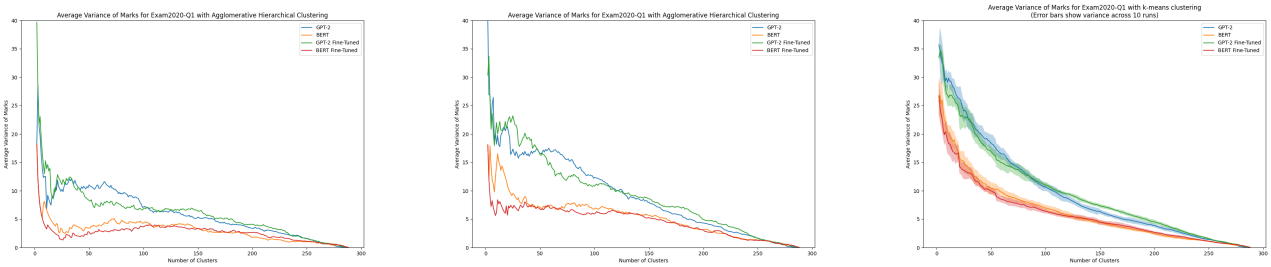


Fig. 6. The performance (in terms of average ground truth variance) of clustering algorithms: AHC average linkage (left), AHC complete linkage (center) and k-means (right)

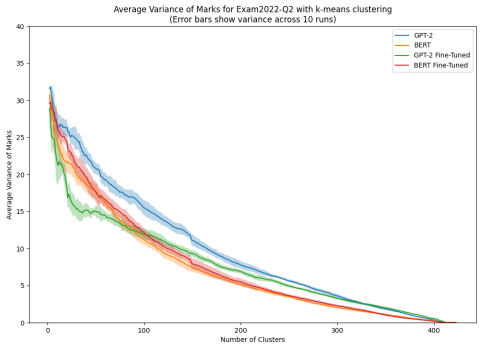


Fig. 7. Ground truth variance for an Exam 2022 Question, where GPT-2 fine-tuned outperforms BERT fine-tuned, highlighting an exception in performance

performance could be influenced by its exposure to certain types of content during its initial pre-training phase.

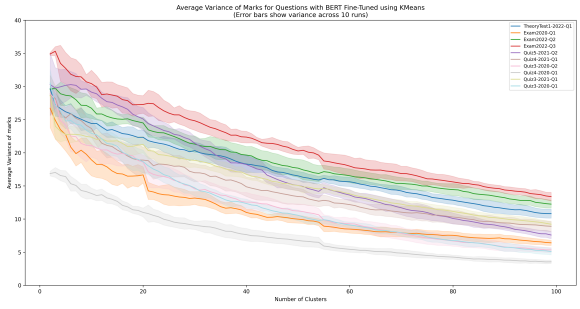


Fig. 8. Clustering results for various questions using the BERT fine-tuned model with k-means clustering, highlighting the ground truth variance across different question types

In conclusion, the diversity of DSA content across questions highlights the need for a deeper understanding of how the question structure, content intensity, and technical language impact the performance of fine-tuned models in a semi-automated grading system. Further research is needed to explore how these factors influence the generalization of clustering and grading performance, and to develop strategies to account for such variability.

D. Evaluation of Greedy Monte Carlo vs. Random Sampling Methods

For the GMC approach, we fine-tuned several hyperparameters, including the number of Monte Carlo iterations, the initial marking subset size, and the sampling rate for unmarked submissions. For GMC, we set the number of iterations to 1000, the initial marking subset size to 50, and the sampling rate to 0.25, as we considered these values to be sensible starting points. A critical parameter common to both GMC and random sampling was the number of clusters. To select the optimal number, we experimented with cluster sizes of 20, 30, 50, 80, 100 on three validation questions. Clusters beyond 100 were ineffective as they led to over-segmentation, where many clusters contained only one submission. Both GMC and random sampling showed similar results with 20 to 30 clusters, so we chose 30 clusters as the optimal number of clusters.

For GMC, we also tuned the number of iterations, initial subset size, and sampling rate. For the number of iterations, we understand that as more submissions are marked in the greedy process, the number of iterations required would be less, so we aimed to determine the upper bound of the number of iterations to ensure sufficient accuracy without unnecessary computation. After testing iteration counts on two IDSA questions, we observed that the largest changes in mark variance occurred within the first 200 iterations, so we set the number of iterations to 300 to balance accuracy and computational efficiency. For the initial marked subset size, we experimented with 50, 90, 140, and 190 initial subset sizes and found that all sizes performed comparably. For one question, larger initial subsets showed slightly better results, but marking 190 submissions would not be meaningful for our system, as we aim to create a more robust and efficient grading process. The sizes of 50 and 90 performed similarly, and thus we intended to go with 3 samples per cluster. However, since some clusters were sparse and contained fewer submissions, we mostly ended up with an initial subset size of 50. For the sampling rate, we tested values ranging from 0.25 to 0.9 on two questions and found that a rate of 0.75 produced the best results. Appendix D contains more details on hyperparameter tuning. It is worth noting that these hyperparameters were selected sequentially, and there may be more optimal combinations that could be found through techniques like Bayesian optimization or grid search. Future work should explore these approaches to further refine the GMC configuration.

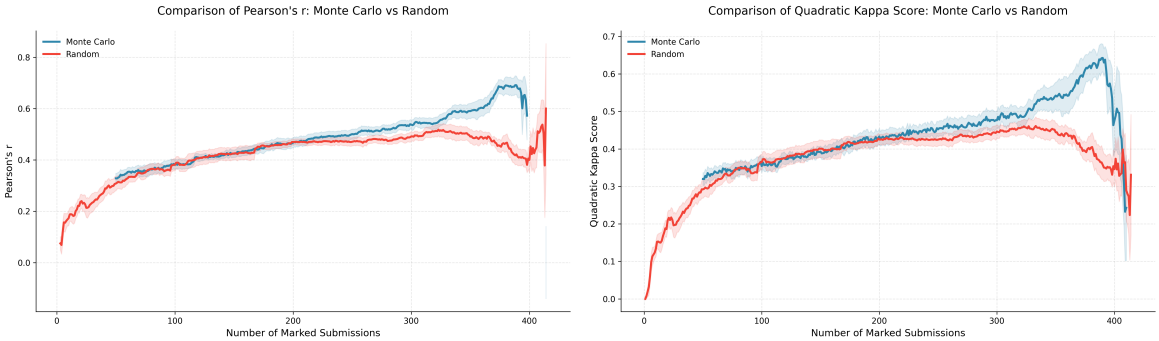


Fig. 9. Performance comparison (over 10 runs) of the GMC and random sampling techniques for Exam 2022, Question 1 from the test set, showing Pearson's r (left) and QWK (right) metrics

We compared the accuracy of our proposed Monte Carlo Greedy (GMC) sampling strategy with the baseline random sampling approach to assess their performance. As shown in Figure 9, the GMC strategy outperformed random sampling, as evidenced by higher Pearson's r and QWK scores. Initially, both strategies performed similarly, but as more submissions were manually marked, GMC diverged and demonstrated significantly better performance. For example, at around 375 manually marked submissions, GMC achieved a Pearson's r of approximately 0.7 and a QWK of 0.6, while the random sampling approach peaked at a Pearson's r of 0.5 and a QWK of 0.45. This pattern was consistent across different questions from the IDSA dataset, with additional graphs available in Appendix D. This trend of increasing Pearson's r and QWK aligns with expectations, as correlation metrics naturally improve with more marked submissions. However, it is important to note that correlation measurements for GMC were only taken after the initial subset (approximately 50 samples) had been marked.

We observed that on certain questions, particularly in the minority, GMC performed similarly to the random sampling approach. Figure 10 illustrates this behavior, where GMC yielded comparable Pearson's r values to random sampling but slightly underperformed in QWK. This finding highlights that the diversity and content type of questions significantly influence the effectiveness of the GMC strategy. Interestingly, for these questions, random sampling also delivered positive results, with Pearson's r approaching 0.7 and QWK nearing 0.7 as we marked around 200 submissions. In comparison, GMC achieved a Pearson's r of 0.7 and a QWK of 0.6, making the difference between the two strategies relatively small but still noteworthy. In conclusion, while GMC performs on par or slightly better than random sampling, it is particularly advantageous in certain non-trivial question contexts where random sampling alone may not yield optimal results.

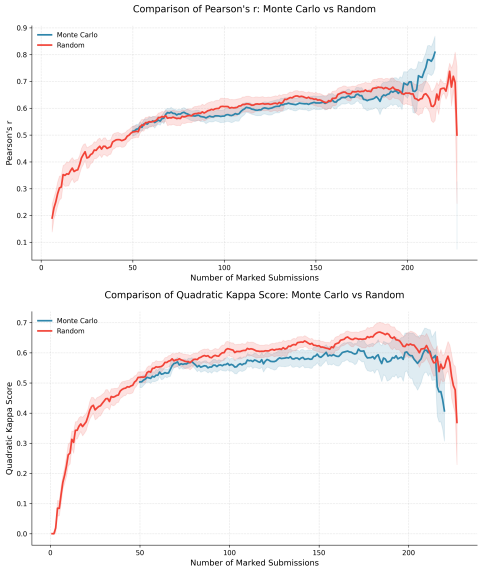


Fig. 10. Performance comparison (over 10 runs) of the GMC and random sampling techniques for a quiz question where GMC performed comparably to random sampling, with Pearson's r shown at the top and QWK at the bottom

E. Confidence and Variance

In contrast to the random sampling approach, our GMC approach not only performs as well as or better than random sampling but also incorporates a confidence estimation for the grades assigned. This feature is exemplified in Fig. 11, which illustrates how confidence evolves as more submissions are manually marked. Alongside confidence, we track the standard deviation of the error to observe the variability in confidence values over time.

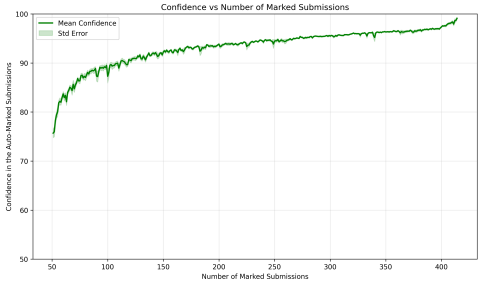


Fig. 11. Confidence measure graph generated by GMC for an exam question (measures lowest confidence after each marked submission)

The increasing confidence curve aligns with expectations, as confidence in unmarked submissions is anticipated to grow as more student answers are manually graded. This behavior

highlights a key feature of our GMC approach: prioritizing submissions with the lowest confidence or highest variance for marking. By addressing these outliers first, the variance of other submissions in the corresponding cluster is reduced, as Monte Carlo sampling iteratively updates their confidence based on the new marks. This dynamic is clearly visible in Fig. 11, (measures lowest confidence after each marked submission) where the lowest confidence in unmarked submissions improves steadily as more answers are graded.

This confidence-based marking strategy offers significant advantages, particularly for educators, who can decide whether to continue grading or rely on the system after marking a subset of submissions. However, a potential limitation of the confidence metric lies in its definition, $c = 100 - v$, where v is the variance. This formulation may inadvertently overestimate confidence, as evidenced by Fig. 8, where the average variance for any question rarely exceeds 35. Consequently, the system might report a confidence of at least 65%, for all questions, which may not always provide a fully accurate representation of certainty. Future work should focus on refining this metric to develop a more robust and realistic measure of confidence.

Despite this limitation, the current confidence metric serves its intended purpose effectively. It provides a reliable heuristic for identifying the next unmarked submission to prioritize, contributing to the efficiency and reliability of the semi-automated grading system.

F. Limitations of our semi-automated grading system

Our semi-automated grading system has several limitations that should be considered when interpreting its results. First, the use of transformer-based models for generating semantic embeddings introduces challenges in interpretability. As these models are inherently complex, the system does not offer full transparency in how grading decisions are made, limiting its ability to explain the rationale behind specific marks. Additionally, the system is currently designed to assess answers only related to data structures and algorithms (DSA), and it has been tested on a specific dataset. This narrow focus may limit the generalizability of the grading system, as the dataset used may not adequately represent the diversity of real-world student responses across all DSA topics. Another limitation is the system's sensitivity to question content variability. As discussed in the results section, the performance of the system can be influenced by variations in question content, posing challenges for its ability to generalize to questions with different structures or complexities. Furthermore, the confidence measure used in our system may be artificially inflated due to the formula applied for its calculation. This could result in confidence levels that do not accurately reflect the true reliability of the grading decisions. Lastly, our system's evaluation is based on a limited set of large language models (LLMs) and clustering techniques. Given the rapid advancements in the field, newer models and alternative clustering approaches may outperform the ones used in our study, suggesting that there are opportunities for improvement and further exploration.

VI. Conclusion and Future Work

This paper presented a semi-automated grading system for Data Structures and Algorithms (DSA) assessments, leveraging Large Language Models, clustering techniques, and a novel Greedy Monte Carlo sampling approach. Our findings demonstrate that BERT embeddings consistently outperformed GPT-2, with fine-tuning showing marginal improvements for BERT but significant enhancements for GPT-2. The k-means clustering algorithm proved more reliable than Agglomerative Hierarchical Clustering, providing well-distributed clusters and more accurate representations of ground truth variance.

The proposed Greedy Monte Carlo sampling strategy generally outperformed random sampling, achieving higher Pearson's r and QWK scores, while also providing valuable confidence metrics for unmarked submissions. However, the system's performance showed sensitivity to question content variability, highlighting the importance of question structure and complexity in automated assessment systems.

Future research directions could significantly enhance the system's capabilities and address its current limitations. The exploration of advanced clustering techniques, such as Gaussian Mixture Models and Enhancement of Clustering by Iterative Classification (ECIC), along with the integration of state-of-the-art pretrained models like LLaMA-3, could potentially improve clustering accuracy and overall system performance. Additionally, addressing the system's sensitivity to question content variability remains a crucial area for investigation, as this would enhance the system's ability to maintain consistent performance across different question types and complexities. Further optimization could be achieved through guided clustering at a cluster level, particularly by decomposing high-variance clusters into smaller, more meaningful subclusters and developing improved methods for estimating cluster variance. The refinement of confidence metrics to provide more accurate representations of grading certainty would also improve the system's practical utility for educators. These improvements could lead to a more robust and generalizable semi-automated grading system, potentially extending its application beyond DSA assessments to other educational domains.

References

[1] R. Klein, A. Kyrilov, and M. Tokman, "Automated assessment of short free-text responses in computer science using latent semantic analysis," in *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*, 2011, pp. 158–162.

[2] S. Valenti, F. Neri, and A. Cucchiarelli, "An overview of current research on automated essay grading," *Journal of Information Technology Education: Research*, vol. 2, no. 1, pp. 319–330, 2003.

[3] O. Mason and I. Grove-Stephensen, "Automated free text marking with paperless school," 2002.

[4] A. Ahmed, A. Joorabchi, and M. J. Hayes, "On deep learning approaches to automated assessment: Strategies for short answer grading." *CSEDU (2)*, pp. 85–94, 2022.

[5] L. Camus and A. Filighera, "Investigating transformers for automatic short answer grading," in *Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020, Proceedings, Part II 21*. Springer, 2020, pp. 43–48.

[6] N. Kaur and K. Jyoti, "Automated assessment of short one-line free-text responses in computer science," *International Journal of Computer Science and Informatics*, vol. 2, no. 1, pp. 105–109, 2012.

[7] X. Bai and M. Stede, "A survey of current machine learning approaches to student free-text evaluation for intelligent tutoring," *International Journal of Artificial Intelligence in Education*, vol. 33, no. 4, pp. 992–1030, 2023.

[8] D. Ramesh and S. K. Sanampudi, "An automated essay scoring systems: a systematic literature review," *Artificial Intelligence Review*, vol. 55, no. 3, pp. 2495–2527, 2022.

[9] S. Philip, "Partially automated grading of short free-text responses in computer science through sentence embedding and clustering," Ph.D. dissertation, University of the Witwatersrand, Johannesburg, 2023.

[10] N. Süzen, A. N. Gorban, J. Levesley, and E. M. Mirkes, "Automatic short answer grading and feedback using text mining methods," *Procedia Computer Science*, vol. 169, pp. 726–743, 2020.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[12] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[13] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*, 1967.

[14] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[15] D. M. Christopher, R. Prabhakar, S. Hinrich *et al.*, "Introduction to information retrieval," *An Introduction To Information Retrieval*, vol. 151, no. 177, p. 5, 2008.

[16] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding, and C.-T. Lin, "A review of clustering techniques and developments," *Neurocomputing*, vol. 267, pp. 664–681, 2017.

[17] D. P. Kroese, T. Brereton, T. Taimre, and Z. I. Botev, "Why the monte carlo method is so important today," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 6, no. 6, pp. 386–392, 2014.

[18] T. K. Kim, "T test as a parametric statistic," *Korean journal of anesthesiology*, vol. 68, no. 6, pp. 540–546, 2015.

[19] T. Neideen and K. Brasel, "Understanding statistical tests," *Journal of surgical education*, vol. 64, no. 2, pp. 93–96, 2007.

[20] F. Wilcoxon, S. Katti, R. A. Wilcox *et al.*, "Critical values and probability levels for the wilcoxon rank sum test and the wilcoxon signed rank test," *Selected tables in mathematical statistics*, vol. 1, pp. 171–259, 1970.

[21] S. Taheri and G. Hesamian, "A generalization of the wilcoxon signed-rank test and its applications," *Statistical Papers*, vol. 54, pp. 457–470, 2013.

[22] L. A. Becker, "Effect size (es)," 2000.

[23] F. Dong, Y. Zhang, and J. Yang, "Attention-based recurrent convolutional neural network for automatic essay scoring," in *Proceedings of the 21st conference on computational natural language learning (CoNLL 2017)*, 2017, pp. 153–162.

[24] B. Riordan, A. Horbach, A. Cahill, T. Zesch, and C. 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.

[25] C. Sung, T. Dhamecha, S. Saha, T. Ma, V. Reddy, and R. Arora, "Pre-training bert on domain resources for short answer grading," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 6071–6075.

[26] P. W. Bible and L. Moser, "An open guide to data structures and algorithms," 2023.

[27] C. A. Shaffer, "A practical introduction to data structures and algorithm analysis third edition (c++ version)," 2010.

[28] C. Fox, "Concise notes on data structures and algorithms," *James Madison University*, 2011.

[29] P. Morin, "Open data structures (in pseudocode) edition 0.1 gβ," *Ramuda Noto*, 2018.

[30] B. Miller and D. Ranum, "Problem solving with algorithms and data structures," *URL: https://www. cs. auckland. ac. nz/.../ProblemSolvingwith AlgorithmsandDataStructures. pdf (Last accessed: 30.03. 2018)*, 2013.

[31] A. Downey, *Think data structures: algorithms and information retrieval in Java*. " O'Reilly Media, Inc.", 2017.

[32] J. Erickson, *Algorithms*, 2023.

[33] Wikipedia, "Data structures and algorithms, sorting, linked lists, etc." https://en.wikipedia.org/wiki/Sorting, https://en.wikipedia.org/wiki/LinkedList, https://en.wikipedia.org/wiki/Algorithm, [Accessed: 28-Sep-2024].

[34] GeeksforGeeks, "Learn data structures and algorithms," https://www.geeksforgeeks.org/learn-data-structures-and-algorithms, [Accessed: 27-Sep-2024].

[35] MIT OpenCourseWare, "Introduction to algorithms," https://ocw.mit.edu, [Accessed: 27-Sep-2024].

[36] ——, "Introduction to computer science and programming in python," https://ocw.mit.edu, [Accessed: 27-Sep-2024].

[37] W3Schools, "Introduction to data structures and algorithms," https://www.w3schools.com, [Accessed: 29-Sep-2024].

[38] CS50, "Cs50," https://cs50.harvard.edu, [Accessed: 28-Sep-2024].

[39] freeCodeCamp, "Data structures and algorithms," https://www.freecodecamp.org, [Accessed: 30-Sep-2024].

[40] M. Beseiso and S. Alzahrani, "An empirical analysis of bert embedding for automated essay scoring," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 10, 2020.

[41] G. A. Katuka, A. Gain, and Y.-Y. Yu, "Investigating automatic scoring and feedback using large language models," *arXiv preprint arXiv:2405.00602*, 2024.

[42] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

APPENDIX A
STATISTICAL TESTS

A. t-test

The t-test is a parametric statistical test used to assess whether there is a significant difference between the means of two groups. In the context of model evaluation, it helps researchers determine if the performance difference between two models or two versions of the same model represents a genuine improvement or if it could have occurred by chance. The t-test assumes that the data follows a normal distribution and is particularly useful for comparing continuous measurements or scores.

The formula for the t-test is as follows:

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{1}$$

where:

- $\bar{X}_1$ and $\bar{X}_2$ are the sample means,
- s_1^2 and s_2^2 are the sample variances,
- n_1 and n_2 are the sample sizes.

A higher $|t|$ value indicates a greater likelihood that the observed difference between the groups is statistically significant. The significance level is determined by comparing the calculated t-value against a critical value from the t-distribution table based on the chosen confidence level (e.g., 95%) and degrees of freedom.

B. Wilcoxon Signed-Rank Test

The Wilcoxon signed-rank test serves as a non-parametric alternative to the t-test. Unlike the t-test, it does not assume a normal distribution of the data, making it more robust for cases where this assumption may not hold. The Wilcoxon test works by comparing the relative rankings of paired differences, making it particularly useful when working with scores that may not be normally distributed or when dealing with outliers.

The test statistic W is calculated as:

$$W = \sum \mathrm{rank}(|x_i - y_i|) \cdot \mathrm{sgn}(x_i - y_i) \tag{2}$$

where:

- x_i and y_i are paired observations,
- rank denotes the rank of the absolute differences,
- sgn is the sign function.

A higher or lower W value, depending on the distribution of ranks, indicates a statistically significant difference between the paired samples.

C. Effect Size: Cohen's d

To quantify the magnitude of differences, researchers often complement these statistical tests with effect size measurements such as Cohen's d. Effect size metrics help interpret the practical significance of a difference, rather than just its statistical significance. To assess the practical significance of these differences, effect size measures like Cohen's d are often used alongside statistical tests, providing context beyond statistical significance alone.

Cohen's d, in particular, expresses the standardized difference between two means, calculated as:

$$d = \frac{\bar{X}_1 - \bar{X}_2}{s_p} \tag{3}$$

where:

- $\bar{X}_1$ and $\bar{X}_2$ are the sample means,
- s_p is the pooled standard deviation, calculated by:

$$s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}} \tag{4}$$

Conventional thresholds for Cohen's d are as follows:

- Small effect: $d = 0.2$
- Medium effect: $d = 0.5$
- Large effect: $d = 0.8$

APPENDIX B
FINE-TUNING OF BERT AND GPT-2 MODELS

The BERT model (110M parameters) and the GPT-2 small model (124M parameters) were used to generate high-quality embeddings for the semi-automated grading pipeline. Both models were fine-tuned with early stopping criteria, halting training if the validation loss did not improve for three consecutive epochs. This resulted in the GPT-2 model running for 50 epochs and the BERT model for 48 epochs.

For BERT, a sequence length of 512 and a batch size of 16 were used, while GPT-2, with its larger sequence length of 1024, required a smaller batch size of 8. All hyperparameters were carefully optimized, including a learning rate of 2×10^{-5}, gradient accumulation steps of 8 for efficient training, 1000 warmup steps, and a weight decay of 0.1. The training and validation loss curves for both models are shown in Fig. 12.

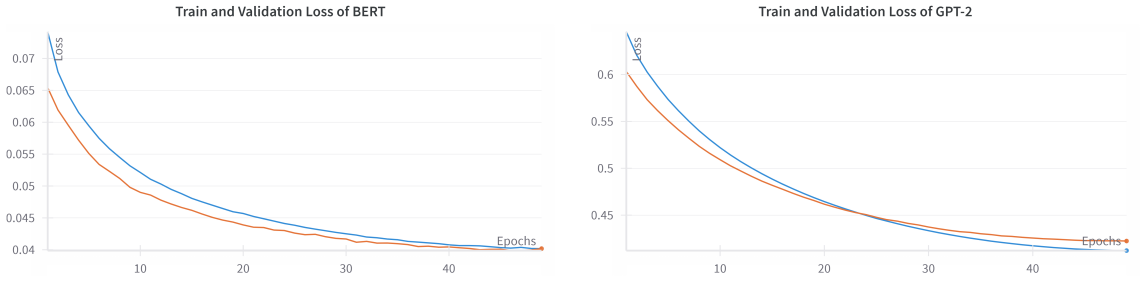


Fig. 12. Training and Validation Loss of BERT (Left) and GPT-2 (Right)

As shown in Fig. 12, both the training and validation losses consistently decreased as the number of epochs increased, indicating that the model successfully learned from the DSA corpus. Both losses appeared to converge with additional training, providing further evidence of the model's effectiveness.

Interestingly, the validation loss started lower than the training loss, which can be attributed to several factors:

- The relatively small batch size of 16 used during training, which may have introduced higher variance in training updates.
- Dropout and other regularization techniques active during training but disabled during validation.
- The validation set potentially containing more frequently occurring patterns, making it easier for the model to predict.

APPENDIX C
GROUND TRUTH VARIANCE EVALUATION

A. Graph sheet for Ground Truth Variance - k-means clustering

B. Graph sheet for Ground Truth Variance - AHC

APPENDIX D
HYPERPARAMETER TUNING AND FINAL RESULTS

A. Hyperparameter tuning graphs for the Monte Carlo Greedy sampling approach

B. Hyperparameter tuning graphs for the random sampling approach

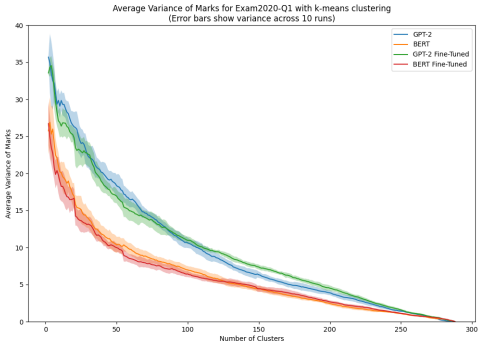
The hyperparameter tuning graphs are attached on the github page. Please find the remaining graphs and code on there: https://github.com/MZSFighters/auto-grading-of-computer-science-answers

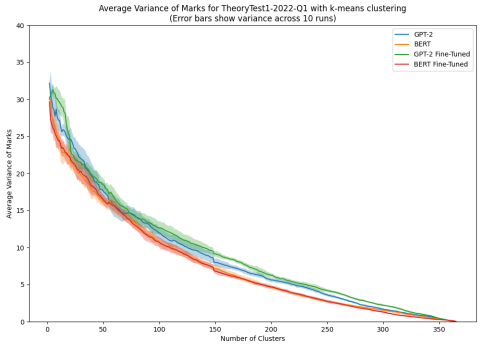
C. Additional Sample graphs of GMC vs Random sampling

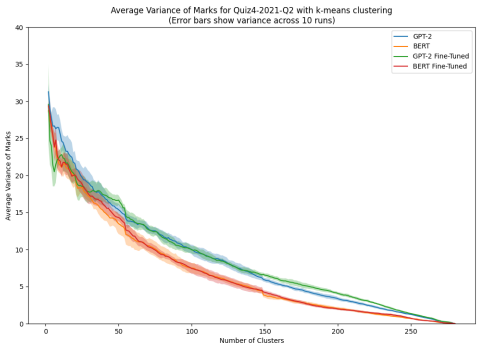
D. Graph sheet of Confidence and variance for different questions

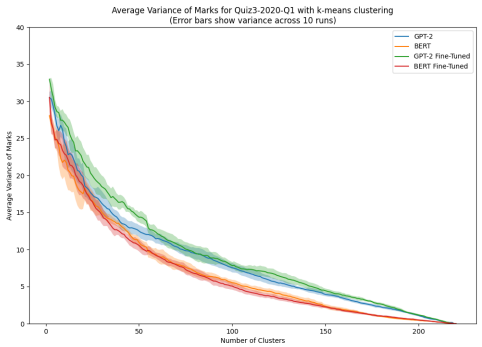
APPENDIX E
ETHICS WAIVER

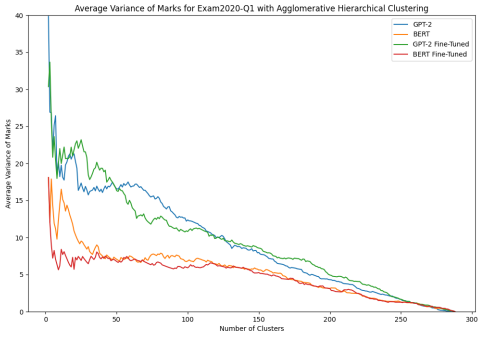
The Data Ethics Waiver number is WCSAM-2024-25.

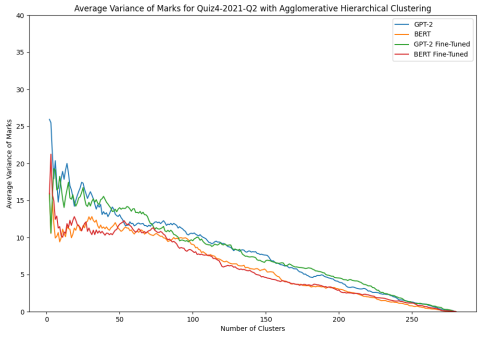

Average Variance of Marks for Exam2020-Q1 with k-means clustering
(Error bars show variance across 10 runs)
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters

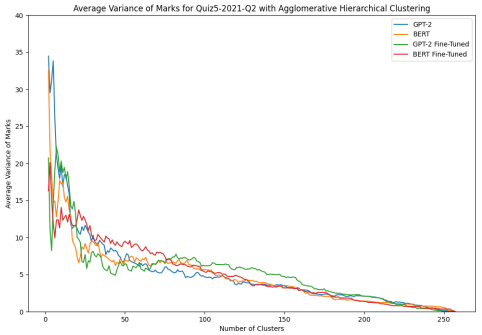

Average Variance of Marks for TheoryTest1-2022-Q1 with k-means clustering
(Error bars show variance across 10 runs)
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters

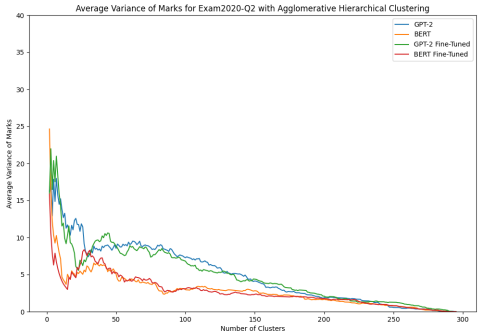

Average Variance of Marks for Quiz4-2021-Q2 with k-means clustering
(Error bars show variance across 10 runs)
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters

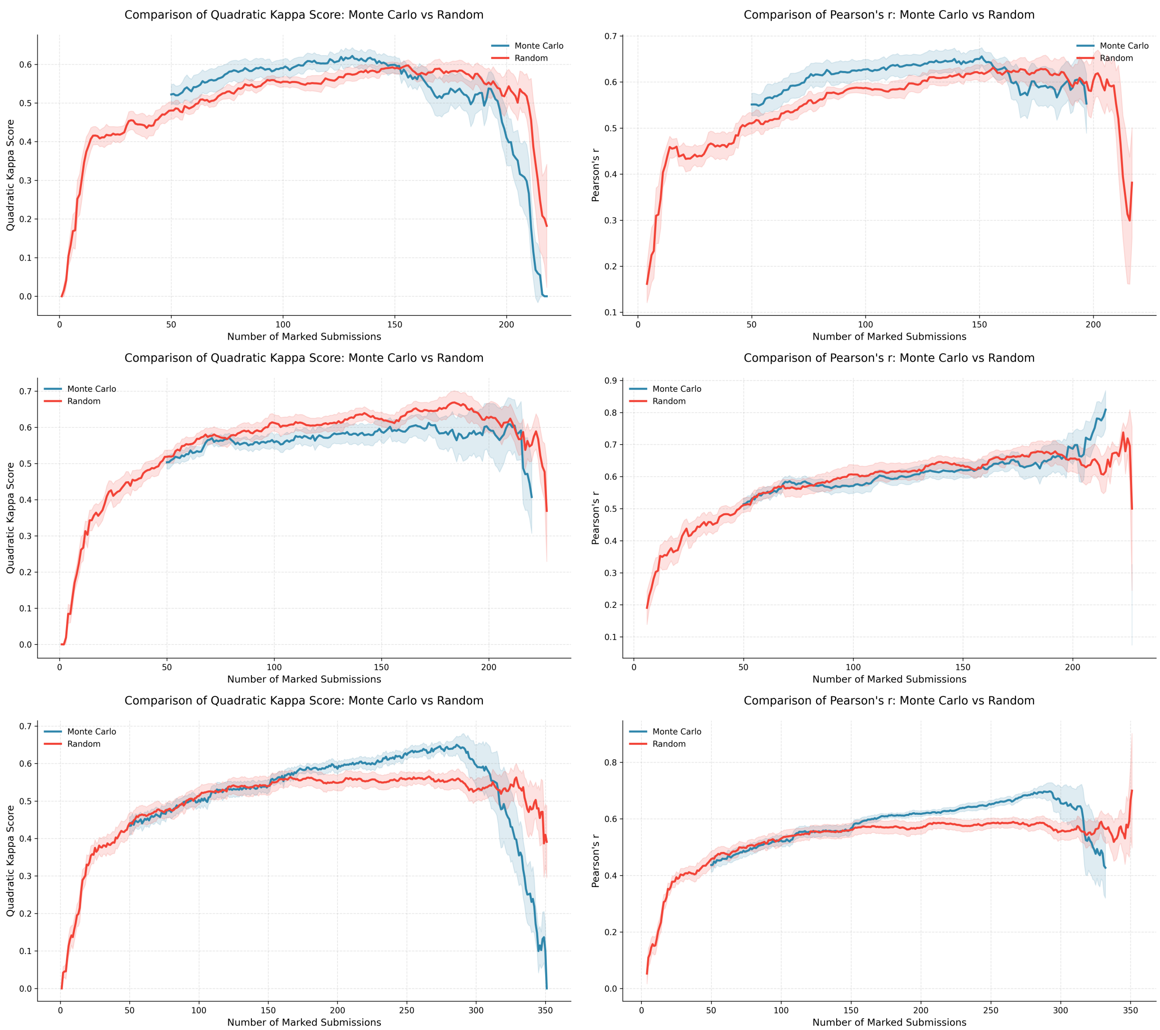

Average Variance of Marks for Quiz3-2020-Q1 with k-means clustering
(Error bars show variance across 10 runs)
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters

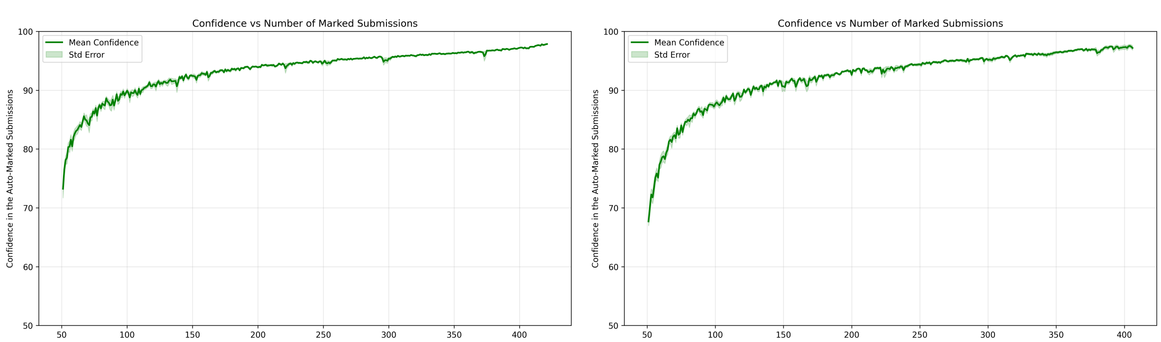

Average Variance of Marks for Exam2020-Q1 with Agglomerative Hierarchical Clustering
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters


Average Variance of Marks for Quiz4-2021-Q2 with Agglomerative Hierarchical Clustering
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters


Average Variance of Marks for Quiz5-2021-Q2 with Agglomerative Hierarchical Clustering
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters


Average Variance of Marks for Exam2020-Q2 with Agglomerative Hierarchical Clustering
GPT-2
BERT
GPT-2 Fine-Tuned
BERT Fine-Tuned
Average Variance of Marks
Number of Clusters


Comparison of Quadratic Kappa Score: Monte Carlo vs Random
Comparison of Pearson's r: Monte Carlo vs Random
Monte Carlo
Random
Quadratic Kappa Score
Pearson's r
Number of Marked Submissions


Confidence vs Number of Marked Submissions
Mean Confidence
Std Error
Confidence in the Auto-Marked Submissions
Number of Marked Submissions