BERT Goes to SQL School:
Improving Automatic Grading of SQL Statements

Pablo Rivas 🆔 , *Senior, IEEE*
School of Eng. & Computer Science
Dept. of Computer Science
Baylor University
Email: Pablo_Rivas@Baylor.edu

Donald R. Schwartz 🆔
School of Computer Science & Math.
Dept. of Computing Technology
Marist College
Email: Donald.Schwartz@Marist.edu

Ernesto Quevedo 🆔
School of Eng. & Computer Science
Dept. of Computer Science
Baylor University
Email: Ernesto_Quevedo1@Baylor.edu

Abstract—**Automated grading of SQL queries is a challenging task due to the complexity of the language and the variety of acceptable solutions for a given problem. In this paper, we propose a novel approach that leverages deep learning with a BERT model to understand the syntax and semantics of SQL statements. By training BERT on a dataset of SQL queries and their corresponding grades, we create a model that can automatically grade new questions accurately. Our experiments demonstrate that the proposed methodology achieves high accuracy and consistency in grading SQL queries, outperforming existing state-of-the-art models. Furthermore, we provide an analysis of the model's explainability, revealing a new capability that can be extremely beneficial for understanding the decision-making process. Overall, our work demonstrates the potential of deep learning with BERT for improving the efficiency and accuracy of SQL query grading.**

Index Terms—**bert, sql, natural language processing, machine learning, deep learning**

I. INTRODUCTION

As is true in almost every field of computer science, grading multiple sets of SQL queries can prove to be time-consuming and challenging. The tediousness of grading hundreds of queries can quickly become daunting. The fact that there are usually very many ways to construct correct queries provides another set of challenges. Awarding consistent partial credit to a wide variety of incorrect queries introduces another level of complexity. These challenges are significant when it comes to hand-grading assignments; they become even more so when trying to automate the process of grading these queries.

Automating the grading of SQL queries can save a lot of time and effort, but systems that do so often lack the capacity to award credit consistently across a variety of query types. These automated systems generally fall into one of two categories: static analysis systems and dynamic analysis systems [1].

The general approach for static analysis systems is to compare the structure of the submitted query to the structure of (one of) the answer-key queries. This means that the grading system does not actually run the submitted query. Instead, it analyzes the query itself. This approach requires that multiple answer-key queries be provided, because of the wide variety of possible correct answers. A submitted query would be compared to each answer-key query, generating many possible scores. The system would then award the highest of these scores to the submitted query.

Dynamic analysis systems, on the other hand, actually run the submitted queries against a variety of fixed data sets and compare the results to the answer-key results. Because only the results are being compared, this approach can suffer from the fact that it often cannot accurately determine whether a query is correct, since incorrect queries could produce what appear to be correct results. For example, consider the case where the answer is an empty table – there are infinitely many queries that produce an empty table. Dynamic analysis systems are very good at identifying incorrect queries, however.

Recent systems have combined static and dynamic analysis into a hybrid approach in an attempt to avoid the problems associated with each and to yield better results, especially in terms of awarding partial credit. Awarding partial credit is important, of course, but developing a system to do so in a consistent, equitable and meaningful way is quite challenging. Consistency is important so that partial credit is evenly awarded across many sets of responses. Partial credit must be equitable – an extra column in the results should likely result in a smaller penalty than a query with significant logical errors, but the correct number of columns in the answer. The amount of partial credit must also be meaningful, so that queries that are very close to correct earn the most partial credit possible.

Although automated grading systems are available, many lack the sophistication to accurately and consistently award credit for each query. This paper proposes using deep learning, specifically bidirectional encoder representations from transformers (BERT) [2], to model SQL statements to address this challenge. BERT is a state-of-the-art language model with outstanding performance in various natural language processing tasks [3]. By training BERT on a dataset of SQL queries and their corresponding grades, we can create a model that automatically grades new questions accurately. The proposed model takes advantage of BERT's pre-training on a large corpus of text, enabling it to learn the complex syntax and structure of SQL statements. Using BERT allows us to avoid the limitations of traditional rule-based systems [4] and achieve state-of-the-art performance in automatic SQL query grading. Therefore, leveraging deep learning with BERT in this area is necessary to improve the efficiency and accuracy of SQL query grading.

This paper aims to address the challenges above by proposing a machine learning-based approach for the automated grading

83

of SQL queries using the BERT model. The proposed approach leverages the power of BERT's contextualized word embeddings to model the syntax and semantics of SQL statements. The paper aims to demonstrate that this approach can achieve high accuracy and consistency in grading SQL queries while reducing the time and effort required for manual grading. The paper also compares the performance of the proposed BERT-based approach with existing state-of-the-art models based on a combination of self-attention mechanism and convolutional neural networks [5], [6], and show that the BERT-based approach outperforms these systems in terms of accuracy.

The rest of the paper is organized as follows. The related work is reviewed in Section II. Section III presents our methodology, including a transformer training description and a neural architecture layout. Section IV discusses an overview and evaluation of our experiments, including our evaluation metrics. Our results are analyzed in Section V, where we compare previous approaches and a baseline model. Conclusions are presented in Section IV.

II. RELATED WORK

A. Literature review of existing approaches for automated grading of SQL statements

An early example of the static analysis approach, described by Aho et al. [7], created a matrix based on various relational expressions that described the query, then used those expressions to construct a variety of equivalence classes. Another approach by Štajduhar et al. used string similarity metrics to compare submitted queries with answer-key queries [8]. The Cosette system, proposed by Chu et al., encoded queries into logical expressions in order to determine whether they were logically equivalent, using an unbound semiring to determine the level of equivalence. This approach allowed the system to provide more feedback to the student [9].

SQLator [10], SQLify [11], and AsseSQL [12] are frequently-cited early examples of the dynamic analysis approach. SQL Tester, a system developed by Kleerekoper et al. [13], does a record-by-record, case-sensitive comparison of the submitted query results with the correct results.

Chandra, et al., have proposed the XData system [14], a hybrid approach that uses dynamic analysis to identify incorrect answers, then uses static analysis to evaluate the SQL query to determine how closely it matches a correct query. They use edit-based grading to award partial credit based on the number of edits required to transform a student submitted solution into a correct solution (equivalent to one of the "answer key" solutions). Wang, et al., also describe a hybrid system that takes a similar approach [1].

B. Overview of deep learning approaches in the context of automated grading

Deep learning has become an increasingly popular approach in automated grading, as it offers the potential for improved accuracy and consistency of grading [15]–[18]. These approaches can help eliminate the burden of grading significant test questions and facilitate performing even more

assessments during lectures, especially when the number of students is large. Deep learning models have been used for generating paragraph embeddings, which are used for short answer automatic scoring [15]. The choice of paragraph embedding model influences accuracy in the task of automated scoring. A hybrid approach that optimizes a deep learning technique called LSTM (Long Short Term Memory) with a recent optimization algorithm called a Grey Wolf Optimizer (GWO) has been proposed for grading short-answer questions automatically [16]. The proposed system is optimized to avoid the problem of overfitting in forecasting the students' scores to improve the learning process and save instructors' time and effort. Deep learning architectures with a combination of Convolutional Neural Network (CNN) and Bidirectional Long Short Term Memory (BiLSTM) have been proposed for automated evaluation of handwritten answer scripts [17]. In particular, the following three recent papers showcase the use of more advanced and sophisticated deep learning in this context.

In 2019, Sung et al. [19] focused on pre-training BERT on domain-specific resources for short answer grading. The authors use domain-specific resources, such as textbooks or curated datasets, to enhance the pre-training process of BERT. The pre-trained BERT model is then fine-tuned on a dataset of short answers and corresponding scores. The paper shows that this pre-trained BERT model achieves state-of-the-art performance on short answer grading tasks.

In 2021, the authors in [20] proposed an interpretable deep learning system for automatically scoring requests for proposals (RFPs). The system is based on a deep neural network architecture that incorporates attention mechanisms, allowing for the identification of essential words and phrases in the RFPs. The system also employs a novel regularization technique to improve its interpretability. The authors show that the proposed system achieves high accuracy and provides insights into the key factors contributing to a successful RFP response.

Most recently, in 2022, Prabhu et al. [21] proposed a hybrid automated essay evaluation approach combining the BERT model with feature engineering techniques. BERT is a powerful language model pre-trained on large corpora and can be fine-tuned for specific tasks. The authors use BERT to extract meaningful features from essays, such as their semantic and syntactic features, while feature engineering techniques are employed to extract additional features, such as sentence length, vocabulary richness, and readability. The resulting essay representation is then used to predict the essay scores. The paper shows that this hybrid approach outperforms other automated essay evaluation systems regarding accuracy and consistency.

These papers demonstrate the potential of deep learning approaches, such as BERT and neural network architectures with attention mechanisms, in the context of automated grading. These approaches have shown to be effective in tasks such as essay evaluation, short answer grading, and RFP scoring.

TABLE I
SAMPLE DATA EXTRACTED FROM DATASET

Submitted Answer	Correct?	Remark	Grade

`SELECT DISTINCT s.snum`
`FROM PARKS P, SNACKS S, DELIVERS D, VENDORS V`
`WHERE V.VNum = D.VNum`
`AND P.PNum=D.PNum`
`AND S.SNum=D.SNum;`
	1	Correct	100

`SELECT V_TABLE.CITYCREATED`
`FROM VACCINES V_TABLE, INGREDIENTS I_TABLE`
`WHERE V_TABLE.CITYCREATED = I_TABLE.SOURCELOC`
`UNION`
`SELECT I_TABLE.SOURCELOC`
`FROM INGREDIENTS I_TABLE, DISEASES D_TABLE`
`WHERE I_TABLE.SOURCELOC = D_TABLE.ORIGIN`
`UNION`
`SELECT D_TABLE.ORIGIN`
`FROM DISEASES D_TABLE, VACCINES V_TABLE`
`WHERE D_TABLE.ORIGIN = V_TABLE.CITYCREATED;`
	0	Partially	20

`Select count(*)`
`From writer_award was, writer w, person p`
`Where wa.title=w.title`
`and w.id=p.id`
`and p.first_name='Woody'`
`and p.last_name='Allen';`
	1	Correct	100

⋮	⋮	⋮	⋮
Total sample count: 3361	Avg: 0.5956	Total Distinct: 4	Avg: 84.91

C. Dataset description and preprocessing

The dataset employed in our research pertains to SQL statements used for grading automatization, obtained from a publicly available database [22], and combined with internal data. The database schema of the public dataset contains tables with anonymous student submissions, feedback, grades, pass/fail flags, and other pertinent information. We have extracted valuable information from this dataset and combined it with our own data, with a few examples shown in Table I. The table illustrates that the dataset has 3361 distinct samples of student submissions with a class imbalance in which, on average, 59.6% of the submissions are graded as correct. Four different remarks are associated with the submissions, namely `Correct`, `Partially Correct`, `Non-Interpretable`, and `Cheating`, and the average grade of the assignments is 84.9. The BERT model will be fine-tuned on these SQL statements, which vary in length and are tokenized to a vocabulary of 30,522 tokens since this is the vocabulary size of BERT. The model is tokenized using the same tokenizer used to train BERT; the fine-tuning process is described in the next section.

III. METHODOLOGY

In this section, we describe our full methodology. We divide the discussion of our transformer-based approach depicted in Fig. 1, the explanation of the pre-training and fine-tuning process, and finally, a discussion about the model architecture and hyperparameters.
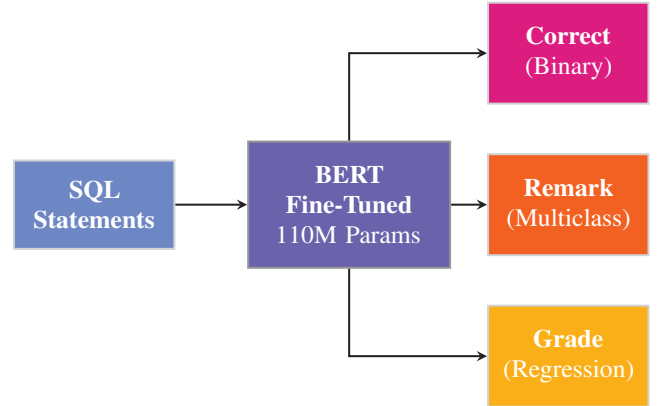

Fig. 1. Block Diagram for SQL Classification using BERT

A. Proposed transformer-based approach

As shown in the dataset description, we want to predict, given a SQL statement, if it is *Correct*, what would have been its *Remark* and a numeric value to its *Grade*. After trying several approaches and different SQL-based transformer-based encoders in Hugging Face, we decided to use the pretrained language model BERT. Specifically, we worked with the variant *bert-base-cased*,[1] which has twelve layers and 768 hidden units. The main reason for this decision was that the difference in performance with other models was low, and we could obtain an easier and more visible explainability from it using tools

[1]https://huggingface.co/bert-base-cased

like Captum.[2] We divide the problem into three separate tasks:

1) **Binary Classification**: Predict if the SQL statement is **Correct**.
2) **Multiclass Classification**: Predict the SQL statement **Remark**.
3) **Regression**: Predict the SQL statement **Grade**.

Our study employed the trainer pipeline from Hugging Face and utilized the BERT-cased model as the encoder of the SQL statement to address three distinct tasks. Our approach involved the classic BERT encoding of a text, followed by the use of the `CLS` token for classification and its encoding as the base for regression. To address the three tasks, we developed three separate BERT-cased finetuned models tailored to each task. The finetuning process was conducted over six epochs for each instance in its corresponding task.

B. Model architecture and hyperparameters

The BERT architecture has been successful because it can learn powerful contextual embeddings for words in a sentence [23]. BERT considers each word's context in the sentence and learns how words relate to each other using a multi-head self-attention mechanism [23]. BERT is pre-trained on a large corpus of text using masked language modeling and next-sentence prediction tasks, which allows the model to capture a wide range of semantic relationships between words [23]. Fine-tuning the pre-trained BERT model on a specific task further improves its performance [24].

Hyperparameter tuning is a crucial step in the fine-tuning of BERT models. BERT is a pre-trained language model that has achieved remarkable results in many natural language processing (NLP) tasks [25]. Fine-tuning BERT involves adapting the pre-trained model to a specific task by training it on a task-specific dataset. Hyperparameters are parameters set before the training process begins and can significantly impact the model's performance. Therefore, selecting optimal hyperparameters is essential for achieving the best possible performance [26].

Several studies have investigated the importance of hyperparameter tuning in the fine-tuning of BERT models. For instance, Mosbach [27] found that fine-tuning is an unstable process, and training the same model with multiple random seeds can result in a large variance in the task performance. Li [28] re-examined several common practices of setting hyperparameters for fine-tuning and found that some are sub-optimal. Xu [29] proposed two effective mechanisms, self-ensemble and self-distillation, to improve the fine-tuning of BERT. Therefore, we empirically determined that the hyperparameters in Table II are sufficient for successful training. The table shows the main hyperparameters for the three models during the training process. In our case, the three models were trained with the same hyperparameters in order to facilitate reproducibility.

IV. EXPERIMENTAL SETUP

The experimental methodology utilized in this study involved employing the trainer pipeline and application programming

[2]https://captum.ai/tutorials/Bert_SQUAD_Interpret

TABLE II
HYPERPARAMETERS USED FOR TRAINING THE THREE MODELS

Hyperparameter	Value
Training Epochs	6
Train batch size	8
Learning Rate	0.00002
Weight Decay	0.01
Optimizer	Adam

interface (API) provided by Hugging Face. The same pipeline was utilized for all three tasks. The sole variation was the number of labels considered by the BERT-cased encoder model and a distinct version of the dataset used for each specific task. To ensure reliable results, the dataset was randomly partitioned into training and testing subsets in a 70/30 ratio for each task. The performance of the model was assessed using various evaluation metrics for each task, including but not limited to the following:

1) **Binary Classification**: Micro F1 (Accuracy), Balanced Accuracy, F1, Precision, Recall.
2) **Multiclass Classification**: Micro F1 (Accuracy), Balanced Accuracy, Macro F1, Weighted F1, Precision, Recall.
3) **Regression**: Squared Mean Error, Mean Absolute Error, Coefficient of Determination and Explained Variance.

A. Baseline model for comparison

Our baseline model is the one introduced in [6]. The paper presents a lightweight neural architecture of a model for predicting the correctness, grade, and grader's remark of SQL statements. The architecture consists of an embedding layer, a self-attention CNN with a Luong-style layer and global average pooling, a dropout layer, a batch normalization layer, a bottleneck dense layer, and three different groups of outputs, each with a batch normalization layer and dense neural units. The model learns to optimize the learned embeddings through the self-attention component, and the bottleneck dense layer enables visualization of the learned representations in two dimensions. The outputs consist of a model for predicting correctness (with one neural unit), a model for predicting the grader's remark (with four neural units), and a model for predicting the grade (with one neural unit).

B. Training Methodology and Hardware Specifications

In order to evaluate the performance of our proposed model, we partitioned the dataset into two distinct sets: 70% for training and 30% for testing. We conducted the training process for a total of six epochs, utilizing the hyperparameters outlined in Table II.

To execute our experiment with optimal computational efficiency, we employed Google Colab, which provided us with access to a state-of-the-art NVIDIA GPU A100 that boasts a total memory capacity of 40 gigabytes. By leveraging this powerful hardware infrastructure, we were able to expedite the training process and obtain reliable performance metrics in an efficient and timely manner.

V. RESULTS AND ANALYSIS

We evaluated the proposed model on the testing set and report the results in Table III. To measure the performance of the classification tasks, we used standard evaluation metrics such as precision, recall, F1-score, and balanced accuracy. These metrics are defined as follows:

Precision is the proportion of true positives among the predicted positive samples, and is computed as:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{1}$$

where TP represents true positives and FP represents false positives.

Recall is the proportion of true positives among the actual positive samples, and is computed as:

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{2}$$

where FN represents false negatives.

F1-score is the harmonic mean of precision and recall, and is computed as:

$$F_1\text{-score} = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{3}$$

Balanced accuracy is the average of sensitivity and specificity, and is computed as:

$$\text{Balanced Accuracy} = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right), \tag{4}$$

where TN represents true negatives. We report the values of these metrics for each task in Table III. For measuring performance on the dataset in the case of regression we use the coefficient of determination (R^2), explained variance (EV), mean absolute error (MAE), and mean squared error (MSE). These metrics are defined as follows:

The coefficient of determination measures the proportion of variance in the dependent variable that is predictable from the independent variables. It is calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}i)^2}{\sum i = 1^n(y_i - \bar{y})^2}, \tag{5}$$

where y_i is the true value of the ith sample, $\hat{y}_i$ is the predicted value of the ith sample, $\bar{y}$ is the mean of the true values, and n is the number of samples.

Explained variance measures the proportion of variance in the dependent variable that is explained by the independent variables. It is calculated as:

$$\text{EV} = 1 - \frac{Var(y - \hat{y})}{Var(y)}, \tag{6}$$

where y is the true value of the dependent variable, $\hat{y}$ is the predicted value of the dependent variable, and Var is the variance.

Mean absolute error measures the average absolute difference between the predicted and true values. It is calculated as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \tag{7}$$

where y_i is the true value of the ith sample, $\hat{y}_i$ is the predicted value of the ith sample, and n is the number of samples.

Mean squared error measures the average squared difference between the predicted and true values. It is calculated as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{8}$$

where y_i is the true value of the ith sample, $\hat{y}_i$ is the predicted value of the ith sample, and n is the number of samples.

A. Comparison with baseline model

The present study compares the proposed model with a baseline model introduced in [5], [6]. The baseline model employs a convolutional neural network with self-attention, designed for inference efficiency and consisting of only 90,926 parameters. In contrast, BERT has a much larger parameter count of 110 million.

The baseline model adopts a multitask learning approach, where a single model shares the main parameters of the body and alternates between the classification and regression heads during training. In contrast, the proposed model involves a separate BERT fine-tuned for each classification and regression task. It is worth noting that the baseline model and its performance metrics, as shown in Table III, were reported using cross-validation. On the other hand, our proposed model uses a simple 70/30 data split for evaluation purposes.

Our results, as shown in Table III, demonstrate that the proposed model outperforms the baseline model on all classification and regression tasks, as we discuss next.

B. Analysis of the model's performance

The performance of the proposed model is analyzed in this section, and the results are presented in terms of accuracy, precision, recall, F_1-scores, balanced accuracy, and error metrics. In the binary classification task, the proposed model outperformed the baseline model by 11% in terms of accuracy. Moreover, precision, recall, and F_1-scores also showed superior performance compared to the baseline.

For the multi-class classification task, the proposed model showed a significant increase in terms of balanced accuracy, with predictions that were above and beyond random chance. Specifically, the model showed a +56% increase in balanced accuracy compared to the baseline. The precision, recall, and F_1-scores also showed superior performance.

In the case of regression, the proposed model outperformed the baseline by predicting with an average error of ± 0.015 of the target grade, while the baseline was off by ± 0.233 on average. The R^2 and EV scores also supported the superior performance of the proposed model. However, the baseline was better in terms of mean squared error (MSE), which may indicate that the proposed BERT-based model is susceptible to outliers in the data. Future studies may investigate methods to address this limitation and improve the performance of the proposed model in regression tasks.

TABLE III
CLASSIFICATION AND REGRESSION PERFORMANCE ANALYSIS ON THE TESTING SET

Class Being Evaluated	Evaluation Metrics							
	Precision		Recall		F_1-score		Support	
	CNN+Attn	BERT	CNN+Attn	BERT	CNN+Attn	BERT	CNN+Attn	BERT
Incorrect	0.76	**0.85**	0.80	**0.94**	0.78	**0.89**	282	360
Correct	0.85	**0.96**	0.82	**0.91**	0.84	**0.94**	393	649
Accuracy					0.81	**0.92**	675	1009
Balanced Accuracy					0.81	**0.92**	675	1009
Cheating	0.0	**1.0**	0.0	**1.0**	0.0	**1.0**	6	1
Correct	0.78	**0.95**	0.80	**0.91**	0.79	**0.93**	393	632
Non Interpretable	0.26	**0.74**	0.09	**0.85**	0.13	**0.79**	57	20
Partially Correct	0.51	**0.85**	0.58	**0.90**	0.54	**0.87**	219	356
Accuracy					0.66	**0.90**	675	1009
Balanced Accuracy					0.37	**0.93**	675	1009
Regression	R^2		EV		MAE		MSE	
$\hat{y}$ = Grade	0.148	**0.62**	0.421	**0.62**	0.233	**0.015**	**0.071**	0.122

Overall, the analysis of the model's performance indicates that the proposed BERT-based model is effective in classification and regression tasks and outperforms the baseline model. These results support the use of BERT-based models in educational data mining tasks, as they provide superior performance compared to traditional models.

C. Discussion of the model's strengths and limitations

Explainability is one of the main model's strengths beyond its superior performance compared to the baseline model. Using attribution score analysis, we can visualize the importance of certain elements in the SQL statements responsible for the model stimulation in favor of a specific outcome. To illustrate this capability, we used the Captum[3] tool. Captum supports explainability in transformer-based models employing gradient-based methods, which use the gradient of the model's output with respect to the input features to determine their importance. Particularly, in Captum the methods used are "integrated gradients" (IGs) and a variation called "layer integrated gradients" (LIGs). IGs are defined as the path integral of the gradients along the straight line path from the output y to the input x [30]. LIGs are a variant of IGs that assign an importance score to layer inputs or outputs, depending on whether we attribute to the former or the latter.

For illustration, Table IV shows the explainability obtained from our model using the Captum tool in the multiclass-classification task. We offer two examples per class. We believe the explainability in this class is the most interesting, although we can obtain the same type of explainability in binary classification and regression tasks.

All the examples included in Table IV are obtained based on the visualization provided by the Captum tool. The table includes the true "label" and the probability (a.k.a. "logit") that was assigned to the class during inference time. The "attribution score" is the numeric value obtained after computing the LIGs of the output class with respect to all the token inputs. The red color in a token shows a negative contribution, the green color in a token illustrates a positive contribution, and the white color is a neutral contribution. The darker the color, the stronger the negative or positive contribution. As shown, the attribution scores occur on the tokens obtained using the BERT-tokenizer, which can sometimes be word-piece tokens.

To discuss some interesting cases, the correct statements in the first and second row of Table IV show how the model emphasizes the SQL instructions and the usual type of structure that should come after each instruction. In the case of the partially correct statements, in the third row, we can appreciate how the model mainly focuses on the HAVING clause and strongly on the inequality signed used, which is the reason why this statement is not completely correct. In addition, the fourth row of the table shows a partially correct statement where the model identified as the main point for its decision on the ORDER BY clause, the value being used as the ordering field, and the use of ascending order instead of descending.

Our model is not limited to identifying correct or partially correct statements. Row five of Table IV shows a wrong statement that is not interpretable. The model emphasizes almost the entire statement due to the absence of SELECT and GROUP BY clauses. Row six illustrates a non-interpretable statement where the model focuses on every token since the WHERE clause has no details associated with it. Finally, the last two rows of Table IV show statements that turn out to be cheating from students. The model mainly focused on these cases because of the use of a specific numerical id within the query.

The limitations of BERT are known [31]; however, there are some that might be of particular interest to our study. Firstly, due to the lack of training data, the model's performance is limited even after fine-tuning. Secondly, BERT, the underlying language model, was originally trained on unstructured text using the Mask Language Modeling and Next Sentence Predictions task; therefore, it may encounter difficulties in processing structured text and understanding concepts related to databases, such as tables and statement relationships. Thirdly, BERT has a token limit of 512, which may lead to incorrect predictions for longer SQL queries and a loss of explainability.

[3]https://captum.ai/tutorials/Bert_SQUAD_Interpret

TABLE IV
EXPLAINABILITY OBTAINED FROM THE MODEL USING THE CAPTUM TOOL IN THE MULTICLASS-CLASSIFICATION TASK. TWO EXAMPLES PER CLASS SHOWN

Label	Logit	Attribution Score	Word Importance	Legend: Negative Neutral Positive
Correct	0.99	2.48	SELECT p1 . pnum FROM parks p1 WHERE p1 . location = (SELECT MIN (p2 . location) FROM parks p2) ;	
Correct	0.99	5.36	SELECT DISTINCT T . VNUM FROM DISEASES D , TREATS T WHERE T . DNUM = D . DNUM AND D . ORIGIN = ' NYC ' ;	
Partially Correct	0.99	1.66	SELECT d . snum FROM delivers d GROUP BY d . snum HAVING AVG (d . amount) > 320 ;	
Partially Correct	0.95	1.29	SELECT p . pnum FROM parks p ORDER BY p . location ASC ;	
Non Interpretable	0.87	2.41	from person having min (year _ born) ;	
Non Interpretable	0.85	0.82	SELECT p . id FROM person WHERE ;	
Cheating	0.71	2.28	select title , production _ year from writer where id = ' 00000402 ' ;	
Cheating	0.59	2.27	SELECT id FROM person WHERE id ! = ' 00000903 ' ORDER BY year _ born DESC LIMIT 1 ;	

Lastly, the use of aliases and the fact that multiple words can refer to the same thing can pose challenges for BERT in terms of handling synonyms and homonyms, as well as detecting relationships between names and their corresponding objects within the SQL statement. These limitations should be taken into consideration when interpreting the results of this study.

Finally, while ChatGPT has demonstrated impressive natural language understanding and generation capabilities, it is unsuitable for our current research project. One of the primary reasons is the privacy concerns associated with student data. Given the sensitive nature of this data, using a model like ChatGPT, which is trained on a vast corpus of internet text, could potentially lead to privacy breaches [32]. Furthermore, the lack of explainability and transparency in ChatGPT's decision-making process is another significant concern [33]. As of today, the architecture of ChatGPT is not publicly accessible, which hinders our ability to understand and interpret the model's behavior. This is particularly crucial in our project, where we aim to provide an analysis of the model's explainability to understand its decision-making process. Therefore, for these reasons, ChatGPT falls outside the scope of our research.

VI. CONCLUSIONS AND FUTURE WORK

The present study proposes a novel methodology for determining the correctness of an SQL statement. The approach employs a fine-tuned version of BERT using a mixture of publicly available and private data. While this extends and improves our earlier work in creating a lightweight baseline model [5], [6], to the best of our knowledge, BERT has not been previously used for the automatic grading of SQL statements.

Our experiments demonstrate the efficacy of the proposed model, achieving a cross-validated balanced accuracy of 92% and 93% for the binary and multiclass classification tasks, respectively. This represents an improvement over the baseline

model by 11% and 56%. The results suggest that the proposed methodology has the potential to achieve good generalization beyond the dataset used. When it comes to predicting the grade using regression, our new model scored higher in the MAE metric but not so in the MSE metric, this suggests that the model may be producing grades with large outliers, which will require further investigation.

Furthermore, analyzing the attribution scores of the model reveals a new capability of explainability that can be extremely beneficial. Specifically, visualizing the fundamental reason why an SQL statement stimulated the multiple attention heads of BERT provides a means of understanding the decision-making process of the model.

Based on our prior work and the results presented in this paper, we can establish that this type of system can effectively model the problem of automatic grading of SQL statements, provided sufficient examples for various assignments. Future work will focus on obtaining more labeled data encompassing a wider range of submissions and assignments, with the goal of further improving the model's performance and generalization capabilities.

REFERENCES

[1] J. Wang, Y. Zhao, Z. Tang, and Z. Xing, "Combining dynamic and static analysis for automated grading sql statements," *J Netw Intell*, vol. 5, no. 4, pp. 179–190, 2020.
[2] J. D. M.-W. C. Kenton and L. K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.
[3] A. Subakti, H. Murfi, and N. Hariadi, "The performance of bert as data representation of text clustering," *Journal of big Data*, vol. 9, no. 1, pp. 1–21, 2022.

[4] Z. Dai, Z. Li, and L. Han, "Bonebert: A bert-based automated information extraction system of radiology reports for bone fracture detection and diagnosis," in *Advances in Intelligent Data Analysis XIX: 19th International Symposium on Intelligent Data Analysis, IDA 2021, Porto, Portugal, April 26–28, 2021, Proceedings 19*. Springer, 2021, pp. 263–274.

[5] D. Schwartz and P. Rivas, "An automated sql query grading system using an attention-based convolutional neural network," in *The 18th International Conference on Frontiers in Education: Computer Science and Computer Engineering*, 2022, pp. 1–12.

[6] P. Rivas and D. R. Schwartz, "Modeling sql statement correctness with attention-based convolutional neural networks," in *The 19th International Conference on Scientific Computing (CSC 2021)*, 2021.

[7] A. V. Aho, Y. Sagiv, and J. D. Ullman, "Equivalences among relational expressions," *SIAM Journal on Computing*, vol. 8, no. 2, pp. 218–246, 1979.

[8] I. Štajduhar and G. Mauša, "Using string similarity metrics for automated grading of sql statements," in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2015, pp. 1250–1255.

[9] S. Chu, B. Murphy, J. Roesch, A. Cheung, and D. Suciu, "Axiomatic foundations and algorithms for deciding semantic equivalences of sql queries," *arXiv preprint arXiv:1802.02229*, 2018.

[10] S. Sadiq, M. Orlowska, W. Sadiq, and J. Lin, "Sqlator: an online sql learning workbench," in *Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education*, 2004, pp. 223–227.

[11] S. Dekeyser, M. de Raadt, and T. Y. Lee, "Computer assisted assessment of sql query skills," in *Proceedings of the 18th Australasian Database Conference (ADC 2007)*, vol. 63. Australian Computer Society Inc., 2007, pp. 53–62.

[12] J. C. Prior and R. Lister, "The backwash effect on sql skills grading," *ACM SIGCSE Bulletin*, vol. 36, no. 3, pp. 32–36, 2004.

[13] A. Kleerekoper and A. Schofield, "Sql tester: an online sql assessment tool and its impact," in *Proceedings of the 23rd annual ACM conference on innovation and technology in computer science education*, 2018, pp. 87–92.

[14] B. Chandra, A. Banerjee, U. Hazra, M. Joseph, and S. Sudarshan, "Automated grading of sql queries," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 1630–1633.

[15] S. Hassan, A. A. Fahmy, and M. El-Ramly, "Automatic short answer scoring based on paragraph embeddings," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 10, pp. 397–402, 2018.

[16] M. Abdul Salam, M. A. El-Fatah, and N. F. Hassan, "Automatic grading for arabic short answer questions using optimized deep learning model," *Plos one*, vol. 17, no. 8, p. e0272269, 2022.

[17] M. A. Rahaman and H. Mahmud, "Automated evaluation of handwritten answer script using deep learning approach," *Transactions on Machine Learning and Artificial Intelligence*, vol. 10, no. 4, 2022.

[18] Y. Wang, M. Gales, K. M. Knill, K. Kyriakopoulos, A. Malinin, R. C. van Dalen, and M. Rashid, "Towards automatic assessment of spontaneous spoken english," *Speech Communication*, vol. 104, pp. 47–56, 2018.

[19] 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.

[20] S. Maji, A. Appe, R. Bali, A. G. Chowdhury, V. C. Raghavendra, and V. M. Bhandaru, "An interpretable deep learning system for automatically scoring request for proposals," in *2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2021, pp. 851–855.

[21] S. Prabhu, K. Akhila, and S. Sanriya, "A hybrid approach towards automated essay evaluation based on bert and feature engineering," in *2022 IEEE 7th International conference for Convergence in Technology (I2CT)*. IEEE, 2022, pp. 1–4.

[22] W. Jinshui, "Combining dynamic and static analysis for automated grading sql statements," *Zenodo*, Jun. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.6526769

[23] L. Rasmy, Y. Xiang, Z. Xie, C. Tao, and D. Zhi, "Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction," *NPJ digital medicine*, vol. 4, no. 1, p. 86, 2021.

[24] J. Phang, H. Liu, and S. R. Bowman, "Fine-tuned transformers show clusters of similar representations across layers," *arXiv preprint arXiv:2109.08406*, 2021.

[25] Y. Hao, L. Dong, F. Wei, and K. Xu, "Visualizing and understanding the effectiveness of bert," *arXiv preprint arXiv:1908.05620*, 2019.

[26] R. Ghawi and J. Pfeffer, "Efficient hyperparameter tuning with grid search for text categorization using knn approach with bm25 similarity," *Open Computer Science*, vol. 9, no. 1, pp. 160–180, 2019.

[27] M. Mosbach, M. Andriushchenko, and D. Klakow, "On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines," *arXiv preprint arXiv:2006.04884*, 2020.

[28] H. Li, P. Chaudhari, H. Yang, M. Lam, A. Ravichandran, R. Bhotika, and S. Soatto, "Rethinking the hyperparameters for fine-tuning," *arXiv preprint arXiv:2002.11770*, 2020.

[29] Y. Xu, X. Qiu, L. Zhou, and X. Huang, "Improving bert fine-tuning via self-ensemble and self-distillation," *arXiv preprint arXiv:2002.10345*, 2020.

[30] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 3319–3328.

[31] M. Podkorytov, D. Biś, and X. Liu, "How can the [mask] know? the sources and limitations of knowledge in bert," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[32] P. Rivas and L. Zhao, "Marketing with chatgpt: Navigating the ethical terrain of gpt-based chatbot technology," *AI*, vol. 4, no. 2, pp. 375–384, 2023.

[33] P. Rivas, K. Holzmayer, C. Hernandez, and C. Grippaldi, "Excitement and concerns about machine learning-based chatbots and talkbots: A survey," in *2018 IEEE International Symposium on Technology and Society (ISTAS)*. IEEE, 2018, pp. 156–162.