IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 September 2024, accepted 11 November 2024, date of publication 15 November 2024,
date of current version 25 November 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3498841

RESEARCH ARTICLE

TURSpider: A Turkish Text-to-SQL Dataset and LLM-Based Study

ALI BUGRA KANBUROGLU[1] **AND FAIK BORAY TEK**[2]
[1]Department of Computer Engineering, Işık University, 34980 Istanbul, Türkiye
[2]Department of Artificial Intelligence and Data Engineering, Istanbul Technical University, 34467 Istanbul, Türkiye

Corresponding author: Ali Bugra Kanburoglu (bugra.kanburoglu@isik.edu.tr)

ABSTRACT This paper introduces TURSpider, a novel Turkish Text-to-SQL dataset developed through human translation of the widely used Spider dataset, aimed at addressing the current lack of complex, cross-domain SQL datasets for the Turkish language. TURSpider incorporates a wide range of query difficulties, including nested queries, to create a comprehensive benchmark for Turkish Text-to-SQL tasks. The dataset enables cross-language comparison and significantly enhances the training and evaluation of large language models (LLMs) in generating SQL queries from Turkish natural language inputs. We fine-tuned several Turkish-supported LLMs on TURSpider and evaluated their performance in comparison to state-of-the-art models like GPT-3.5 Turbo and GPT-4. Our results show that fine-tuned Turkish LLMs demonstrate competitive performance, with one model even surpassing GPT-based models on execution accuracy. We also apply the Chain-of-Feedback (CoF) methodology to further improve model performance, demonstrating its effectiveness across multiple LLMs. This work provides a valuable resource for Turkish NLP and addresses specific challenges in developing accurate Text-to-SQL models for low-resource languages.

INDEX TERMS Text-to-SQL, LLM, large language models, Turkish, dataset, TURSpider.

I. INTRODUCTION

The Text-to-SQL task aims to translate natural language statements into SQL queries based on given databases. This task has attracted significant interest from both academia and industry due to its potential to enable non-expert users to automatically extract information from databases using natural language. Text-to-SQL facilitates the development of question-answering systems [1] and chatbots capable of retrieving and presenting information stored within databases.

Recently, the development of large language models (LLMs) such as PaLM [2], GPT-4 [3] has driven substantial progress in the Text-to-SQL domain. These models use advanced techniques to manage the challenges [4] of converting natural language into SQL queries. The Spider [5] benchmark is frequently employed to evaluate these models due to its complexity, which includes many nested queries and advanced SQL clauses such as GROUP BY and ORDER BY.

Most research in this field has focused on the English language, with limited attention given to non-English languages. There are several datasets available in languages such as Chinese [6], Russian [7], Arabic [8], and Portuguese [9], including counterparts to the popular Spider dataset; however, the resources for Turkish remain notably limited. Although a Turkish Text-to-SQL dataset [10] exists, it only includes basic SQL queries, typically with a SELECT statement and a single WHERE clause, and does not have complex nested queries. This highlights the need for a more comprehensive benchmark for the Turkish Text-to-SQL task. The primary purpose of this study is to develop this benchmark dataset to allow evaluation of models and cross-language comparisons. With the development of the dataset, we were able to seek answers to the following research questions:

1) Can inference-based state-of-the-art LLMs already solve the Turkish Text-to-SQL problem? We address this by evaluating state-of-the-art LLMs on the Turkish Text-to-SQL task.

The associate editor coordinating the review of this manuscript and approving it for publication was Qichun Zhang.

2) Can Turkish LLMs fine-tuned on the Turkish Spider (TURSpider) dataset solve the Turkish Text-to-SQL problem? We explore this by fine-tuning Turkish LLMs on the TURSpider dataset and assessing their performance.
3) How do fine-tuned Turkish LLMs perform compared to the GPT-3.5 Turbo and GPT-4 on the TURSpider dataset? We investigate this by comparing the performance of fine-tuned Turkish LLMs with GPT-3.5 Turbo and GPT-4 on TURSpider.
4) How can we improve LLM performances for the Turkish Text-to-SQL problem? We approach this by applying the CoF methodology to enhance LLM performance on Turkish Text-to-SQL.

To address these research questions, the TURSpider dataset, with complex domain queries and a substantial number of examples similar to the Spider dataset, was essential. Instead of creating an entirely new dataset from scratch, we chose to translate the original Spider dataset to facilitate cross-language comparisons. Moreover, rather than relying on machine translation, we employed a rigorous human translation approach to ensure a high-quality, authentic Turkish dataset.

Open-source Turkish-supported large language models do not perform well for the Turkish Text-to-SQL problem. Therefore, we fine-tuned these models using our proposed TURSpider dataset to enhance their capabilities. We observed that this fine-tuning significantly improves the models' performance in handling Turkish Text-to-SQL tasks.

This paper presents the development and release of the TURSpider dataset, along with an exploration of fine-tuning LLMs to enhance their capabilities in Turkish Text-to-SQL tasks. Our contributions can be summarized as follows:

- Construction of the TURSpider Dataset: We created and released the TURSpider dataset, which is specifically designed for Turkish. This provides a benchmark for evaluating Text-to-SQL systems in Turkish, improving the availability of large-scale Turkish Text-to-SQL datasets.
- Fine-tuning Turkish Language Models: Using the TURSpider dataset, we fine-tuned Turkish large language models. By refining these models with the new dataset, we show that one can improve their ability to handle Text-to-SQL tasks, making them more useful in real-world applications.
- Comparative Analysis with Advanced Language Models: Our study compares the fine-tuned Turkish large language models with advanced models like OpenAI GPT-3.5 Turbo and GPT-4. This comparison helps us understand the performance and abilities of Turkish-specific models compared to globally trained ones, providing insights into language modeling across different languages.

The organization of the paper is as follows: In Section II, we describe the task formulation of the Text-to-SQL task in terms of large language models. In Section III, we explain related works. Section IV describes how we constructed the TURSpider dataset with detailed steps. Section V shows the experimental setup for our comparative experiments and fine-tuning of large language models, along with results and error analysis. Finally, Section VI presents the conclusion of the study.

II. TASK FORMULATION

In the context of Language Model-based Text-to-SQL (LLM-based Text-to-SQL), the task involves processing an input comprising a natural language statement (N) and corresponding database schema information (S). The database schema (S) is represented as $S = (T, C)$, where $T = \{t_1, \ldots, t_m\}$ denotes multiple tables and $C = \{c_1, \ldots, c_n\}$ represents columns within those tables.

The main goal of the Text-to-SQL task is to automatically generate a target SQL query (Q) based on the provided natural language statement (N) and the database schema information (S). This involves using an LLM to understand the natural language statement (N) within the context of the given database schema (S) and subsequently generate the corresponding SQL query (Q). The task can be formulated as:

$$Q = \text{LLM}(N, S)$$

The database schema (S) serves as essential contextual information for understanding the semantics of the natural language statement (N) and creating an accurate SQL query (Q) that properly interacts with the database tables and columns defined within S.

Several challenges in the Text-to-SQL domain were discussed in a recent study [11]. The *ordering issue* arises when SQL queries with identical outcomes have predicates listed in different orders, leading to multiple valid query representations. The complex and cross-domain task requires models to handle diverse and complicated SQL queries across various domains, needing a high level of generalization. The *lack of information* issue is caused by insufficient domain-specific knowledge and rare entities in natural language queries, resulting in inaccurate translations. The *mismatch problem* occurs when there is a discrepancy between SQL column names and their natural language descriptions, causing confusion. The *lexical problem* comes from the absence of key words in the training set, which makes it difficult for the model to handle out-of-domain terms. Lastly, the *schema representation problem* involves the difficulty of generalizing models to unseen database schemas, which requires accurately encoding and linking natural language with schema information.

III. RELATED WORK

Text-to-SQL research has been an active area of study in recent years, supported by the availability of diverse datasets and benchmarks, particularly designed for English [12], [13], [14], [15] and Chinese [6], [16], [17], [18], [19] languages. The Spider dataset, introduced by Yu et al. [5], features

complex and cross-domain SQL queries, distinguishing it from other datasets like WikiSQL [14]. Spider consists of 10,181 questions and 5,693 unique SQL queries sourced from 200 databases covering 138 different domains. The dataset was carefully curated and reviewed by 11 computer science students, spending a total of 1,000 man-hours.

Efforts have been made to adapt the Spider dataset into other languages in chronological order. Min et al. [6] developed CSpider, a Chinese adaptation, by translating the English questions while maintaining the original database schemas in English. They translated all English questions into Chinese for training and development sets but kept the database schema in English. This translation involved two NLP researchers and one computer science student, ensuring accuracy through multiple checks.

Nguyen et al. [20] created ViText2SQL, a large Text-to-SQL dataset for Vietnamese, by translating questions and database schemas into Vietnamese for the training and development sets of Spider. This was done by one NLP researcher and two proficient computer science students, with careful validation to maintain accuracy. Their human-translated dataset is significantly more reliable than those consisting of machine-translated questions. They found that the overall results for Vietnamese were slightly lower but comparable to the results in English.

Jose and Cozman [9] created a Portuguese version of the Spider dataset by translating the questions while keeping the original English database schema. They extracted and translated the questions using the Google Cloud Translation API.[1] They concluded that a multilingual approach was necessary, as conducting all experiments solely in Portuguese was insufficient. Their experiments demonstrated that training with both the original and translated datasets yields better results, even if the target language is Portuguese.

Bakshandaeva et al. [7] contributed PAUQ, the first publicly available Russian Text-to-SQL dataset, by localizing all components of Spider into Russian, including questions, SQL queries, and database content. They also added new samples for improvement. The translation process was similar to CSpider, utilizing human and machine translation datasets. They created a machine-translated (MT) dataset using the Yandex Translate API.[2] Their experiments compared the performance of models trained on manually translated (MT) PAUQ data with those trained on machine-translated (MT) or combined (MT + HT) datasets. Results indicated that MT + HT models performed comparably to those trained on PAUQ.

Almohaimeed et al. [8] introduced Ar-Spider, the first Arabic cross-domain Text-to-SQL dataset, by translating Spider into Arabic using GPT-3 [21] and professional translators post-edited for training and development sets. The translation was performed by two professional translators and verified by three computer science graduate students. Their experiments showed that combining English and Arabic datasets during training did not improve performance. Models trained on the Arabic Ar-Spider dataset performed worse than those trained on the English dataset.

Additionally, Dou et al. [22] presented MultiSpider, the largest multilingual Text-to-SQL dataset, which covers seven languages (English, German, French, Spanish, Japanese, Chinese, and Vietnamese).

IV. TURSPIDER DATASET CONSTRUCTION

We created the TURSpider dataset by manually translating the original Spider dataset from English to Turkish. We chose to translate instead of developing a new dataset to facilitate cross-language studies, given that Spider includes translations in various languages, which simplifies comparative analysis. Initially, we conducted a comprehensive analysis of the Spider dataset, consisting of 166 databases, each containing 684 unique table names. These tables contain various columns with varying amounts of data, all data is stored in SQLite [23] files.

For the translation process, we employed a human translation approach as illustrated in Figure 1. We collaborated with three third-year Computer Engineering students from Istanbul Technical University, who are proficient in English and have achieved high scores in database courses. The data to be translated was divided into three parts, with each student responsible for translating their assigned portion. The process involved a multi-layered quality check system: the second student cross-checked, corrected, and validated the initial translation done by the first student, and the third student then cross-checked, corrected, and verified the second student's work. This approach ensured that each student had the opportunity to contribute to every round of translation, enabling efficient cross-checking, correction, and verification.

The human translation approach was applied to both the database schema, including database names, table names, and column names, as well as the dataset translation, which included questions and queries from the training and development sets. We did not include the Spider test set because it was not publicly available.

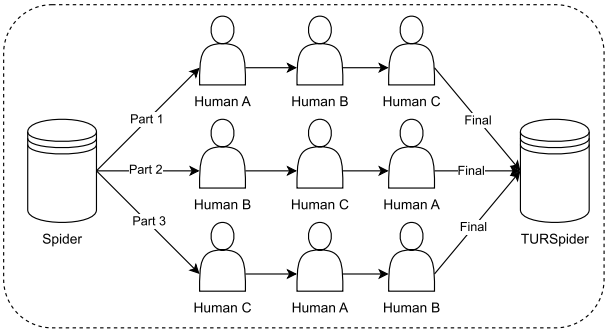

FIGURE 1. TURSpider human translation approach.

We followed several steps for the translation process: Firstly, we translated the English database names into

[1] https://googleapis.dev/python/translation/latest/index.html
[2] https://yandex.cloud/ru/docs/translate/

Turkish. Next, we translated the table names within these databases into Turkish as well. Moreover, we translated both the table names and column names of the tables. Despite some table names being located under different databases, we translated them differently based on their domains. For example, the word ''department'' has been translated as both ''departman'' and ''bakanlık'' (ministry in English) in some databases. Finally, the questions and queries within the training and development datasets were translated based on the translated database, table, and column names. Figure 2 illustrates the database schema structure along with examples from the English Spider and TURSpider datasets for both training and development data.

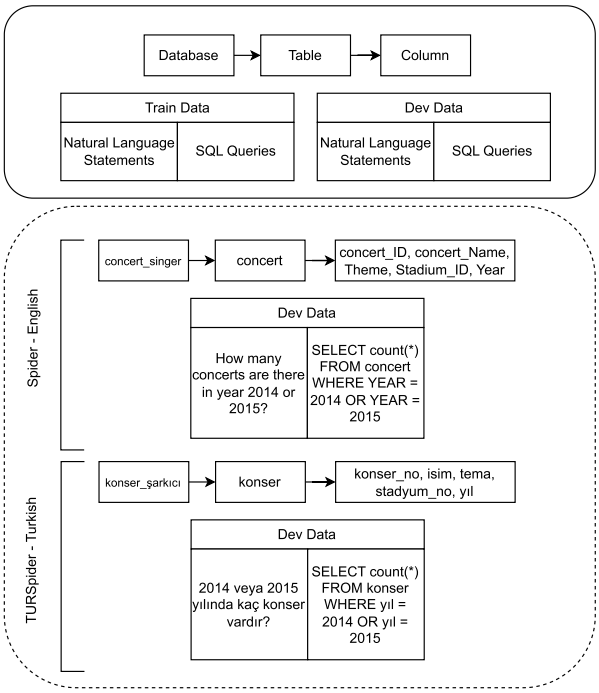

FIGURE 2. TURSpider database schema info.

A. MAN-HOURS SPENT

In the process of translating the dataset, we also analyzed the total man-hours spent. In the first round of database translation, 52 man-hours were utilized. For the development set's first round, 28 man-hours were consumed, while the first round of the training set's translation required 170 man-hours. Moving on to the second round of translations, a total of 72 man-hours were invested, followed by 42 man-hours in the third and final round. Thus, a total of 364 man-hours was dedicated to translating the Spider dataset into the TURSpider dataset, as shown in Figure 3.

B. DATA STATISTICS

The TURSpider dataset comprises two main subsets: a development set and a training set, aligned with the structure and scale of the popular Spider dataset. The development

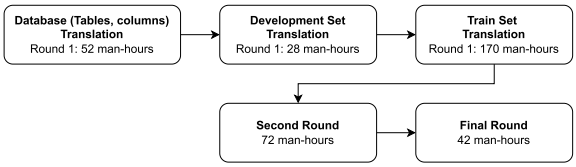

FIGURE 3. Man-hours spent across translation phases.

set contains 1034 data rows with 1023 unique questions and 584 distinct SQL queries. In the training set, there are 8659 data rows, 8506 unique questions, and corresponding SQL queries. Additionally, SQL query difficulty levels were considered. Table 2 presents both simple and challenging NL statements and their corresponding SQL queries in both English and Turkish. For detailed statistics, including question and SQL query uniqueness, and difficulty levels, please refer to Table 1. The size and variety of TURSpider, with its unique questions and diverse SQL queries, suggest that the findings could be useful for real-world applications.

TABLE 1. TURSpider dataset distribution. # Q, # SQL, # E, # M, # H, and # EH denote the numbers of unique questions, unique SQL queries, easy, medium, hard, and extra hard level SQL queries, respectively.

Dataset	Total	# Q	# SQL	# E	# M	# H	# EH
Dev	1034	1023	584	248	446	174	166
Train	8659	8506	4736	1983	2999	1922	1755

TABLE 2. Example natural language statements with corresponding SQL queries in TURSpider.

Sample 1: Easy NL statement with a basic SQL query.
Original SQL Query: SELECT count(*) FROM farm **Original English NL Statement:** How many farms are there? **Translated Turkish SQL Query:** SELECT count(*) FROM çiftlik **Translated Turkish NL Statement:** Burada kaç tane çiftlik var?
Sample 2: Hard NL statement with a complex SQL query.
Original SQL Query: SELECT T1.city FROM city AS T1 JOIN hosting_city AS T2 ON T1.city_id = T2.host_city GROUP BY T2.host_city ORDER BY count(*) DESC LIMIT 1 **Original English NL Statement:** Find the city that hosted the most events. **Translated Turkish SQL Query:** SELECT T1.Şehir FROM şehir AS T1 JOIN evsahibi_şehir AS T2 ON T1.Şehir_No = T2.Evsahibi_No GROUP BY T2.Evsahibi_No ORDER BY count(*) DESC LIMIT 1 **Translated Turkish NL Statement:** En çok etkinliğe ev sahipliği yapan şehri bul.

C. EVALUATION

We evaluated the translated dataset by applying an inter-annotator agreement measure. Three different students translated the same set of sentences, and the agreement among their translations was calculated. The inter-annotator

agreement measured 86.36% for the training set and 81.04% for the development set, indicating substantial agreement in the translations.

V. EXPERIMENTS

Two types of experiments were conducted to evaluate the TURSpider dataset for the Text-to-SQL task. Firstly, we employed an inference-only approach, utilizing the pre-trained LLM for predictions without any training. Secondly, we implemented a fine-tuning approach, where the LLM was further trained on our training set of TURSpider.

A. EXPERIMENTAL SETUP

In the literature, there is ongoing effort concerning Turkish large language models (LLMs), with new models continuously emerging. In April 2024, we conducted our experiments on selected models from Hugging Face [24]. In selecting Turkish language models for fine-tuning, we prioritized popular options such as Llama-2 [25] and Mistral [26] based models, opting for models with similar sizes, namely 7B. Thus, our choices included Trendyol LLM, Turkcell LLM, and Sambalingo LLM, all of which are industry-specific models from the Hugging Face repository. After the fine-tuning process, we call them TrendyolSQL, TurkcellSQL, and SambaLingoSQL, respectively.

1) DATASET

We conducted experiments on our proposed TURSpider dataset.

2) EVALUATION

We use execution accuracy (EX) [14] as the evaluation metric for all experiments. Execution accuracy measures the correctness by executing the ground truth (G) and the predicted (P) SQL queries over the underlying database and comparing the execution results. If the results are the same, then the prediction is considered as correct.

We first calculated the match score between ground truth SQL and prediction SQL:

$$\text{match}(G, P) = \begin{cases} 1 & : G = P \\ 0 & : G \neq P \end{cases} \quad (1)$$

Then, the execution accuracy (EX) is calculated by:

$$EX = \frac{\sum_{n=1}^{N} match(G, P)}{N} \quad (2)$$

3) IMPLEMENTATION

In our finetuning methodology, we utilized models configured with 4-bit quantization and 16-bit floating-point precision for computations. Additionally, the tokenizer was adjusted for right-side padding. During training, we employed SFTTrainer from Hugging Face with a batch size of 4 on a single A100 GPU, implementing a cosine learning rate schedule.

For GPT-3.5 Turbo [27] and GPT-4 [3] inferences, default values, such as temperature settings, were employed within the OpenAI API.

An example of an LLM prompt (in Turkish) used as input for Text-to-SQL translation is shown in Figure 4. The prompt translation in English is: "Write SQL query for given natural language statement and table information. Natural language statement: Where is the youngest teacher from? Table Information: teacher (Teacher_ID, Name, Age, Hometown)"

We have shared the code used in our experiments via a GitHub repository, which can be accessed at.[3]

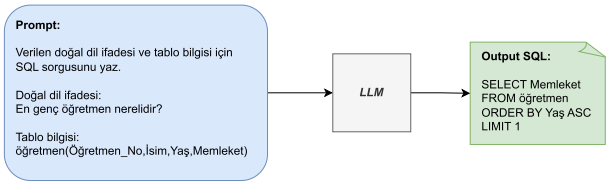

FIGURE 4. LLM prompt for Text-to-SQL translation.

4) LARGE LANGUAGE MODELS

We conducted experiments on the TURSpider dataset by using inference-only and fine-tuning approaches with the following models:

- **GPT-3.5 Turbo** is an OpenAI model available through the OpenAI API, which features a context window of 16,385 tokens, allowing it to process and understand more extensive text inputs.
- **GPT-4** is an OpenAI model that represents a remarkable advancement in artificial intelligence technology. With a context window of 128,000 tokens and approximately 1 trillion parameters, this model is designed to tackle complex tasks with exceptional accuracy and efficiency.
- **Trendyol-LLM-7b-base-v0.1** is a generative model based on the Mistral 7B model, which is trained in both English and Turkish.
- **SambaLingo-Turkish-Base** is a pretrained bilingual Turkish and English model that adapts Llama-2-7b to Turkish by training on 42 billion tokens.
- **Turkcell-LLM-7b-v1** is an extended version of a Mistral-based Large Language Model (LLM) for Turkish. It was trained on a cleaned Turkish raw dataset containing 5 billion tokens. The training process involved using the DORA [28] and LORA [29] methods.

B. RESULTS

We provided 1034 natural language statements and database schemas from the TURSpider development set to GPT-3.5 Turbo and GPT-4. The execution accuracy values of SQL queries obtained through inference-based methods were measured to be 55.99 and 57.25, respectively. However, fine-tuning Turkish-supported large language models (LLMs) with 8659 natural language statements and SQL query

[3]https://github.com/alibugra/TURSpider

pairs from the TURSpider training set, along with database schemas, resulted in different accuracies for different LLMs. For TrendyolSQL, an accuracy of 25.04 was achieved, for SambaLingoSQL it was 30.65, and for TurkcellSQL it was 58.22. These results are presented in Table 3. The TurkcellSQL model demonstrated higher accuracy compared to inference-based models and other fine-tuned models.

TABLE 3. Execution accuracy (EX) of LLMs on the TURSpider development set.

Approach	Model	Execution Accuracy (EX)
Inference-only	GPT-3.5 Turbo	55.99
Inference-only	GPT-4	**57.25**
Fine-tuning	TrendyolSQL	25.04
Fine-tuning	SambaLingoSQL	30.65
Fine-tuning	TurkcellSQL	**58.22**

Table 4 presents the execution accuracy results obtained from various LLMs when evaluated against different difficulty levels of SQL queries using the TURSpider development dataset. The analysis reveals distinct performance variations among the LLMs across varying query complexities. For SQL queries categorized under medium difficulty, GPT-4 demonstrates superior execution accuracy compared to other LLMs included in the study. This finding underscores the enhanced capabilities of GPT-4 in accurately interpreting and executing moderately challenging SQL queries. TurkcellSQL emerges as the standout performer across all levels of query difficulty, including easy, hard, and extra hard categories, as well as across the entire spectrum of query complexities evaluated. TurkcellSQL consistently outperforms its counterparts, highlighting its robustness and effectiveness in handling diverse SQL query challenges.

Recently, several advanced methods [11] have demonstrated superior performance on the original Spider dataset compared to the results we obtained on our dataset. For instance, custom methods have been tested on the Spider dataset, with remarkable results. The DIN-SQL model, for example, achieved an execution accuracy of 82.8% with GPT-4. Additionally, SQL-PaLM reached an execution accuracy of 82.7% on the Spider dataset.

TABLE 4. Execution accuracy (EX) of LLMs with different hardness levels on the TURSpider development set.

Model	Easy	Medium	Hard	Ex. Hard	All
GPT-3.5 Turbo	80.64	58.29	39.65	30.12	55.99
GPT-4	76.20	**62.10**	41.95	31.92	57.25
TrendyolSQL	40.72	21.97	20.68	14.45	25.04
SambaLingoSQL	50.80	24.43	34.48	13.25	30.65
TurkcellSQL	**82.25**	54.93	**55.74**	**33.73**	**58.22**

C. CHAIN-OF-FEEDBACK

Chain-of-Thought (CoT) [30] has been a popular prompting technique that improves the reasoning capabilities of large language models. CoT works by breaking down a complex problem into smaller and sequential steps. Inspired by CoT prompting, a number of Chain-of-X (CoX) methods [31] have been developed.

One significant variant of CoX is the Chain-of-Feedback (CoF), which uses the output of LLMs as feedback throughout the generation process to improve responses. In our study, we employ the CoF method to improve the Text-to-SQL capabilities of LLMs. Our approach involves an iterative feedback loop where the generated SQL queries are executed, and errors from this execution are fed back into the system. The initial prompt and the error feedback are then used to regenerate the SQL query.

Table 5 provides a comparative overview of model execution accuracy (EX) and the number of execution errors for different iterations of language models, including GPT-3.5 Turbo, GPT-4, and TurkcellSQL, both individually and with the Chain-of-Feedback (CoF) method.

TurkcellSQL achieves the highest execution accuracy at 58.22, with 111 execution errors. However, upon integrating the CoF approach, there is a noticeable improvement across all models. GPT-4 + CoF reaches the highest execution accuracy at 59.57, with 59 execution errors, followed closely by TurkcellSQL + CoF at 60.05, with 70 execution errors. Interestingly, while GPT-3.5 Turbo + CoF and GPT-4 + CoF experience a reduction in execution accuracy compared to their unaugmented counterparts, they demonstrate significant reductions in the number of execution errors, indicating a more refined and precise execution process facilitated by the CoF methodology. Overall, the integration of Chain-of-Feedback proves to be a beneficial enhancement strategy, leading to improved execution accuracy and reduced errors across various language model architectures.

TABLE 5. Execution accuracy (EX) of LLMs with and without Chain-of-Feedback (CoF) on TURSpider development set.

Model	Execution Acc. (EX)	# of Execution Errors
GPT-3.5 Turbo	55.99	47
GPT-4	57.25	103
TurkcellSQL	**58.22**	111
GPT-3.5 Turbo + CoF	56.38	28
GPT-4 + CoF	59.57	59
TurkcellSQL + CoF	**60.05**	70

D. ERROR ANALYSIS

GPT-3.5 Turbo, GPT-4, and TurkcellSQL were evaluated based on their ability to generate SQL queries against a database schema. The analysis revealed discrepancies between the generated queries and their execution results.

For GPT-3.5 Turbo, out of 1034 generated queries, 579 (55.99%) matched the execution results, while the remaining 455 queries either failed to match or were inaccurately generated. A manual inspection of these 455 queries showed that 408 (89.67%) were syntactically correct SQL queries but did not match the execution results. Additionally, 47 (10.33%) queries were found to be inaccurately generated.

Among these inaccuracies, it was observed that 25 queries used non-existent column names, 14 used non-existent table names, 1 had an ambiguous column reference, 1 had the wrong number of arguments for a function, 2 used non-existent functions, and 4 had syntax errors.

Similarly, for GPT-4, out of 1034 generated queries, 592 (57.25%) matched the execution results, leaving 442 queries with discrepancies. Upon manual inspection, 339 (76.70%) of these queries were syntactically correct but did not match the execution results. Additionally, 103 (23.30%) queries were inaccurately generated. Among these inaccuracies, it was found that 26 queries used non-existent column names, 66 used non-existent table names, 2 had ambiguous column references, 1 had the wrong number of arguments for a function, 2 used non-existent functions, and 6 had syntax errors.

TABLE 6. Error distribution on generated SQL queries for different LLMs.

Error Type	Models		
	GPT-3.5 Turbo	GPT-4	TurkcellSQL
	Occurrences		
No such table	14	66	15
No such column	25	26	83
No such function	2	2	0
Ambiguous column	1	2	7
Syntax error	4	6	4
Misuse	0	0	2
Wrong # of arguments	1	1	0
TOTAL	47	103	111

Further analysis focused on the TurkcellSQL model. For this model, out of 1034 generated queries, 602 (58.22%) matched the execution results, while the remaining 432 queries had issues. Manual examination showed that 321 (74.31%) of these queries were syntactically correct but did not match the execution results. Additionally, 111 (25.69%) queries were inaccurately generated. Among these inaccuracies, it was observed that 83 queries used non-existent column names, 15 used non-existent table names, 7 had ambiguous column references, 2 had misuse errors, and 4 had syntax errors.

We also conducted a more detailed analysis of the error types shown in Table 6. Since most errors are caused by "no such table" and "no such column" errors, we focused on these two error types in more detail. We provide examples for "no such table" and "no such column" errors with gold and predicted SQL queries in Table 7. We observed that errors mostly arise from Turkish character errors, plural suffix errors, and hallucinations. For instance, in the case of a Turkish character error, if the table name is "öğrenci" (student in English), LLM predicted it as "ogrenci". Plural suffix errors were seen where "arabalar" (cars in English) was written as "araba" (car in English). For hallucinations, we noticed instances where English words were used instead of Turkish table or column names. There were also made-up words, and underscores were used to separate compound words. The distribution of these errors for GPT-3.5 Turbo, GPT-4, and TurkcellSQL is shown in Figure 5.

TABLE 7. Examples of errors in generated SQL queries.

Error Type	Example
No such table	Gold: SELECT isim , soyisim , eposta_adresi FROM Sahipler WHERE eyalet LIKE '%North%' Pred: SELECT isim , soyisim , eposta_adresi FROM sahip WHERE eyalet LIKE '%North%'
No such column	Gold: SELECT max(ağırlık), EvcilHayvanTürü FROM Evcil_hayvanlar GROUP BY EvcilHayvanTürü Pred: SELECT max(ağırlık) , evcil_hayvan_yaşı , evcil_hayvan_türü FROM Evcil_Hayvanlar GROUP BY evcil_hayvan_türü

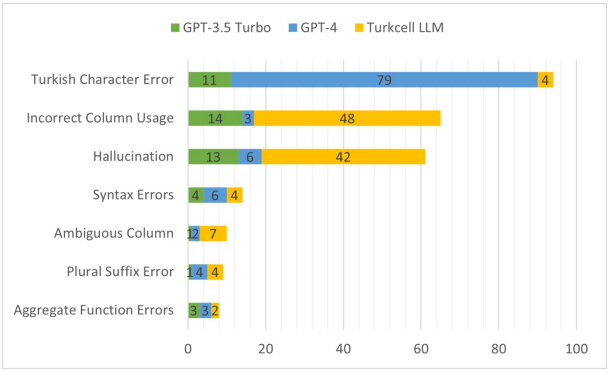

FIGURE 5. Error detail distribution.

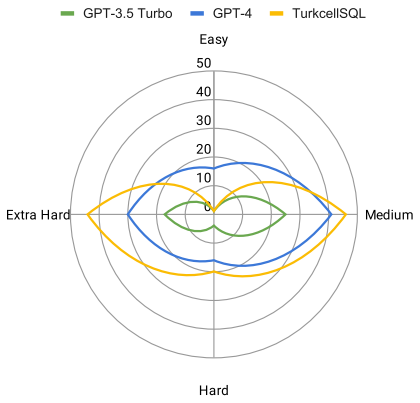

FIGURE 6. Number of errors made by LLMs across different difficulty levels.

Figure 6 compares the number of errors made by the three models, GPT-3.5-Turbo, GPT-4, and TurkcellSQL, across different difficulty levels. GPT-3.5-Turbo performs well on easy and hard tasks but struggles with medium and extra hard challenges. GPT-4 shows a more balanced distribution of errors, particularly with increases in medium and extra hard tasks. TurkcellSQL does well in easy tasks but has significant difficulties in medium and extra hard

levels. Overall, these findings indicate that while GPT-3.5-Turbo and TurkcellSQL are effective with simpler tasks, all models face challenges with more complex queries.

VI. CONCLUSION

In this paper, we introduced the TURSpider, a large-scale, complex, and cross-domain Text-to-SQL dataset that contains SQL queries in different hardness levels, translated from the original English Spider dataset into Turkish. TURSpider establishes a robust benchmark for Turkish Text-to-SQL tasks, addressing the current lack of advanced Turkish datasets. Additionally, we compared the performance of fine-tuned Turkish large language models (LLMs) with inference-based models such as GPT-3.5 Turbo and GPT-4, demonstrating that the Turkish LLMs perform competitively and often surpass the inference-based models. Furthermore, we conducted an error analysis based on generated SQL queries to provide valuable insights and directions for future research. We believe that the public release of our dataset[4] can serve as a useful research benchmark for further Turkish and cross-language Text-to-SQL research and applications.

For future work, to achieve more robust results in Turkish Text-to-SQL studies, existing Turkish-supported LLMs need to perform better in Turkish benchmarks. By fine-tuning models that are good at reasoning, we may achieve more successful results. Moreover, when we examine LLM studies conducted with the English Spider dataset, we observe that pre-trained models are less effective, whereas models fine-tuned with Spider perform better. Similarly, we observe that models that are good at coding and reasoning also perform better in the Text-to-SQL task. Additionally, LLM prompts can be rewritten to improve performance compared to existing prompts.

LIMITATIONS

The TURSpider dataset, while valuable for advancing Turkish Text-to-SQL research, has limitations related to dataset bias [32]. These include potential translation bias from the original Spider dataset, which could lead to inaccuracies due to linguistic differences between English and Turkish. Additionally, the dataset may not fully reflect Turkish cultural or contextual nuances, limiting its applicability in local contexts.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

ACKNOWLEDGMENT

The authors would like to sincerely thank Giray Yıldırım, Aydan Günaydın, and Metin Soyalp, students at the Department of Computer Engineering, İstanbul Technical University, for their outstanding efforts in translating the original Spider dataset into Turkish.

[4]https://github.com/alibugra/TURSpider

REFERENCES

[1] P. Wang, T. Shi, and C. K. Reddy, "Text-to-SQL generation for question answering on electronic medical records," in *Proc. Web Conf.*, Apr. 2020, pp. 350–361.
[2] R. Anil et al., "PaLM 2 technical report," 2023, *arXiv:2305.10403*.
[3] J. Achiam et al., "GPT-4 technical report," 2023, *arXiv:2303.08774*.
[4] B. Zhang, Y. Ye, G. Du, X. Hu, Z. Li, S. Yang, C. Harold Liu, R. Zhao, Z. Li, and H. Mao, "Benchmarking the text-to-SQL capability of large language models: A comprehensive evaluation," 2024, *arXiv:2403.02951*.
[5] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev, "Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3911–3921.
[6] Q. Min, Y. Shi, and Y. Zhang, "A pilot study for Chinese SQL semantic parsing," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, 2019, pp. 3650–3656.
[7] D. Bakshandaeva, O. Somov, E. Dmitrieva, V. Davydova, and E. Tutubalina, "Pauq: Text-to-SQL in Russian," in *Findings Assoc. for Comput. Linguistics: EMNLP*. Abu Dhabi, UAE: Association for Computational Linguistics, 2022, pp. 2355–2376.
[8] S. Almohaimeed, S. Almohaimeed, M. Al Ghanim, and L. Wang, "Ar-spider: Text-to-SQL in Arabic," 2024, *arXiv:2402.15012*.
[9] M. A. José and F. G. Cozman, "Mrat-SQL+ gap: A Portuguese text-to-SQL transformer," in *Proc. 10th Brazilian Conf.*, Apr. 2021, pp. 511–525.
[10] A. B. Kanburoglu and F. Boray Tek, "TUR2SQL: A cross-domain Turkish dataset for Text-to-SQL," in *Proc. 8th Int. Conf. Comput. Sci. Eng. (UBMK)*, Sep. 2023, pp. 206–211.
[11] A. B. Kanburoglu and F. B. Tek, "Text-to-SQL: A methodical review of challenges and models," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 32, no. 3, pp. 403–419, May 2024.
[12] P. J. Price, "Evaluation of spoken language systems: The ATIS domain," in *Proc. Workshop Speech Natural Lang.*, 1990, pp. 91–95.
[13] S. Iyer, I. Konstas, A. Cheung, J. Krishnamurthy, and L. Zettlemoyer, "Learning a neural semantic parser from user feedback," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 963–973.
[14] V. Zhong, C. Xiong, and R. Socher, "Seq2SQL: Generating structured queries from natural language using reinforcement learning," 2017, *arXiv:1709.00103*.
[15] J. Sen, C. Lei, A. Quamar, F. Özcan, V. Efthymiou, A. Dalmia, G. Stager, A. Mittal, D. Saha, and K. Sankaranarayanan, "ATHENA++: Natural language querying for complex nested SQL queries," *Proc. VLDB Endowment*, vol. 13, no. 12, pp. 2747–2759, Aug. 2020.
[16] N. Sun, X. Yang, and Y. Liu, "TableQA: A large-scale Chinese Text-to-SQL dataset for table-aware SQL generation," 2020, *arXiv:2006.06434*.
[17] L. Wang, A. Zhang, K. Wu, K. Sun, Z. Li, H. Wu, M. Zhang, and H. Wang, "DuSQL: A large-scale and pragmatic Chinese Text-to-SQL dataset," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2020, pp. 6923–6935.
[18] J. Guo, Z. Si, Y. Wang, Q. Liu, M. Fan, J.-G. Lou, Z. Yang, and T. Liu, "Chase: A large-scale and pragmatic Chinese dataset for cross-database context-dependent text-to-SQL," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, 2021, pp. 2316–2331.
[19] S. Huang, L. Wang, Z. Li, Z. Liu, C. Dou, F. Yan, X. Xiao, H. Wu, and M. Zhang, "Sesql: A high-quality large-scale session-level Chinese text-to-SQL dataset," in *Proc. CCF Int. Conf. Natural Lang. Process. Chin. Comput.*, 2023, pp. 537–550.
[20] A. T. Nguyen, M. H. Dao, and D. Q. Nguyen, "A pilot study of text-to-sql semantic parsing for Vietnamese," in *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, 2020, pp. 4079–4085.
[21] T. B. Brown et al., "Language models are few-shot learners," in *Proc. NIPS*, Apr. 2020, pp. 1877–1901.
[22] L. Dou, Y. Gao, M. Pan, D. Wang, W. Che, D. Zhan, and J.-G. Lou, "MultiSpider: Towards benchmarking multilingual text-to-SQL semantic parsing," *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 11, pp. 12745–12753, Jun. 2023.
[23] M. Owens, *The Definitive Guide to SQLite*. Cham, Switzerland: Springer, 2006.

[24] T. Wolf et al., "HuggingFace's transformers: State-of-the-art natural language processing," 2019, *arXiv:1910.03771*.
[25] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," 2023, *arXiv:2307.09288*.
[26] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. Le Scao, T. Lavril, T. Wang, T. Lacroix, and W. El Sayed, "Mistral 7B," 2023, *arXiv:2310.06825*.
[27] J. Ye, X. Chen, N. Xu, C. Zu, Z. Shao, S. Liu, Y. Cui, Z. Zhou, C. Gong, Y. Shen, J. Zhou, S. Chen, T. Gui, Q. Zhang, and X. Huang, "A comprehensive capability analysis of GPT-3 and GPT-3.5 series models," 2023, *arXiv:2303.10420*.
[28] S.-Y. Liu, C.-Y. Wang, H. Yin, P. Molchanov, Y.-C. F. Wang, K.-T. Cheng, and M.-H. Chen, "DoRA: Weight-decomposed low-rank adaptation," 2024, *arXiv:2402.09353*.
[29] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," 2021, *arXiv:2106.09685*.
[30] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 24824–24837.
[31] Y. Xia, R. Wang, X. Liu, M. Li, T. Yu, X. Chen, J. McAuley, and S. Li, "Beyond chain-of-thought: A survey of chain-of-X paradigms for LLMs," 2024, *arXiv:2404.15676*.
[32] V. Dogra, S. Verma, Kavita, M. Wouniak, J. Shafi, and M. F. Ijaz, "Shortcut learning explanations for deep natural language processing: A survey on dataset biases," *IEEE Access*, vol. 12, pp. 26183–26195, 2024.

ALI BUGRA KANBUROGLU received the B.Sc. degree in mathematics with a minor in computer science and engineering and the M.Sc. degree in computer science and engineering from FMV Işık University, in 2015 and 2018, respectively, where he is currently pursuing the Ph.D. degree in computer science and engineering. He is also the Master Expert AI Technical Product Manager at Turkcell. His research interests include large language models, recommendation systems, natural language processing, and deep learning.

FAIK BORAY TEK received the B.Sc. degree in electrical and electronics engineering from Başkent University, in 2000, the M.Sc. degree in electrical and electronics engineering with a robotics option from Middle East Technical University, in 2003, and the Ph.D. degree in biomedical image processing from the School of Electronics and Computer Science, University of Westminster, in 2007. He is currently an Assistant Professor with the Department of AI and Data Engineering, Istanbul Technical University. His research interests include artificial neural networks, medical image analysis, and computer vision.