algorithms

MDPI

Article

Domain-Specific Few-Shot Table Prompt Question Answering via Contrastive Exemplar Selection

Tianjin Mo [1,*], Qiao Xiao [2], Hongyi Zhang [2], Ren Li [2] and Yunsong Wu [3]

[1] Business School, Chongqing College of Electronic Engineering, Chongqing 401331, China
[2] School of Information Science and Engineering, Chongqing Jiaotong University, Chongqing 400074, China; qiaoxiao@mails.cqjtu.edu.cn (Q.X.); hongyizhang@mails.cqjtu.edu.cn (H.Z.); renli@cqjtu.edu.cn (R.L.)
[3] School of Big Data and Software Engineering, Chongqing University, Chongqing 400044, China; lcy@cqu.edu.cn
* Correspondence: 202125007@cqcet.edu.cn

Abstract: As a crucial task in natural language processing, table question answering has garnered significant attention from both the academic and industrial communities. It enables intelligent querying and question answering over structured data by translating natural language into corresponding SQL statements. Recently, there have been notable advancements in the general domain table question answering task, achieved through prompt learning with large language models. However, in specific domains, where tables often have a higher number of columns and questions tend to be more complex, large language models are prone to generating invalid SQL or NoSQL statements. To address the above issue, this paper proposes a novel few-shot table prompt question answering approach. Specifically, we design a prompt template construction strategy for structured SQL generation. It utilizes prompt templates to restructure the input for each test data and standardizes the model output, which can enhance the integrity and validity of generated SQL. Furthermore, this paper introduces a contrastive exemplar selection approach based on the question patterns and formats in domain-specific contexts. This enables the model to quickly retrieve the relevant exemplars and learn characteristics about given question. Experimental results on the two datasets in the domains of electric energy and structural inspection show that the proposed approach outperforms the baseline models across all comparison settings.

Keywords: table question answering; text-to-SQL; few-shot; large language models; prompt learning

Citation: Mo, T.; Xiao, Q.; Zhang, H.; Li, R.; Wu, Y. Domain-Specific Few-Shot Table Prompt Question Answering via Contrastive Exemplar Selection. *Algorithms* **2024**, *17*, 278. https://doi.org/10.3390/a17070278

Academic Editor: Frank Werner

Received: 13 May 2024
Revised: 17 June 2024
Accepted: 21 June 2024
Published: 26 June 2024

1. Introduction

As an important medium for structured data storage and presentation, the two-dimensional table can effectively organize and summarize large quantities of data in different scenarios, presenting them in a concise and clear manner. Building upon this foundation, structured query language (SQL) has emerged to enable rapid operations (such as querying, adding, and deleting) on large volumes of structured tabular data, greatly improving data management and organizational efficiency. However, the correct composition of SQL statements requires humans to learn its specific syntax and become familiar with the tables it operates on, resulting in high learning costs and time consumption. To address this, researchers in natural language processing have proposed the table question answering task, which is also known as text-to-SQL [1]. It aims to translate natural language questions and corresponding table information into SQL statements through the appropriate design of algorithms or models. Due to its ability to intelligently retrieve and analyze structured data, it can be widely applied in various fields such as finance, power, and bridge maintenance, enhancing their levels of digitization and intelligence [2–4].

In recent years, language models trained through the paradigm of "pre-training + fine-tuning" have significantly improved the performance of table question answering tasks by leveraging large-scale pre-training on tables and text [5,6]. In some datasets, these

models have approached human-level performance [7]. However, the aforementioned pre-trained language models require a large amount of annotated data during the fine-tuning process. In some specific domains, obtaining high-quality annotated data may even require hiring domain experts, greatly limiting the development and application of table question answering and similar tasks [8]. To address this issue, some researchers have employed data augmentation or knowledge enhancement methods, effectively reducing the model's reliance on annotated data at the cost of a slight decrease in performance [9–11]. However, due to the limitations of model parameter capacity, the initial language representation and understanding capabilities of these models in table question answering tasks remain relatively low, necessitating the annotation of a portion of the data (100–1000 instances) during the application process to allow the model to fully learn task and domain characteristics.

Benefiting from the advancements in high-performance computing hardware and algorithms, the parameter capacity of pre-trained language models has rapidly increased, significantly improving their language representation and understanding capabilities [12]. OpenAI has discovered that large language models (LLMs) do not require fine-tuning but can instead be driven to answer questions according to human intent by providing a natural language prompt and a few annotated data exemplars (0–5 instances) for the downstream task [13]. This greatly reduces the model's reliance on high-quality annotated data. Among them, generative large language models such as ChatGPT [14], LLaMA [15], and BaiChuan [16] have made significant progress in general domain table question answering and other tasks. However, in specific domains, general purpose language models lack sufficient domain knowledge, and training domain-specific language models to incorporate domain knowledge requires enormous computational resources and massive amounts of training data. Furthermore, for specific table question answering tasks, there may be differences in task characteristics and data distribution between the given annotated data exemplars and the given question, making it difficult to accurately understand the prompt content and potentially resulting in the generation of invalid SQL or NoSQL statements [8]. Therefore, reasonable prompt construction and exemplar selection strategies have become important requirements in domain-specific scenarios where resources are limited.

In this paper, we propose a novel few-shot table prompt question answering approach for the domain-specific databases. Firstly, this approach is based on the opensource Vicuna-13B language model, and it involves redesigning prompt templates to restrict the output format of text-to-SQL tasks, thereby standardizing the model's output to generate valid SQL statements. Secondly, an exemplar selection strategy is employed to assist the model in understanding the current problem and guide the model in completing the prediction task. Lastly, the large language model is used to encode the context and output accurate SQL statements. The main contributions of this paper are as follows:

(1) We design a structured prompt template construction strategy for the table QA task. It utilizes prompt templates to restructure the input for each test data and standardizes the model output, which can improve the integrity and validity of SQL statements generated by the LLMs.

(2) We propose a contrastive exemplar selection strategy covering domain-specific question patterns and methods based on the characteristics of QA tasks and data features, combined with prompt engineering on LLMs. It can enable LLMs to retrieve the most relevant exemplars and fully learn task and data characteristics about given question.

(3) Experimental results on two table QA datasets from the electric energy and structural inspection domain show that our approach outperforms the other baselines in all comparison settings, and achieves the state-of-the-art results.

2. Related Work

2.1. Table QA on General Domains

Before the emergence of pre-trained language models, researchers primarily conducted table question answering studies using seq2seq models [17]. The seq2seq neural network consists of an encoder and a decoder. The encoder is responsible for encoding the input

natural language query and table schema, capturing their semantic and structural information, while the decoder generates the predicted SQL statement. With the widespread application of pre-trained language models such as BERT [5] and GPT [18], the field of table question answering has also gradually explored methods based on pre-trained models. Among them, TABERT [6] encodes natural language texts and structured tables jointly to obtain semantic representations of natural language questions and tables. It introduces the masked column prediction (MCP) and cell value recovery (CVR) training objectives specific to table schemas, effectively improving the performance of general-domain table question answering tasks. Tapas [19] extends the MLM training objective to structured data and predicts answers by selecting subsets of cells and their aggregation operations. Tapas concatenates the question and serialized table as input, and two classification layers predict the selected cell subset and the aggregation operator, respectively. The aggregation operation is then performed on the selected cell subset to obtain the final answer.

However, Tapas cannot handle large tables or databases containing multiple tables, since it processes the entire table in memory as context. TAPEX [7] simulates SQL executors through pre-training, with the actual results of executing SQL queries on tables serving as supervised signals to train the model's generated results. Wang et al. [20] propose an execution-guided strategy to detect and exclude erroneous programs during the decoding process using partially generated SQL, further improving performance. This mechanism can be applied to any autoregressive generation model, assisting most semantic parsing pre-training models in achieving further performance improvements. Hu et al. [21] propose a novel Chinese few-shot text classification method called CIPLUD that combines an improved prompt learning method and existing unlabeled data, which are used for the classification of a small amount of Chinese text data. They used the Chinese pre-trained language model to generate prompt prefixes with multiple masks and constructs suitable prompt templates for Chinese labels. Then, the one-class support vector machine-based unlabeled data leveraging (OCSVM-UDL) module to establish an OCSVM model in the trained text vector space, selects reasonable pseudo-label data for each category from a large amount of unlabeled data. Dang et al. [22] propose to enhance the verbalizer and construct the refined external knowledge into a prompt-tuning (REKP) model. They employ external knowledge bases to increase the mapping space of tagged terms and design three refinement methods to remove noise data.

Currently, table question answering techniques based on pre-trained language models are becoming increasingly mature, with evaluation results on datasets such as WiKiSQL [23] surpassing human accuracy. However, existing methods mainly focus on general domains, and further research is needed for table question answering methods targeting specific domains.

2.2. Table QA on Specific Domains

Currently, table-based question answering methods for general domains have demonstrated strong performance. However, there are certain limitations in terms of scalability and transferability to specific domains. Challenges still exist in domain and task adaptation training strategies, as well as generalization capabilities. To address these challenges, researchers often incorporate specific domain data characteristics and domain expertise to improve the effectiveness of table-based question answering models in specific domains. Specifically, Guo et al. [24] enhanced the prediction accuracy of WHERE clauses by leveraging table content, header, and cell features specific to the electricity domain. Pan et al. [25] combined medical text features and SQL structures to link medical entities and used SQL syntax trees as intermediate representations to reduce the search space for each decoding step, thereby alleviating the issue of mismatch between medical questions and SQL queries. He et al. [26] proposed a divide-and-conquer approach for a class of complex computational queries in the financial domain. They decomposed a row-based computational query into multiple sub-queries, generated corresponding SQL statements for each sub-query, and then combined the sub-query SQL statements to form the original query's SQL statement,

thereby improving the effectiveness of complex queries in the financial domain [27]. Lv et al. [28] addressed the problem of poor interpretability and strong dispersion in the aluminum smelting domain dataset by incorporating factory metadata information during data utilization to resolve issues of inconsistent representation and implied semantics in Chinese column names. However, although the aforementioned work has made progress in table-based question answering tasks in specific domains, they require a significant amount of annotated data for table-based question answering tasks. In some cases, hiring domain experts is necessary to obtain high-quality annotated data, which greatly limits the development and application of table-based question answering and similar tasks. To reduce the data requirements in real-world application scenarios, improve the accuracy of table-based question answering models, and further empower industry applications, further research is needed on few-shot table-based QA in specific domains.

2.3. Few-Shot Table QA

In response to the lack of high-quality annotated data in domain-specific, the emergence of few-shot learning provides a feasible solution [29]. Some studies address the few-shot table QA problem through transfer learning. Ye et al. [30] propose a decomposed prompt learning named entity recognition (NER) framework for few-shot settings, decomposing the NER task into two stages: entity locating and entity typing. In training, the location information of distant labels is used to train the entity locating model. In inference, a pipeline approach is used to handle the entire NER task. Yang et al. [31] proposed SEQZERO, which decomposes the problem into a series of sub-problems, generates concise answers through prompts to predict SQL clauses, and alleviates the combination generalization problem caused by overfitting in few-shot scenarios. Wei et al. [32] introduced a shared mechanism for multitask learning to reduce the complexity of model sub-task decoding and designed a secondary weighted loss to balance the complexity of SELECT and WHERE clauses. However, the approach of sharing decoders weakened the model's expressive power. Change et al. [9] explicitly mapped natural language entities to table headers through auxiliary tasks, defining mapping relationships between condition columns and condition values. However, this approach requires additional BIO data annotation, limiting its practicality. GAZP [33] adapted semantic parsers to new domains through synthetic periodic consistency data, proposing a new paradigm for zero-shot semantic parsing. Additionally, some studies have applied prompt engineering to table QA task, significantly improving the representation performance of generative LLMs [34]. Gu et al. [35] divided table QA into two stages, using prompts to predict SQL structures and fill content separately. Shin et al. [36] proposed a language model-based semantic parsing method, converting questions into controlled sub-languages through dynamic prompts or fine-tuning and constraining decoding to ensure the legality of generated results. Prompts effectively improve the generalization ability of models in few-shot scenarios, but the design of prompt templates requires domain knowledge support and is susceptible to data distribution influences.

In-context learning is a novel approach within the field of natural language processing (NLP) that involves utilizing one or more training exemplars of downstream tasks as input for language models (LMs) [13]. These exemplars are directly decoded to produce outputs without updating the model parameters, which effectively enhance the accuracy of LM outputs through exemplar guidance. However, research conducted by Liu et al. [37] has demonstrated that the performance of downstream tasks can significantly vary depending on the chosen contextual exemplars. Hence, several studies have been dedicated to exploring methodologies for selecting appropriate exemplars to improve the performance of LMs. Rubin et al. [38] proposed an approach that employs the language model itself as a scoring function to retrieve high-quality prompt exemplars from the context. Wang et al. [39] treated large language models as topic models and adopted a strategy to select training data exemplars that are most relevant to the task topic as input. However, the aforementioned methods utilize large models for exemplar selection, which can significantly reduce the

efficiency of inference. In response to this, Das et al. [40] introduced a neural–symbolic exemplar reasoning approach that includes a non-parametric memory network for storing exemplars and a parametric model for retrieving exemplars relevant to the question, thereby generating logical forms. Zhang et al. [41] applied reinforcement learning to treat the selection process of input exemplars as a sequential decision problem and utilized a Markov decision process to select the most relevant input exemplars. Nonetheless, the aforementioned exemplar selection methods are primarily tailored to general domains [42]. The development of exemplar selection algorithms for domain-specific table prompt-based question answering is still in its early stages, and the construction of prompt templates for structured table-based question answering is also yet to be established.

3. Methodology

3.1. Task Formulation

This paper mainly focuses on the domain-specific text-to-SQL task, as referenced in [23], where SQL statements are decomposed into six clauses: SELECT-Column (SC), SELECT-Aggregation (SA), Where-Number (WN), Where-Column (WC), Where-Operator (WO), and Where-Value (WV). These clauses are formally defined as follows: Given a natural language question $Q = \{q_1, q_2, \cdots, q_n\}$ and the corresponding table $T = \{c_1, c_2, \cdots, c_m\}$, the goal is to generate the corresponding SQL query Y. Here, q_i represents the i-th token in the question, n represents the length of Q, c_j represents the j-th column, and m represents the total number of columns.

In few-shot table QA tasks, given a training set of question-SQL pairs $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ for a text-to-SQL task and a test set x_{test}, the objective is to train a retriever $\mathcal{R}(x_{test}, \mathcal{D})$ to retrieve a subset $\mathcal{P} = \{(x_j, y_j)\}_{j=1}^{m} \subset \mathcal{D}$ of training sets, where $m \ll n$. Given an arbitrarily large language model g, efficient prompt engineering concatenates the test set x_{test} with the relevant prompt subset $\mathcal{P}$ as a prefix, passing it to g to complete the target sequence output, resulting in $g([\mathcal{P}; x_{test}])$ decoding to obtain y_{test}. In the text-to-SQL task, x is the natural language expression, and y is the corresponding SQL logical form.

3.2. Data Characteristics in Specific Domains

The text-to-SQL dataset ESQL [43] in the electricity domain is sourced from real power energy agencies. Its tabular data cover multiple dimensions such as time, region, power suppliers, user scale, project progress, enterprise profitability, and more. By analyzing information from these dimensions, various queries and analyses can be performed on electricity data from different perspectives, for example, understanding electricity consumption and price fluctuations during different periods, in different regions, and from different suppliers to evaluate the supply–demand situation, comprehend the scale of the electricity market, and identify growth trends.

The text-to-SQL dataset SMI-SQL in the bridge maintenance domain is established by the research team after conducting multiple text-to-SQL studies on industrial databases. This dataset includes crucial information such as bridge basic information, structural damage descriptions, maintenance and repair recommendations, etc.

The distribution of table column ranges in the ESQL and SMI-SQL datasets is shown in Table 1. The ESQL dataset shows that within the range of 0 to 5 columns, both the quantity and proportion are 0. When the number of columns is between 5 and 10 and 10 and 15, the quantity and proportion are still 0. However, when the number of columns is in the range of 15 to 20, there are 300 instances, accounting for 20% of the total. In contrast, for tables with more than 20 columns, there are 1200 instances, constituting 80% of the total. This indicates that most tables in the ESQL dataset have more than 20 columns and dominate the dataset. On the other hand, the SMI-SQL dataset shows a different distribution. Within the range of 15 to 20 columns, it has the highest proportion, reaching 36.47%. The next highest proportion is in the range of 10 to 15 columns, accounting for 22.80%. In the range of 5 to 10 columns, the proportion is 21.33%, while in the range of 0 to 5 columns, the data

quantity is the lowest, at only 0.93% of the total. It is worth noting that there is still a certain quantity of data with more than 20 columns, accounting for 18.47%.

Table 1. The distribution of table column ranges in the ESQL and SMI-SQL datasets.

Dataset	Number of Columns	0~5	5~10	10~15	15~20	>20
ESQL	Quantity	0	0	0	300	1200
	Proportion	0	0	0	20%	80%
SMI-SQL	Quantity	14	320	342	547	277
	Proportion	0.93%	21.33%	22.80%	36.47%	18.47%

3.3. Architecture

This study integrates heuristic prompt template generation strategies to build a few-shot text-to-SQL model adapted to domain-specific tasks. Based on the Vicuna-13B language model, the model adds prompt-engineered exemplars before input to control the output format. The overall architecture of the model is shown in Figure 1.

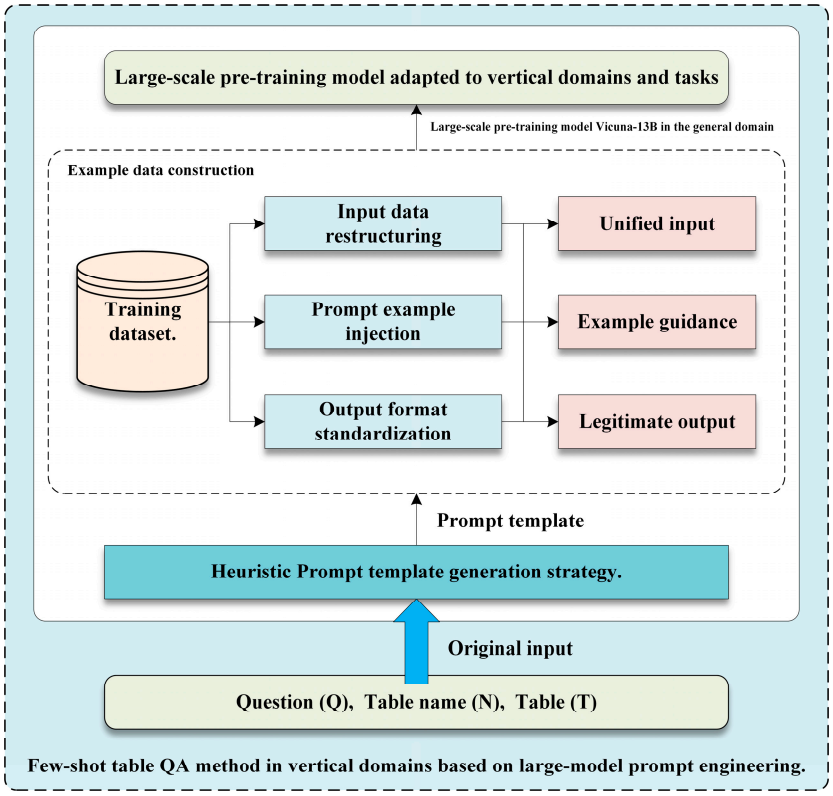

Figure 1. The overview architecture of the proposed model.

As shown in Figure 1, the proposed approach in this study takes the question Q, table name N, and table T as the original inputs of the model. First, the model adopts heuristic prompt template generation strategies to design input–output prompt templates, reconstruct input data, and inject information about aggregation function lists and conditional operators needed for text-to-SQL tasks to achieve input format standardization, denoted as I. Then, based on test set x_{test}, retrieve a subset $\mathcal{P} = \{(x_j, y_j)\}_{j=1}^{m}$ of training set with higher relevance from the training dataset. Concatenate this subset of training samples according to the input format as prompt exemplars to form the complete input I_Final, which guides the model's output. Finally, following the constraints of the prompt output template, the model modularly outputs according to the six clauses used in the evaluation,

generating a complete SQL statement for evaluation. The input, reconstructed based on prompt template engineering exemplars, is fed into the Vicuna-13B large language pre-trained model in the general domain. Through exemplar-guided and output-constrained learning, the model further achieves domain-specific and task adaptation.

3.4. Subsection

Based on the characteristics of text-to-SQL tasks and output formats, a prompt template controlling the input and output of large-scale text-to-SQL models was designed according to the submodule evaluation method. The specific algorithm for the prompt template generation strategy is shown in Algorithm 1

Algorithm 1. The Prompt Template Generation Strategy Algorithm

```
The input information is as follows:
1: Question: {0}
2: Table name: {1}
3: Table headers: {2}
4: Given function list: ["", "AVG", "MAX", "MIN", "COUNT", "SUM"]
5: Given condition operator list: [">", "<", "=", "!="]
6: Return results in the following format:
7: {
8: "SQL": "",
9: "Aggregate function": "",
10: "Query column": "",
11: "Where condition column": [],
12: "Where condition operator": [],
13: "Where condition value": [],
14: "Number of where conditions": ""
15: }
```

The placeholders in the template represent dynamic input information as follows:

1: Question: {0}: Represents the question, i.e., the query posed by the user.

2: Table name: {1}: Represents the table name, i.e., the name of the database table to be queried.

3: Table headers: {2}: Represents the table headers, i.e., the column names of the database table.

The template also provides predefined lists used to generate different parts of the SQL query.

Given Function List: Provides the available aggregation function list, including an empty string (no aggregation function), Average (AVG), Maximum (MAX), Minimum (MIN), Count (COUNT), and Sum (SUM).

Given Condition Operator List: Provides the available condition operator list, including Greater Than (>), Less Than (<), Equal To (=), and Not Equal To (!=).

The template defines an expected result format, including SQL query statement, aggregation function, query column, condition column, condition operator, condition value, and condition count. By filling and replacing placeholders, results are returned in the specified format, generating SQL query statements and related information that meet the requirements and are compliant with the standards for evaluation.

3.5. Exemplar Retrieval Approach

To inject training exemplars targeted to assist the model in understanding the task and standardize model outputs, this paper proposes an exemplar injection prompt template retrieval strategy, which retrieves relevant exemplars to guide model outputs, as shown in Algorithm 2.

Algorithm 2. The Prompt Template Retrieval Strategy

1: Input: Training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$, test set x_{test}.
2: Output: Relevant subset $\mathcal{P} = \left\{\left(x_j, y_j\right)\right\}_{j=1}^{m} \subset \mathcal{D}$ for the test set.
3: (1). Initialize a candidate set of training exemplars $\varepsilon = \mathcal{R}_u((x, y), \mathcal{D})$.
4: (2). Use the large language model g to independently score each $e_l \in \varepsilon$ as $s(e_l) = \text{Prob}_g(y|e_l, x)$.
5: (3). Based on the scores $s(e_l)$ from (2), define a positive exemplar set ε_{pos} and a negative exemplar set ε_{neg} for each e_l.
6: (4). Construct contrastive learning training data $\left\langle x_i, e_i^+, e_{i,1}^-, \cdots, e_{i,2B-1}^-\right\rangle$.
7: (5). Compute the inner product $\text{sim}(x, e) = E_X(x)^T E_P(e)$.
8: (6). Minimize $\text{L}\left(x_i, e_i^+, e_{i,1}^-, \cdots, e_{i,B-1}^-\right) = -\log \frac{e^{\text{sim}(x_i, e_i^+)}}{e^{\text{sim}(x_i, e_i^+)} + \sum_{j=1}^{2B-1} e^{\text{sim}(x_i, e_{i,j}^-)}}$.

The algorithm describes the prompt template retrieval strategy for exemplar injection, where the input consists of a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ and a test set x_{test}, and the output is the relevant subset $\mathcal{P} = \left\{\left(x_j, y_j\right)\right\}_{j=1}^{m}$ of test exemplars, to aggregate exemplar prompts to guide the large language model. The steps of the algorithm are as follows: first, initialize the candidate training dataset $\varepsilon = \mathcal{R}_u((x, y), \mathcal{D})$; then, use the large language model g to independently calculate the relevance probability score $s(e_l)$ of each $e_l \in \varepsilon$ with the test set x_{test}. Specifically, we utilize the output of the large model g to compute $y|e_l$. The computed result is then used to calculate the relevant probability with the test set, yielding the probability of $y|e_l$ being x_{test}. Because we used x_{test} in the input, $s(e_l)$ indicating how helpful each candidate object is for decoding the target exemplar. Next, apply $s(e_l)$ to all training samples and define a positive instance set ε_{pos} and a negative instance set ε_{neg} for each e_l, where ε_{pos} and ε_{neg} respectively consist of the top k candidate instances and the bottom k candidate instances based on the calculated scores $s(e_l)$. Then, construct contrastive learning training data in the form of $\left\langle x_i, e_i^+, e_{i,1}^-, \cdots, e_{i,2B-1}^-\right\rangle$, where the positive instance e_i^+ is a randomly sampled instance from $\varepsilon_{pos}^{(i)}$, while the negative instance is composed of a randomly sampled instance from $\varepsilon_{neg}^{(i)}$, along with $B-1$ positive instances sampled from the same batch and $B-1$ negative instances sampled from the same batch. Finally, compute the inner product $\text{sim}(x, e) = E_X(x)^T E_P(e)$ and minimize the loss to train the exemplar retriever. Here, $E_X(\cdot)$ and $E_P(\cdot)$ represent the input encoder and prompt encoder as described in reference [44].

Based on the detailed explanation of the prompt template generation strategy and the prompt template retrieval algorithm, the exemplar-based injection approach based on large language model prompting engineering is further demonstrated through specific application examples. An example application of ESQL is shown in Algorithm 3, where the input exemplar question is "For projects with an estimated net profit year-on-year growth rate (compared to the previous period) below 44.33% and project estimated total profit (excluding price reduction impact) higher than 3471.44, what is the average project estimated total profit?" The exemplar retrieval strategy proposed in this paper injects exemplars similar in semantics to this question from the same table to guide the model in parsing the text-to-SQL task.

Similarly, Algorithm 4 shows an application example in SMI-SQL. For the question "How many provincial-level departments are there?", the method proposed retrieves the formatted representation of the most relevant question in the training set, "How many municipal-level departments are there?" as a prompt output to guide the model in generating the correct SQL statement.

Algorithm 3. ESQL Application Example

Input	1: Question: For projects with an estimated net profit year-on-year growth rate (compared to the previous period) below 44.33% and project estimated total profit (excluding price re-duction impact) higher than 3471.44, what is the average project estimated total profit? 2: Table Name: table_6
Output	Example 1: 1: Question: What is the lowest estimated net profit year-on-year growth rate (compared to the previous period) in projects with funding source types and year-end balances equal to 3975.46 and 1985.13, respectively? 2: Expected SQL Query Result: 3: { 4: "sql": "select MIN(estimated net profit year-on-year growth (compared to the previous period)) from table_6 where funding source type = '3975.46' and year-end balance = '1985.13'", 5: "Aggregation Function": "MIN", 6: "Query Column": "estimated net profit year-on-year growth (compared to the previous period)", 7: "where condition columns": ["funding source type", "year-end balance"], 8: "where condition operators": ["=", "="], 9: "where condition values": ["3975.46", "1985.13"], 10: "where condition count": "2" 11: }

Algorithm 4. SMI-SQL Application Example

Input	1: Question: How many provincial-level departments are there? 2: Table Name: cbms_dept
Output	Example 1: 1: Question: How many municipal-level departments are there? 2: Expected SQL Query Result: 3: { 4: "sql": "select COUNT(DepartmentCode) from cbms_dept where Department-Level = 'municipal'", 5: "Aggregation Function": "COUNT", 6: "Query Column": "DepartmentCode", 7: "Where Condition Columns": ["DepartmentLevel"], 8: "Where Condition Operators": ["="], 9: "Where Condition Values": ["municipal"], 10: "Where Condition Count": "1" 11: }

4. Experiments

4.1. Experimental Settings

This study conducted extensive experiments on the ESQL and SMI-SQL datasets, injecting 1 prompt training exemplar for each test data to guide the model output. Regarding the selection of the number of examples, we chose one exemplar for each input. The reason for not selecting multiple examples is that, in our preliminary experiments, we tried selecting from 1 to 5 examples and found that choosing one exemplar yielded the best results. Multiple examples led to model learning biases and formatting errors, with five examples exceeding the maximum length limit. It is worth noting that the experiment requires at least one exemplar; for instance, this study uses one exemplar. However, the number of examples needed may vary across different domains and should be determined based on the specific circumstances. During testing, 1500 data points were used. The sample sizes for each dataset used in the experiments are shown in Table 2.

Table 2. ESQL, SMI-SQL dataset few-shot setting.

Dataset	1 Shot	Test
ESQL	1	1500
SMI-SQL	1	1500

The experiments were conducted on a platform based on Linux 5.15.0 Ubuntu 20.04 64-bit operating system and PyTorch 1.13.1 deep learning framework. The hardware environment included an Intel Core i9-10900X CPU, 128GB RAM, 4 NVIDIA RTX-3090Ti 24 GB GPUs, and a 4 TB hard drive. The model's hyperparameter settings are shown in Table 3. Due to the constraints of text length and hardware conditions, the batch size was set to 120. In the preliminary experiments to prevent overfitting, various hyperparameter combinations were tested. To prevent overfitting and achieve optimal performance, the optimizer used was Adam, with a learning rate of 1×10^{-4}, and the number of epochs was set to 30.

Table 3. Hyperparameter settings.

Hyperparameter	Optimizer	Batch Size	Learning Rate	Epoch
Value	Adam	120	1×10^{-4}	30

In this study, we did not employ fine-tuning. Instead, we used a contextual learning approach, which does not suffer from the low generalization issue associated with the fine-tuning paradigm, where parameters cannot be reused. In this approach, the model parameters remain frozen, and the input consists of given samples along with data to be predicted, allowing the model to output the predicted values. Since the model parameters do not change during this process, there is no need to update them when applying to downstream tasks, making it effectively applicable to the tasks addressed in this paper.

To evaluate the performance of the proposed approach, this paper uses Vicuna-13B as the selected LM and compares prompt exemplar selection approaches, including the following: (1) ZERO SHOT: zero-shot setting. This method supports diverse application scenarios, but its performance cannot be guaranteed. (2) RANDOM: random sampling with replacement from the training set D. This method is easy to understand and involves random sampling, but it still faces challenges in handling complex tasks. (3) BM25 [45]: TF-IDF-based sparse retrieval method. This method easily handles documents containing query terms, but it is not good at dealing with semantic relatedness of words. (4) EPR: prompt exemplar retrieval approach proposed in this paper. To maintain consistency with existing research [23], this paper uses text-to-SQL task standard evaluation metrics logical form accuracy (LF) and execution result accuracy (EX) to measure performance. LF compares the synthesized SQL queries by the model with the actual SQL queries, and its calculation method is shown in Formulas (1) and (2).

$$Score_{lf} = \begin{cases} 1, SQL' = SQL \\ 0, SQL' \neq SQL \end{cases} \tag{1}$$

$$Acc_{lf} = \frac{1}{N} \sum_{i=1}^{N} Score_{lf}^{i} \tag{2}$$

where SQL' represents the predicted SQL statement, and SQL represents the true SQL statement. ACC_{lf} represents the logical form matching accuracy, and N is the total number of questions.

EX compares the execution result of the generated SQL statement with the correct answer. Its calculation method is shown in Formulas (3) and (4).

$$Score_{ex} = \begin{cases} 1, Y' = Y \\ 0, Y' \neq Y \end{cases} \tag{3}$$

$$Acc_{ex} = \frac{1}{N}\sum_{i=1}^{N} Score_{ex}^{i} \quad (4)$$

where Y' and Y represent the execution results of the predicted *SQL* statement and the execution results of the true *SQL* statement, respectively, Acc_{ex} represent the accuracy of the execution results, with N being the total number of questions.

Therefore, the experiment comprehensively evaluated the performance of nine tasks, including the syntax legality of the generated SQL (Syn-SQL), LF, and EX, as well as the performance of six sub-tasks (SC, SA, WN, WC, WO, and WV). It is important to note that Syn-SQL is constructed through an exception handling mechanism; that is, if an invalid statement occurs, the system will throw an exception.

4.2. Experimental Results

The experimental results of various prompt template retrieval strategies on the ESQL dataset are compared in Table 4. The evaluation of Syn-SQL under the zero-shot condition is 57.7%, and the legality of each method is significantly improved after injecting exemplars. In particular, when using the EPR approach, the legality score of SQL reaches 92.6%, indicating excellent performance of the EPR retrieval strategy in assisting large language models to improve grammatical correctness. In terms of LF and EX metrics, the EPR approach also performs well, achieving 23.3% and 35.8%, respectively, surpassing the zero-shot 20.2% and 27.2% and significantly outperforming other comparative strategies. This demonstrates that EPR can generate SQL statements that match the query intent. It is worth noting that although EPR's LF and EX are lower compared to Syn-SQL, primarily due to the strong domain-specific nature and complexity of the ESQL and SMI-SQL datasets, this does not negate the practical benefits it brings.

Table 4. Comparison of prompt retrieval strategies in the ESQL dataset.

Models	Syn-SQL	LF	EX
ZERO SHOT	57.7%	3.1%	8.6%
RANDOM	84.9%	9.8%	13.1%
BM25	88.4%	15.4%	27.6%
EPR(Ours)	92.6%	23.3%	35.8%

Table 5 shows the experimental comparison results of various prompt template retrieval strategies on the SMI-SQL dataset. The results indicate that the Syn-SQL score under zero-shot conditions is 58.6%. However, after injecting exemplars, the legality of SQL queries significantly improved. Specifically, the proposed EPR strategy achieved the highest legality score of 95.8%, surpassing other methods significantly. Additionally, the EPR model demonstrated outstanding performance in LF and EX accuracy evaluations, reaching 22.4% and 36.2%, respectively, outperforming the random strategy's 11.9% and 20.9%. This further proves the advantage of the EPR model in generating semantically accurate SQL queries and demonstrates its robustness across multiple domain-specific datasets.

Table 5. Comparison of prompt retrieval strategies in the SMI-SQL dataset.

Models	Syn-SQL	LF	EX
ZERO SHOT	58.6%	2.9%	11.0%
RANDOM	87.9%	10.5%	15.3%
BM25	90.5%	17.8%	33.4%
EPR(Ours)	95.8%	22.4%	36.2%

Table 6 presents the comparison results of exemplar retrieval strategies for submodules in the ESQL dataset. Compared to the zero-shot scenario, the exemplar injection

showed performance improvements for each submodule task. Taking the SC submodule as an example, the random sampling strategy and BM25 strategy improved by 19.5% and 37.3%, respectively, compared to zero-shot. The proposed EPR method achieved a 39.5% improvement. The experimental results demonstrate the significant effectiveness of exemplar injection in improving submodule tasks, especially with the best performance achieved by the EPR method proposed in this paper. This indicates that the EPR method can better utilize exemplar information to enhance the performance of text-to-SQL tasks. The experimental results across different submodule tasks confirm the effectiveness of exemplar injection and provide valuable insights for further research and improvement.

Table 6. Comparison results of exemplar retrieval prompt strategies for ESQL submodules.

Models	SC	SA	WN	WC	WO	WV
ZERO SHOT	22.3%	28.9%	41.4%	17.6%	24.4%	22.9%
RANDOM	41.8%	33.5%	44.6%	27.4%	36.8%	26.4%
BM25	59.6%	37.6%	49.7%	34.6%	37.4%	27.5%
EPR(Ours)	61.8%	44.5%	52.3%	37.5%	45.8%	34.2%

Table 7 presents the comparison results of exemplar retrieval strategies for submodules under the ESQL dataset. The results show the performance scores of various exemplar injection strategies on six text-to-SQL subtasks. In terms of the SC metric, the random model achieved a score of 53.4% compared to zero-shot, an improvement of 8.1 percentage points. The BM25 model scored 58.9% on SC, showing an improvement of 13.6 percentage points compared to zero-shot. On the other hand, the proposed EPR method achieved a score of 63.5% on SC, showing an improvement of 18.2% compared to zero-shot. In terms of the SA metric, the random model and the BM25 model achieved scores of 32.9% and 37.4%, respectively, showing improvements of 8.4% and 12.9% compared to zero-shot. The proposed EPR method, on the other hand, showed an improvement of 19.1%. For the WN, WC, WO, and WV metrics, the random model, BM25 model, and EPR method all exhibited varying degrees of performance improvement. Overall, the experimental results demonstrate that exemplar injection has a significant effect on improving sub-module tasks, especially with our proposed EPR method showing the best performance across multiple metrics. This indicates that the EPR method can better utilize exemplar information to enhance the performance of SMI-SQL tasks.

Table 7. Comparison results of exemplar retrieval prompt strategies for SMI-SQL submodules.

Models	SC	SA	WN	WC	WO	WV
ZERO SHOT	45.3%	24.5%	33.7%	15.6%	20.3%	17.1%
RANDOM	53.4%	32.9%	45.2%	22.8%	32.7%	22.5%
BM25	58.9%	37.4%	47.9%	30.5%	38.9%	27.8%
EPR(Ours)	63.5%	43.6%	50.4%	39.7%	47.5%	33.7%

4.3. Case Analysis

Further qualitatively compare and analyze the impact of zero-shot, random sampling, BM25, and the proposed EPR strategy exemplar selection on downstream text-to-SQL tasks by selecting two examples from ESQL and SMI-SQL, respectively. As shown in Table 8, under the zero-shot condition, the SQL generated by the ESQL dataset is formatted in a disorderly fashion, leading to situations where the generated SQL is NOSQL, and it exhibits a strong degree of randomness, resulting in lower accuracy. In comparison, the random sampling and BM25 strategies generate SQL with higher relevance to the question after injecting samples, but there are still errors in predicting submodules such as condition columns, aggregation functions, etc., leading to SQL errors. However, the proposed EPR strategy in this paper can better understand the problem, provide high-quality exemplars

to guide the large language model's generation, and achieve correct predictions under the above circumstances.

Table 8. Case analysis of comparative models on the ESQL dataset.

Models	Result
Question-1 For an organization with a net profit lower than 4845.59 and a factory reporting a total amount higher than 1383.54, what is the highest amount of goods replenishment?	
Golden	SELECT MAX ('The amount of goods replenishment') FROM table_2 WHERE 'The net profit' < "4845.59" AND 'The factory reporting total amount' > "1383.54"
ZERO SHOT	SELECT MAX ('The amount of goods replenishment') FROM table_2 WHERE 'Institution' < "4845.59" AND 'The factory reporting total amount' > "1383.54" GROUP BY 'Institution'
RANDOM	SELECT ~~MAX~~ ('The amount of goods replenishment') FROM table_2 WHERE 'The net profit' < "4845.59" AND 'The factory reporting total amount' > "1383.54"
BM25	SELECT MAX ('The amount of goods replenishment') FROM table_2 WHERE 'The net profit' < "4845.59" OR 'The factory reporting total amount' > "1383.54"
EPR(Ours)	SELECT MAX ('The amount of goods replenishment') FROM table_2 WHERE 'The net profit' < "4845.59" AND 'The factory reporting total amount' > "1383.54"
Question-2 What is the total amount of non-current assets due within one year outside of Taizhou City?	
Golden	SELECT SUM('The non-current assets due within one year') FROM table_5 WHERE 'City of residence'!="Taizhou"
ZERO SHOT	NOSQL
RANDOM	SELECT SUM ('The non-current assets due within one year') AS 'Sum' FROM table_5 WHERE 'Province of residence.' = "Taizhou" AND 'The non-current assets due within one year' IS NOT NULL GROUP BY 'Province of residence' HAVING 'Province of residence' != "Taizhou"
BM25	SELECT ~~SUM~~ ('The non-current assets due within one year') FROM table_5 WHERE 'City of residence'!="Taizhou"
EPR(Ours)	SELECT SUM ('The non-current assets due within one year') FROM table_5 WHERE 'City of residence'!="Taizhou"

Table 9 presents a comparison of different exemplar retrieval strategies in SMI-SQL. Similar to ESQL, the model's output under zero-shot conditions still includes NOSQL and complex SQL outputs that do not align with the questions. After injecting exemplars using different strategies, the relevance of the model's output to the questions is significantly improved. However, random sampling and BM25 strategies still exhibit some errors in predicting certain sub-modules. In contrast, the output guided by the EPR strategy proposed in this paper per-forms well in terms of sub-module logical form and overall SQL execution result accuracy.

Table 10 presents the case analysis of examples where the proposed EPR method still fails to correctly parse the ESQL and SMI-SQL datasets. Firstly, the proposed method still exhibits sub-module prediction errors such as WHERE clause operators and query columns in text-to-SQL tasks. For instance, in Question-1, the model incorrectly predicted '!=' as '<', while in Question-3, it made an extra prediction of the query column 'level code'. For the simple query in Question 1, the EPR model failed to produce the correct output condition. This is because the "!=" symbol in Question 1 consists of two character symbols. While this combination is straightforward for humans to interpret, it poses a challenge for the

machine model to recognize the judgment operator composed of two symbols. On the other hand, when dealing with complex WHERE conditions, the accuracy of SQL results generated by the model is not satisfactory. As shown in Questions-2 and Questions-4, when understanding complex questions, the model failed to include conditions like 'province' = "Jiangxi" and 'City' = "Yuanzhou District". This indicates that even under the guidance of the EPR method, the large language model still struggles to accurately parse and generate the corresponding number of WHERE conditions, especially when dealing with semantic continuity. In most cases, the model parsed fewer WHERE conditions than the actual count, leading to inaccurate final pre-diction results.

Table 9. Case analysis of comparative models on the SMI-SQL dataset.

Models	Result
Question-1 The expansion joint device on the Dongdagou Bridge was evaluated at what time?	
Golden	SELECT 'Evaluation time' FROM evaluate_bridge_part_evaluation WHERE 'Bridge name' = 'Dongdagou Bridge' AND 'Bridge component name' = "Expansion joint"
ZERO SHOT	NOSQL
RANDOM	SELECT 'Evaluation time' FROM evaluate_bridge_part_evaluation WHERE 'Bridge name' = 'Dongdagou Bridge' AND 'Bridge component code' = "Expansion joint"
BM25	SELECT SUM ('Evaluation time') FROM evaluate_bridge_part_evaluationWHERE 'Bridge name' = 'Dongdagou Bridge' AND 'Bridge component name' = "Expansion joint"
EPR(Ours)	SELECT 'Evaluation time' FROM evaluate_bridge_part_evaluation WHERE 'Bridge name' = 'Dongdagou Bridge' AND 'Bridge component name' = "Expansion joint"
Question-2 The minimum abnormal value collected for anchor cable gauge 28-9 on Yuehai Bridge is how much?	
Golden	SELECT MIN ('The abnormal value collection amount') FROM analy_monitor_type WHERE 'Bridge name' = 'Yuehai Bridge' AND 'Sensor name' = "Anchor cable gauge 28-9"
ZERO SHOT	SELECT MIN ('The abnormal value collection amount') FROM analy_monitor_type WHERE 'Bridge code' = "Yuehai Bridge" AND 'Sensor code' = "Anchor cable gauge 28-9" AND 'The abnormal value collection amount' < (SELECT MIN('The abnormal value collection amount') FROM analy_monitor_type WHERE 'Bridge code' = "Yuehai Bridge" AND 'Sensor code' = "Anchor cable gauge 28-9") GROUP BY 'Sensor code'
RANDOM	SELECT MIN ('The abnormal value collection amount') FROM analy_monitor_type WHERE 'Bridge name' = 'Yuehai Bridge' AND 'Sensor code' = "Anchor cable gauge 28-9"
BM25	SELECT AVG ('The abnormal value collection amount') FROM analy_monitor_type WHERE 'Bridge name' = 'Yuehai Bridge' AND 'Sensor name' = "Anchor cable gauge 28-9"
EPR(Ours)	SELECT MIN ('The abnormal value collection amount') FROM analy_monitor_type WHERE 'Bridge name' = 'Yuehai Bridge' AND 'Sensor name' = "Anchor cable gauge 28-9"

Table 10. EPR case analysis.

Dataset	Models	Result
ESQL		Question-1 The total number of units for which the Engineering Project Management Index is not equal to 3959.82 is how many?
	Golden	SELECT COUNT (‘Company name’) FROM table_4 WHERE ‘the Engineering Project Management Index’ != “3959.82”
	EPR	SELECT COUNT (‘Company name’) FROM table_4 WHERE ‘the Engineering Project Management Index’ < “3959.82”
		Question-2 In companies located in Jiangxi province with a product inventory turnover rate of less than 45.42%, how many regions have operating revenues higher than 4730.85 units?
	Golden	SELECT COUNT (‘Company name’) FROM table_8 WHERE ‘province’ = “Jiangxi” AND ‘The product inventory turnover rate’ < “45.42%” AND ‘The regional operating income’ > “4730.85”
	EPR	SELECT COUNT(*) FROM table_8 WHERE ‘The product inventory turnover rate’ < “45.42%” AND ‘The regional operating income’ > “4730.85” AND ~~‘province’ = “Jiangxi”~~
SMI-SQL		Question-3 The status of anchor cable gauge 45–14 of the main bridge (steel-concrete composite arch bridge) of the Yuehai Bridge is what?
	Golden	SELECT ‘Sensor status’ FROM t_sensor WHERE ‘Bridge name’ = “The main bridge (steel-concrete composite arch bridge) of the Yuehai Bridge” AND ‘Sensor name’ = “The anchor cable gauge 45-14”
	EPR	SELECT ‘Grade code’, ‘Sensor status’ FROM t_sensor WHERE ‘Grade code’ = “45-14” AND ‘Sensor name’ = “The anchor cable gauge”
		Question-4 How many very long tunnels are there in Yuanzhou District?
	Golden	SELECT COUNT (‘Tunnel name’) FROM tunnel_basic WHERE ‘Locality’ = “Yuanzhou District” AND ‘Tunnel classification’ = “Very long tunnels”
	EPR	SELECT COUNT(*) FROM tunnel_basic WHERE ‘Tunnel classification’ = “Very long tunnels” ~~AND ‘Locality’ = “Yuanzhou District”~~

4.4. Model Performance

From the experimental results, it is evident that the exemplar retrieval method proposed in this paper achieved excellent performance on both the ESQL and SMI-SQL datasets, reaching comprehensive optimality. This section will summarize the method’s performance and, to better illustrate the performance comparison of various strategies, visualize the experimental results on the two datasets. As shown in Figures 2 and 3, the method proposed in this paper achieved better results than other comparative methods on both ESQL and SMI-SQL tasks. The proposed strategy retrieval method can select exemplars that are close to the input question in terms of question type and semantics, thus effectively guiding the model output. In contrast, the random sampling method randomly selects exemplars from relevant samples, but its consistency with the current problem cannot be guaranteed, making its guidance relatively weak. The BM25 retrieval strategy only filters based on surface semantic relevance, and when the main subject of the problem changes, the relevance of the selected exemplars may decrease, leading to lower accuracy of the selected exemplars. The prompt retrieval strategy proposed in this paper relies on the large language model itself to score each exemplar and retrieves exemplars with high relevance through comparative learning as input, thus guiding the model output more efficiently during testing.

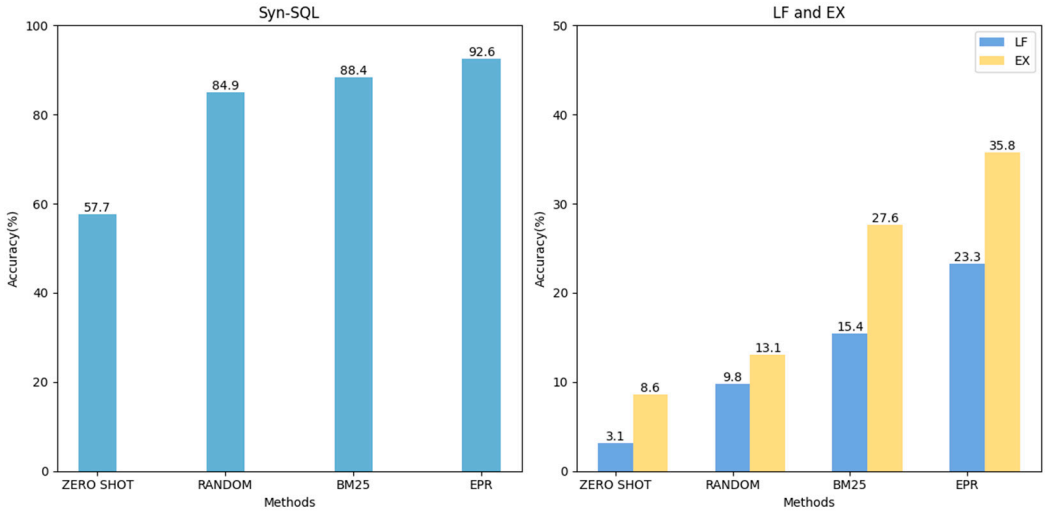

Figure 2. The comparison results of various models on the ESQL dataset.

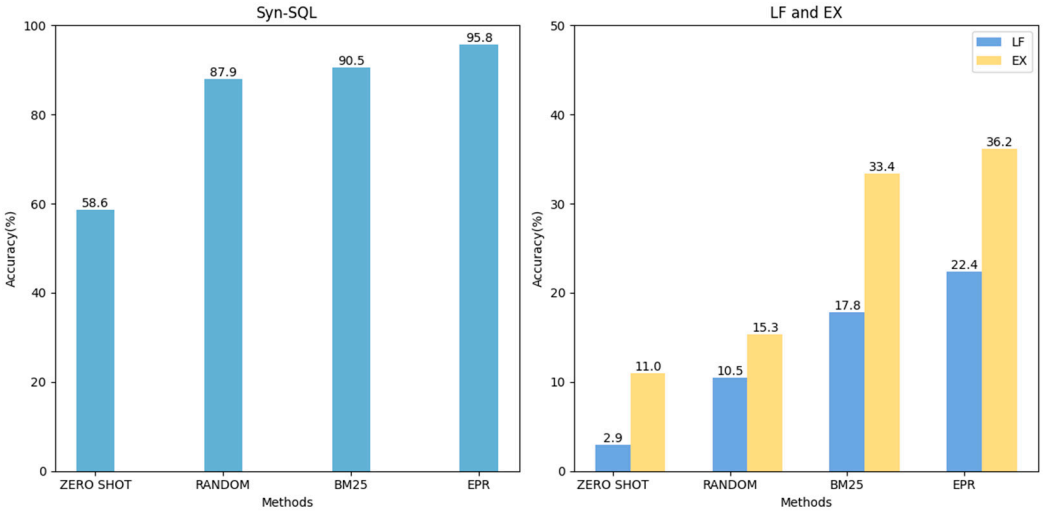

Figure 3. The comparison results of various models on the ESQL dataset.

5. Conclusions

This paper proposes a few-shot table QA method based on prompt engineering on large language models in domain-specific contexts. This method involves restructuring input using input and output prompt templates for all test data, standardizing the model's output, and employing an efficient exemplar retrieval strategy to inject exemplars with the highest relevance to the current question to guide the large language model output, significantly improving accuracy in zero-shot scenarios. We evaluated the proposed method on the ESQL and SMI-SQL datasets and achieved optimal performance, particularly outperforming the compared random sampling strategy and BM25 strategy.

Author Contributions: Methodology, T.M.; validation, T.M. and H.Z.; formal analysis, T.M., Q.X., R.L. and Y.W.; investigation, Q.X.; data curation, Q.X.; writing—original draft preparation, T.M.; writing—review and editing, R.L.; visualization, H.Z.; supervision, Y.W.; funding acquisition, R.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the science and technology research program of the Chongqing Municipal Education Commission of China under grant KJZD-M202300703 and KJQN202200720, the Natural Science Foundation of Chongqing, China, under grant CSTB2023NSCQ-MSX0145.

Data Availability Statement: The data presented in this study are available on request from the corresponding author due to funding agreement restrictions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Meimarakis, G.K.; Koutrika, G. A survey on deep learning approaches for text-to-SQL. *VLDB J.* **2023**, *32*, 905–936. [CrossRef]
2. Wang, L.-J.; Zhang, A.; Wu, K.; Sun, K.; Li, Z.-H.; Wu, H.; Zhang, M.; Wang, H.-F. DuSQL: A large-scale and pragmatic Chinese text-to-SQL dataset. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 8–12 November 2020.
3. Chen, Z.-Y.; Chen, W.-H.; Smiley, C.; Shah, S.; Borova, I.; Langdon, D.; Moussa, R.; Beane, M.; Huang, T.-H.; Routledg, B.; et al. FinQA: A Dataset of Numerical Reasoning over Financial Data. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 7–11 November 2021.
4. Demirhan, H.; Zadrozny, W. Survey of Multimodal Medical Question Answering. *BioMedInformatics* **2023**, *4*, 50–74. [CrossRef]
5. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019.
6. Yin, P.-C.; Neubig, G.; Yih, W.T.; Riedel, S. TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data. In Proceedings of the 2020 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020.
7. Liu, Q.; Chen, B.; Guo, J.-Q.; Ziyadi, M.; Lin, Z.-Q.; Chen, W.-Z.; Lou, J.-G. Tapex: Table pre-training via learning a neural sql executor. In Proceedings of the 2021 International Conference on Learning Representations, Vienna, Austria, 4–8 May 2021.
8. He, B.; Dong, B.; Guan, Y.; Yang, J.-F.; Jiang, Z.-P.; Yu, Q.-B.; Cheng, J.-Y.; Qu, C.-Y. Building a comprehensive syntactic and semantic corpus of Chinese clinical texts. *J. Biomed. Inform.* **2017**, *100*, 203–217. [CrossRef] [PubMed]
9. Chang, S.-C.; Liu, P.-F.; Tang, Y.; Huang, J.; He, X.-D.; Zhou, B.-W. Zero-shot text-to-SQL learning with auxiliary task. In Proceedings of the 2020 AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020.
10. Wang, J.-N.; Wang, C.-Y.; Qiu, M.-H.; Shi, Q.-H.; Wang, H.-B.; Huang, J.; Gao, M. KECP: Knowledge Enhanced Contrastive Prompting for Few-shot Extractive Question Answering. In Proceedings of the 2022 Conference on Empirical Methods in Natu-ral Language Processing, Abu Dhabi, United Arab Emirates, 7–11 December 2022.
11. Xu, W.-W.; Li, X.; Zhang, W.-X.; Zhou, M.; Lam, W.; Si, L.; Bing, L.-D. From Cloze to Comprehension: Retrofitting Pre-trained Masked Language Models to Pre-trained Machine Reader. In Proceedings of the 2023 Thirty-seventh Conference on Neural Information Processing Systems, New Orleans, LA, USA, 10–16 December 2023.
12. Shoeybi, M.; Patwary, M.; Puri, R.; Gresley, P.L.; Casper, J.; Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv* **2019**, arXiv:1909.08053.
13. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. In Proceedings of the 2020 34th International Conference on Neural Information Processing Systems, Online, 6–12 December 2020.
14. Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. Train-ing language models to follow instructions with human feedback. In Proceedings of the 2022 Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022.
15. Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv* **2023**, arXiv:2307.09288.
16. Yang, A.-Y.; Xiao, B.; Wang, B.-G.; Zhang, B.-R.; Bian, C.; Yin, C.; Lv, C.-X.; Pan, D.; Wang, D.; Yan, D.; et al. Baichuan 2: Open large-scale language models. *arXiv* **2023**, arXiv:2309.10305.
17. Vincenzo, S.; Licia, S.; Roberto, T. A primer on seq2seq models for generative chatbots. *ACM Comput. Surv.* **2023**, *56*, 1–58.
18. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding with unsupervised learning. *Citado* **2018**, *17*, 1–12.
19. Herzig, J.; Nowak, P.K.; Müller, T.; Piccinno, F.; Eisenschlos, J. TaPas: Weakly Supervised Table Parsing via Pre-training. In Proceedings of the 2020 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020.
20. Wang, C.-L.; Tatwawadi, K.; Brockschmidt, M.; Huang, P.-S.; Mao, Y.; Polozov, O.; Singh, R. Robust text-to-sql generation with execution-guided decoding. *arXiv* **2018**, arXiv:1807.03100.
21. Hu, T.; Chen, Z.; Ge, J.; Yang, Z.; Xu, J. A Chinese few-shot text classification method utilizing improved prompt learning and unlabeled data. *Appl. Sci.* **2023**, *13*, 3334. [CrossRef]
22. Dang, Y.; Chen, W.; Zhang, X.; Chen, H. REKP: Refined External Knowledge into Prompt-Tuning for Few-Shot Text Classification. *Mathematics* **2023**, *11*, 4780. [CrossRef]
23. Zhong, V.; Xiong, C.-M.; Socher, R. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv* **2017**, arXiv:1709.00103.
24. Guo, X.-N.; Chen, Y.-R.; Qi, G.-L.; Wu, T.-X.; Xu, H. Improving Few-Shot Text-to-SQL with Meta Self-Training via Column Specificity. In Proceedings of the 2022 International Joint Conference on Artificial Intelligence, Vienna, Austria, 23–29 July 2022.
25. Pan, Y.-C.; Wang, C.-H.; Hu, B.-T.; Xiang, Y.; Wang, X.-L.; Chen, Q.-C.; Chen, J.-J.; Du, J.-C. A BERT-based generation model to transform medical texts to SQL queries for electronic medical records: Model development and validation. *JMIR Med. Inform.* **2021**, *9*, e32698. [CrossRef] [PubMed]
26. He, J.-H.; Liu, X.-P.; Shu, Q.; Wan, C.-X.; Liu, D.-X.; Liao, G.-Q. SQL generation from natural language queries with complex calculations on financial dat. *J. Zhejiang Univ. (Eng. Sci.)* **2023**, *57*, 277–286.

27. Lin, P.-G.; Li, Q.-T.; Zhou, J.-Q.; Wang, J.-H.; Jian, M.-W.; Zhang, C. Financial Forecasting Method for Generative Adversarial Networks based on Multi-model Fusion. *J. Comput.* **2023**, *34*, 131–145.
28. Lv, J.-Q.; Wang, X.-B.; Chen, G.; Zhang, H.; Wang, M.-G. Chinese Text-to-SQL model for industrial production. *J. Comput. Appl.* **2022**, *42*, 2996–3002.
29. Lu, J.; Gong, P.-H.; Ye, J.-P.; Zhang, J.-W.; Zhang, C.-S. A survey on machine learning from few samples. *Pattern Recognit.* **2023**, *139*, 109480. [CrossRef]
30. Ye, F.; Huang, L.; Liang, S.; Chi, K. Decomposed Two-Stage Prompt Learning for Few-Shot Named Entity Recognition. *Information* **2023**, *14*, 262. [CrossRef]
31. Yang, J.-F.; Jiang, H.-M.; Yin, Q.-Y.; Zhang, D.-Q.; Yin, B.; Yang, D.-Y. SEQZERO: Few-shot Compositional Semantic Parsing with Sequential Prompts and Zero-shot Models. In Proceedings of the 2022 Association for Computational Linguistics: NAACL 2022, Online, 10–15 July 2022.
32. Wei, C.; Huang, S.-B.; Li, R.-S. Enhance text-to-SQL model performance with information sharing and reweight loss. *Multimed. Tools Appl.* **2022**, *81*, 15205–15217. [CrossRef]
33. Zhong, V.; Lewis, M.; Wang, S.I.; Zettlemoyer, L. Grounded Adaptation for Zero-shot Executable Semantic Parsing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 8–12 November 2020.
34. Nan, L.-Y.; Zhao, Y.-L.; Zou, W.-J.; Ri, N.; Tae, J.; Zhang, E.; Cohan, A.; Radev, D. Enhancing Text-to-SQL Capabilities of Large Language Models: A Study on Prompt Design Strategies. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Singapore, 6–10 December 2023.
35. Gu, Z.-H.; Fan, J.; Tang, N.; Cao, L.; Jia, B.-W.; Madden, S.; Du, X.-Y. Few-shot text-to-sql translation using structure and content prompt learning. *ACM Manag. Data* **2023**, *1*, 1–28. [CrossRef]
36. Shin, R.; Lin, C.; Thomson, S.; Chen, C.; Roy, S.; Platanios, E.A.; Pauls, A.; Klein, D.; Eisner, J.; Durme, B.V. Constrained Language Models Yield Few-Shot Semantic Parsers. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 7–11 November 2021.
37. Liu, J.-C.; Shen, D.-H.; Zhang, Y.-Z.; Dolan, B.; Carin, L.; Chen, W.-Z. What Makes Good In-Context Examples for GPT-3? In Proceedings of the 2022 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, Dublin, Ireland, 26–27 May 2022.
38. Rubin, O.; Herzig, J.; Berant, J. Learning to Retrieve Prompts for In-Context Learning. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 10–15 July 2022.
39. Wang, X.-Y.; Zhu, W.-R.; Saxon, M.; Steyvers, M.; Wang, W.-Y. Large Language Models Are Latent Variable Models: Explaining and Finding Good Demonstrations for In-Context Learning. In Proceedings of the 2023 Thirty-seventh Conference on Neural Information Processing Systems, New Orleans, LA, USA, 10–16 December 2023.
40. Das, R.; Zaheer, M.; Thai, D.; Godbole, A.; Perez, E.; Lee, J.Y.; Tan, L.-Z.; Polymenakos, L.; McCallum, A. Case-based Reasoning for Natural Language Queries over Knowledge Bases. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 7–11 November 2021.
41. Zhang, Y.-M.; Feng, S.; Tan, C.-H. Active Example Selection for In-Context Learning. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, 7–11 December 2022.
42. Liu, D.; Yao, S.-W.; Zhao, H.-L.; Sui, X.; Guo, Y.-Q.; Zheng, M.-L.; Li, L. Research on Mutual Information Feature Selection Algorithm Based on Genetic Algorithm. *J. Comput.* **2022**, *33*, 131–143.
43. Chen, Y.-R.; Guo, X.-N.; Wang, C.-J.; Qiu, J.; Qi, G.-L.; Wang, M.; Li, H.-Y. Leveraging table content for zero-shot text-to-sql with meta-learning. In Proceedings of the 2021 AAAI Conference on Artificial Intelligence, Virtually, 2–9 February 2021.
44. Karpukhin, V.; Oğuz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.-Q.; Yih, W.T. Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 8–12 November 2020.
45. Stephen, R.; Hugo, Z. The probabilistic relevance framework: BM25 and beyond. *Found. Trends® Inf. Retr.* **2009**, *3*, 333–389.