Research on the Text2SQL Method Based on Schema Linking Enhanced

Jun Tie[1,2], Boer Zhu[1,2](✉), Chong Sun[1,2], and Ziqi Fan[1,2]

[1] College of Computer Science, South-Central Minzu University, Wuhan 430074, China
aulaia_zhu@163.com

[2] Hubei Provincial Engineering Research Center for Intelligent Management of Manufacturing Enterprises, Wuhan 430074, China

Abstract. In recent years, natural language generation of SQL sentences (Text2SQL) has received a lot of attention as an important research direction natural language processing (NLP). Text2SQL makes it easier for users to query complex databases without learning SQL sentences and the underlying database schema. The current mainstream Text2SQL method is the grammar-based IRNet, which attempts to present an efficient method with explanations from the perspective of constructing reasonable grammars and provides a good solution for solving complex nested queries. Still, it makes simple use of external database ontology knowledge, resulting in natural language problems in which words do not correspond well to tables and columns in the database. To address this problem, a new method that considers the entity relationships between natural language problems and data in the database - SLESQL is proposed, which extends some of the functionality of IRNet by using schema linking enhanced Experiments show that SLESQL achieves 6.8% improvement in accuracy over IRNet on the publicly available dataset Spider.

Keywords: Deep learning · IRNet model · Text2SQL · schema linking

1 Introduction

Most of the previous studies have focused on converting natural language problems into SQL query statements that can be executed by existing database software, and now providing users with similar convenient interfaces to query data directly through natural language is the focus of any data opening work that cannot be ignored [1].

The typical Example of Text2SQL is shown in Fig. 1. An effective SQL query sentence is generated by a natural language query sentence, the corresponding database schema (DB Schema), and database content. For example, a query sentence: "Show origin and destination for flights with a price higher than 300." reflects some problems faced by the current Text2SQL method. In fact, "higher"

Industry-University-Research Innovation Fund Project of Science and Technology Development Center of Ministry of Education (No. 2020QT08).

is supposed to refer to the height column (not in table flight) and cannot be extracted directly from the problem. There is also another case: "flights" refers to table flights, and the tag "flights" in natural language problems cannot be mapped directly to a value in a column, table, or database content in a database.

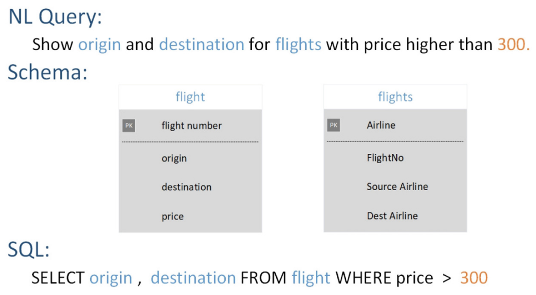
NL Query:
Show origin and destination for flights with price higher than 300.
Schema:
flight
PK
flight number
origin
destination
price
flights
PK
Airline
FlightNo
Source Airline
Dest Airline
SQL:
SELECT origin , destination FROM flight WHERE price > 300

Fig. 1. A Typical Example of Text2SQL

Choosing the correct columns, tables, and values in the data tables involves a major challenge for Text2 SQL - Schema Linkling [4]. Schema linking refers to identifying references of columns, tables, and conditional values in the natural language query sentence. Schema linking involves three difficulties. Firstly, any Text2SQL model must encode the database schema into a representation suitable for decoding into SQL queries, and the decoding process may involve given columns or tables. Secondly, these represent all the information encoded about the schema, such as its column type, foreign key relationship, and the main key used for table connection in the database, as well as external knowledge not in the database and natural language queries, such as the entity type and relationship of each word. Finally, even if the table names, column names, and values in the data tables in the query are different from those encountered during training, the model must identify natural language queries that refer to the values in the table names, column names, and data tables.

Given the above difficulties, we start from the schema linking so that the model can not only extract important information from natural language query sentences but also take into account the relevance of database schema and database content. At the same time, a new named entity recognition method is used to provide more sufficient external knowledge for the model, and a complete candidate strategy is designed to help model selection. Finally, the model generates correct executable SQL query sentences.

2 Related Work

Most current methods use advanced neural network architecture to synthesize SQL queries for given user problems. The SQLOVA method developed by Hwang et al. [5] in 2019 introduces BERT preprocessing model. IRNet [7] used a Transformer encoder and decoder based on an LSTM network, and the general processing flow is shown in the blue part of Fig. 2. In the stage of schema linking, the segmentation method of IRNet is only based on the N-gram method of the string. This will cause the input information to be redundant, and a large number of invalid word segmentations are input, which increases the difficulty of neural network selection. At the same time, IRNet uses a largescale open knowledge map ConceptNet to predict the relationship between the values in the data table in the natural language query sentence and the columns in the database model and puts the fragments in the natural language query sentence into the knowledge map ConceptNet. The results returned by ConceptNet contain two noteworthy information, namely, "relevant terms" and "the same type". This only by looking up the "upper words" to filter the database schema matching column name effect is not ideal, can not accurately understand the relationship between the data table median and the column in the database schema, and because of some spelling errors and ignore the relevant words, thus reducing the accuracy of generating SQL.

We propose a Text2SQL model based on schema linking enhanced (SLESQL) by combining external knowledge and database content. Starting from the schema linking, a new candidate strategy is adopted to make up for the defects of single string comparison (such as N-gram). It not only considers the association between natural language query sentences and database schema, but also considers linking the values in the data table to the database schema so that the model can extract the correct entity type, enhance the reasoning ability of the model, and improve the accuracy of generating SQL sentences.

3 Algorithm Description

The input of IRNet is the natural language query sentence and database schema, and the SLESQL proposed in our work adds the database content (DB Content) as the input on this basis, as shown in the orange section in Fig. 2. In the encoder, SLESQL introduces the named entity recognition model-TENER [8] to enhance the ability of the model to extract the correct entity. After screening and verification of candidate sets, the encoder can more accurately align the database schema with each part of the natural language query sentence, and the process can be seen in Fig. 3.

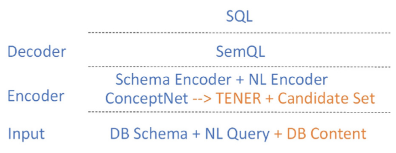
SQL
Decoder
SemQL
Schema Encoder + NL Encoder
Encoder
ConceptNet --> TENER + Candidate Set
Input
DB Schema + NL Query + DB Content

Fig. 2. SLESQL Overall Flow

3.1 Schema Linking Enhanced

The schema linking enhanced phase mainly completes two tasks : (1) Propose an appropriate candidate set. (2) Provide adequate information for predicting values in table names, column names, and data tables. In addition to the database schema and natural language query sentences, the model can predict the correct table name, column name, and value in the data table in the decoder by viewing the database content to provide more information for the neural network.

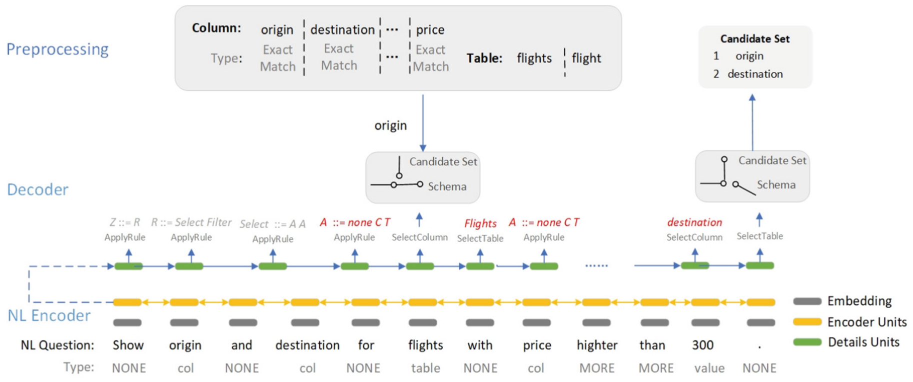

Fig. 3. Schema Linking Enhanced

Candidate Extraction. For a natural language query sentence, we use the named entity recognition (NER) model to extract potential candidates. Since Spider is an English dataset, the TENER model proposed by Yan et al[17] in 2019 has also been applied in our work. This model is an improvement on the original Transformer based on the NER task. The improved Transformer encoder is used to model the character-level features and word-level features. The TENER model is the best indepth learning method on the English dataset CoNLL-2003 so far. we also make an entity analysis of the Question section in Spider [2]. In addition, some simple and effective heuristic methods are used to extract and generate candidates : (1) Content in quotation marks. (2) Terms in capital letters. (3) Single alphabet.

Candidate Generation. After extracting the value from the problem, the candidate needs to be generated. For the value in the data table, the extracted value itself is likely to be the only necessary candidate. Three candidate value generation methods are used in SLESQL : (1) method based on string similarity, (2) heuristic method based on artificial production, and (3) N-gram method based on a string. To evaluate the similarity between the text value extracted from the problem and the value in the database, using the text distance based on word embedding, the model only scans the value of the similarity in the database that exceeds a certain threshold. we further use Damerau-Levenshtein [9] to measure the similarity between each marker, because Damerau-Levenshtein has a good performance between accuracy and running time.

Verify Candidate and Encode. Since candidate generation will generate a large number of potential candidates, it is necessary to verify the candidate set. Three aspects should be considered : (1) the similarity threshold, (2) the number of candidates extracted from natural language problems, and (3) the total number of values in the database. Since the number of candidates directly affects the accuracy of the model, too many candidates make it difficult for the model to select the correct value. Therefore, this article uses database content again to reduce the number of candidate sets. In contrast, Exact Matching is used instead of using similarity to verify the candidate set. We exclude candidates extracted from quotation marks from database validation. In the validation process of the candidate set, the SLESQL model will record the table column position of the candidate in the database.

The work of candidate encoding is similar to that of the table and column encoding. The position of the candidate (table and column) is encoded with the candidate itself. Because of the extra table and column information, not only for the numerical itself, the encoder can also find the location of the numerical attention. Each candidate, together with its location, is separated from other values by using the [SEP] specified by the encoder. Each marker value is further marked as a lexical chunk using the WordPiece segmentation algorithm. Encoder input is a pretrained embedded list, and each lexical block corresponds to it.

3.2 Encoder-Decoder

SLESQL encoder input is information about database schema and candidates extracted from database content. Therefore, SLESQL encoders can also learn the relationship between the tag of natural language problems and the actual values in the database. The nonoverlapping sequence of queries is expressed as $x = [(x_1, T_1), \ldots, (x_n, T_n)]$, where x_1 is the first sequence and is the type of x_1. The encoder takes as input and converts each word into its vector. Then, running bidirectional LSTM for all sequences, the output states of forwarding LSTM and reverse LSTM are connected as the output of natural language query sentences in the encoder. The database model is expressed as a set of different columns and their types assigned in the preprocessing, which is a set of tables.

The decoder receives the encoding of natural language query questions, table names, column names, and data table medians from the encoder as the input, and the output is the synthesized close semantic query language (SemQL). The decoder is composed of LSTM architecture and multiple pointer networks, which are used to select tables, columns, and values. Using a syntax-based decoder (TRANX), LSTM is used to simulate the generation process of SemQL. Formally, the generation process y of SemQL can be formalized as $p(y \mid x, s) = \prod_{i=1}^{T} p(a_i \mid x, s, a < i)$, where is the action taken at the time i, $a < i$ is the action sequence before i, and is the total number of the whole action sequence. The decoder interacts with three types of actions to generate SemQL, including APPLYRULE, SELECTCOLUMN, and SELECTTABLE

[10]. APPLYRULE(r) applies a generation ruler to the current derivation tree of SemQL, where r is the generation rule designed in IRNet. Also, using a Coarse-to-Fine framework [11], the decoding process for SemQL is shown in Fig. 4.

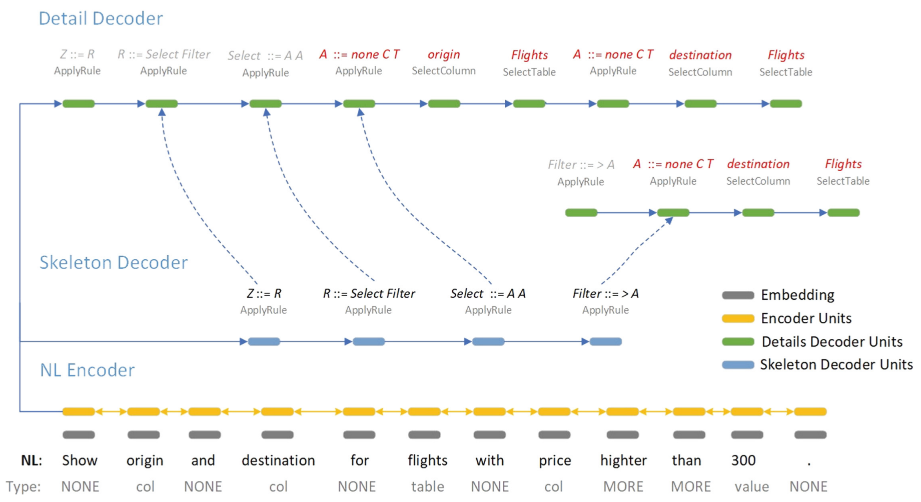

Fig. 4. Encoder-Decoder Processing Flow

4 Experiments and Analysis

The experiment uses the Spider public dataset of Yale University and compares SLESQL with SyntaxSQLNet [3] and IRNet to verify the effectiveness of SLESQL.

4.1 Dataset and Evaluating Indicator

The experiment uses the public dataset Spider, which contains most SQL operators (ORDER BY/GROUP BY/HAVING and nested queries) and is distributed in 200 publicly available databases in 138 fields. Each database has multiple tables, and each database has an average of 5.1 tables. The dataset is divided into a training set, validation set, and test set. The validation set covers 20 different databases that have not appeared in the training set.

To quantitatively evaluate the accuracy of SQL generated by Text2SQL, this paper conducts a comparative test based on the Spider dataset. Exact Matching index is used for evaluation. If the complete statement is correct, the accuracy rate is used as a measure. The specific calculation formula is shown in Formula (1).

$$\text{Exact Matching} = \frac{\#\text{count}}{\#\text{total}} \tag{1}$$

where *#total* is the total number of samples, and *#count* is the consistent number of complete SQL sentences.

4.2 Experiment Setting

The model uses an Adam optimizer, the encoder is 2e–5, the decoder is 1e–3, and the intermediate connection parameter is 1e–4. The learning rate of the encoder is the default parameter for BERT [6] fine-tuning, and all other hyperparameters are set based on experience hyperparameters. After experimental verification, the optimal parameter configuration is as follows: $\mathrm{Batch_size} = 64, \mathrm{Dropout} = 0.3$.

4.3 Experimental Results and Analysis

To verify the effectiveness of SLESQL proposed in this paper, SyntaxSQLNet and IRNet models are mainly used as the comparison models of experiments. Spider evaluation index defines difficulty according to the number of SQL components, selection and conditions, so queries containing many SQL keywords will be considered difficult. Spider defines four difficulty levels, simple, medium, Hard and Extra Hard. According to the difficulty level of SQL defined in the Spider dataset, this experiment further studies the performance of SLESQL in different difficulty parts of the test set.

Table 1. SLESQL Exact Match Results in Four Difficulty Levels on the Test Set

Algorithms	Easy	Medium	Hard	Extra Hard
SyntaxSQLNet(BERT)	42.9%	24.9%	21.9%	8.6%
IRNet(BERT)	77.2%	58.7%	48.1%	25.3%
SLESQL	77.6%	60.1%	52.2%	32.1%

As shown in Table 1, SLESQL has higher accuracy than SyntaxSQLNet and IRNet at all difficulty levels. Moreover, the experimental data show that for the more Hard queries, SLESQL is more sufficient to show excellent analytical power and accuracy. For example, compared with IRNet (with BERT to enhance IRNet), SLESQL improves the matching rate of SQL sentences at the Extra Hard level by 6.8%. Experiments show that schema linking enhanced can better solve the problem of containing more database schemas and data tables.

Some problems can be solved by BERT, such as field prediction error and operator misuse, but the improvement for complex nested queries is not obvious. In Fig. 5, we can see that the highest accuracy of SLESQL is 65% in the Spider training set. Moreover, with the increase of training rounds, the accuracy of SQL generated by the model steadily increases. After the convergence in the late training, SLESQL has a higher performance. Similarly, using BERT as the pretraining model, SLESQL performs significantly better than IRNet, indicating that the research work on SLESQL in complex nested queries is effective.

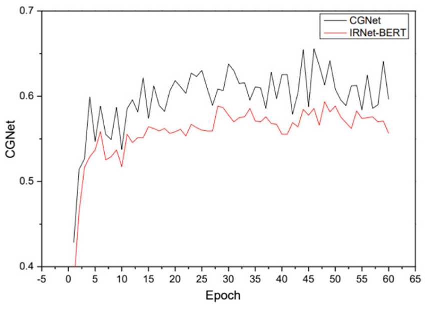

Fig. 5. Encoder-Decoder Processing Flow

5 Summary

In this paper, the method of schema linking enhanced is proposed to link the values in the data table with the database schema, to propose the correct candidate set. The neural network is used to determine the most consistent with the problem intention in these candidate sets as the filling value in the decoder, aiming to solve the mismatch problem between the schema linking and the intermediate representation. The experiment proves the effectiveness of SLESQL in the Spider dataset under complex cross-domain scenarios, which enables the technology to quickly adapt to the enterprise database, simplify the process of data fetching and text analysis, help enterprises freely interact with the database, and effectively activate the knowledge value of the enterprise database.

Acknowledgments. This work is supported and assisted by the Industry-University-Research Innovation Fund Project of Science and Technology Development Center of Ministry of Education (No. 2020QT08), National Ethnic Affairs Commission of the People's Republic of China (Training Program for Young and Middle-aged Talents, MZR20007) and the Hubei Provincinal Science and Technology Major Project of China (2020AEA011).

References

1. Pliego, P.I.: Programming an intelligent personal assistant for biomedical database analysis. Universitat Politècnica de Catalunya, Catalonia (2017)
2. Yu, T., Zhang, R., Yang, K.: Spider: a large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3911–3921. ACL, Brussels (2018)
3. Yu, T., Yasunaga, M., Yang, K.: SyntaxSQLNet: syntax tree networks for complex and cross-domain text-to-SQL task, pp. 1653–1663. ACL, Brussels (2018)
4. Taniguchi, Y., Nakayama, H., Takahiro, K.: an investigation between schema linking and text-to-SQL performance https://arxiv.org/abs/2102.01847
5. Hwang, W., Yim, J., Park, S.: A comprehensive exploration on WikiSQL with table-aware word contextualization https://arxiv.org/abs/1902.01069

6. Devlin, J., Chang, M.W., Lee, K. and Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–4186. ACL, Minneapolis (2019)
7. Guo J, Zhan Z, Gao Y.: Towards complex text-to-SQL in cross-domain database with intermediate representation, pp. 4524–4535. ACL, Florence (2019)
8. Yan, H., Deng, B., Li, X.: TENER: adapting transformer encoder for named entity recognition https://arxiv.org/abs/1911.04474
9. Damerau, F.J.: A technique for computer detection and correction of spelling errors. Commun. ACM **7**(3), 171–176 (1964)
10. Yin, P., Neubig, G.: A syntactic neural model for general-purpose code generation. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 440–450. ACL, Vancouver (2017)
11. Dong, L., Lapata, M.: Coarse-to-fine decoding for neural semantic parsing. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pp. 731–742. ACL, Melbourne (2018)