Leveraging Table Content for Zero-shot Text-to-SQL with Meta-Learning

35th AAAI Conference on Artificial Intelligence

Yongrui Chen[1], Xinnan Guo[1], Chaojie Wang[2], Jian Qiu[2], Guilin Qi[1] & Huiying Li[1]

[1] School of Computer Science and Engineering, Southeast University

[2] Alibaba Group

{220171664, gqi, huiyingli}@seu.edu.cn
{chaojie.wcj, qiujian.qj}@alibaba-inc.com
guoxinnan0727@163.com

Abstract

Single-table text-to-SQL aims to transform a natural language question into a SQL query according to one single table. Recent work has made promising progress on this task by pre-trained language models and a multi-submodule framework. However, zero-shot table, that is, the invisible table in the training set, is currently the most critical bottleneck restricting the application of existing approaches to real-world scenarios. Although some work has utilized auxiliary tasks to help handle zero-shot tables, expensive extra manual annotation limits their practicality. In this paper, we propose a new approach for the zero-shot text-to-SQL task which does not rely on any additional manual annotations. Our approach consists of two parts. First, we propose a new model that leverages the abundant information of table content to help establish the mapping between questions and zero-shot tables. Further, we propose a simple but efficient meta-learning strategy to train our model. The strategy utilizes the two-step gradient update to force the model to learn a generalization ability towards zero-shot tables. We conduct extensive experiments on a public open-domain text-to-SQL dataset WikiSQL and a domain-specific dataset ESQL. Compared to existing approaches using the same pre-trained model, our approach achieves significant improvements on both datasets. Compared to the larger pre-trained model and the tabular-specific pre-trained model, our approach is still competitive. More importantly, on the zero-shot subsets of both the datasets, our approach further increases the improvements.

Introduction

In this paper, we propose a new approach called *Meta-Content text-to-SQL* (MC-SQL) to handle zero-shot tables. The motivation comes from the following two intuitions: 1) The first one is that table content can provide abundant information for predicting headers. Figure 1 shows an example. The cell *son* in the table is relevant to the question word "son", thus reveals the potential header *Relationship to Monarch*. 2) The second one is that meta-learning can help the model learn the generalization ability between different tables from the training data. It is because meta-learning has the capability that only needs a few gradient steps to quickly adapt to new tasks.

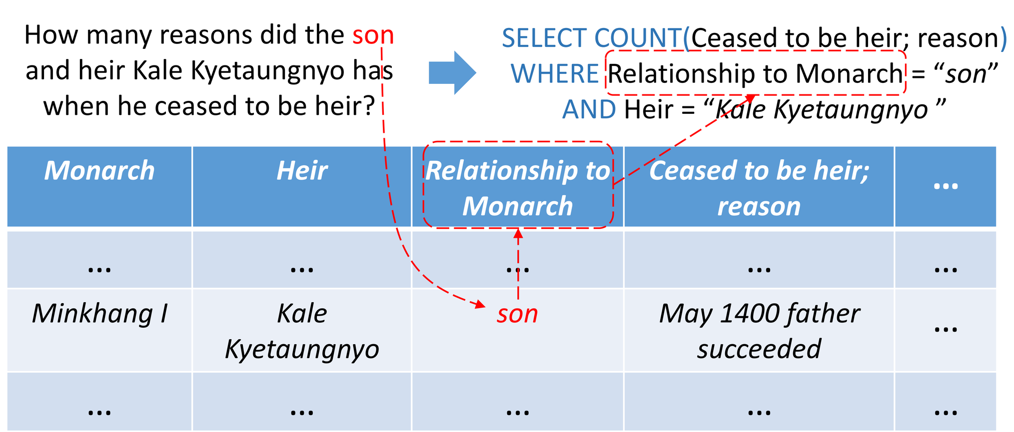

Figure 1: An example of table content to help predict headers. Red indicates the matching.

To comprehensively evaluate our approach, we conduct experiments on public open-domain benchmark WikiSQL and domain-specific benchmark ESQL. Our approach achieves a significant improvement over the baselines that utilizes the same pre-trained model as ours, and also achieves competitive results over the baselines that utilize the larger or tabular-specific pre-trained model.

Process Overview

The framework of our approach is shown in Figure 2, which consists of two parts. First, the *table content enhanced model* (left) captures the semantic relevance of questions with headers and cells at the same time, and predict subtasks comprehensively. Further, *zero-shot meta-learning* (right) is leveraged to train the table content enhanced model. In each training batch, the model parameters are updated in two stages to force the model to learn the generalization ability.

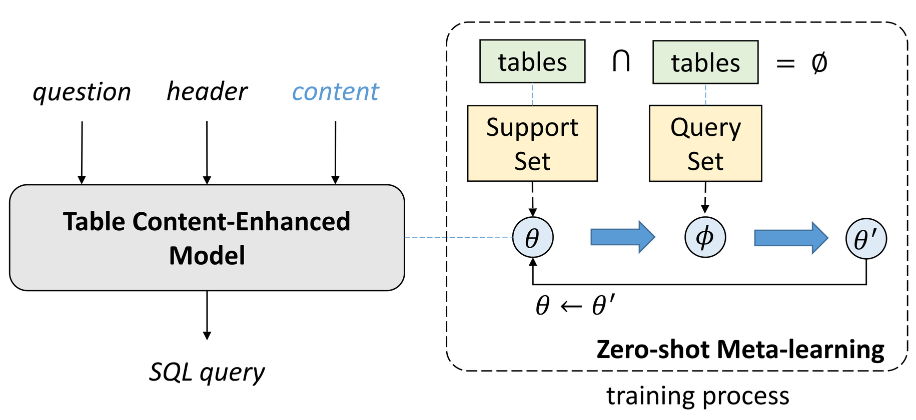

Figure 2: Overall framework of our approach.

Table Content Enhanced Model

To overcome this challenge, we adopt coarse-grained filtering before embedding. Specifically, for each header h, only the cell c with the highest literal similarity to question q will be retained. The literal similarity is computed by

$$\varphi(c; q) = \max_{\mathrm{n}(q)} \frac{\mathrm{lcs}(\mathrm{n}(q), c)}{2|\mathrm{n}(q)|} + \frac{\mathrm{lcs}(\mathrm{n}(q), c)}{2|c|} \tag{1}$$

The overall architecture of our table content-enhanced model is shown in Figure 3. It consists of an encoding module and six sub-modules corresponding to six sub-tasks. Intuitively, table content is mainly helpful to the sub-tasks for the WHERE clause, especially WN, WC, and WV. Therefore, we will detail these three sub-modules that utilize table content.

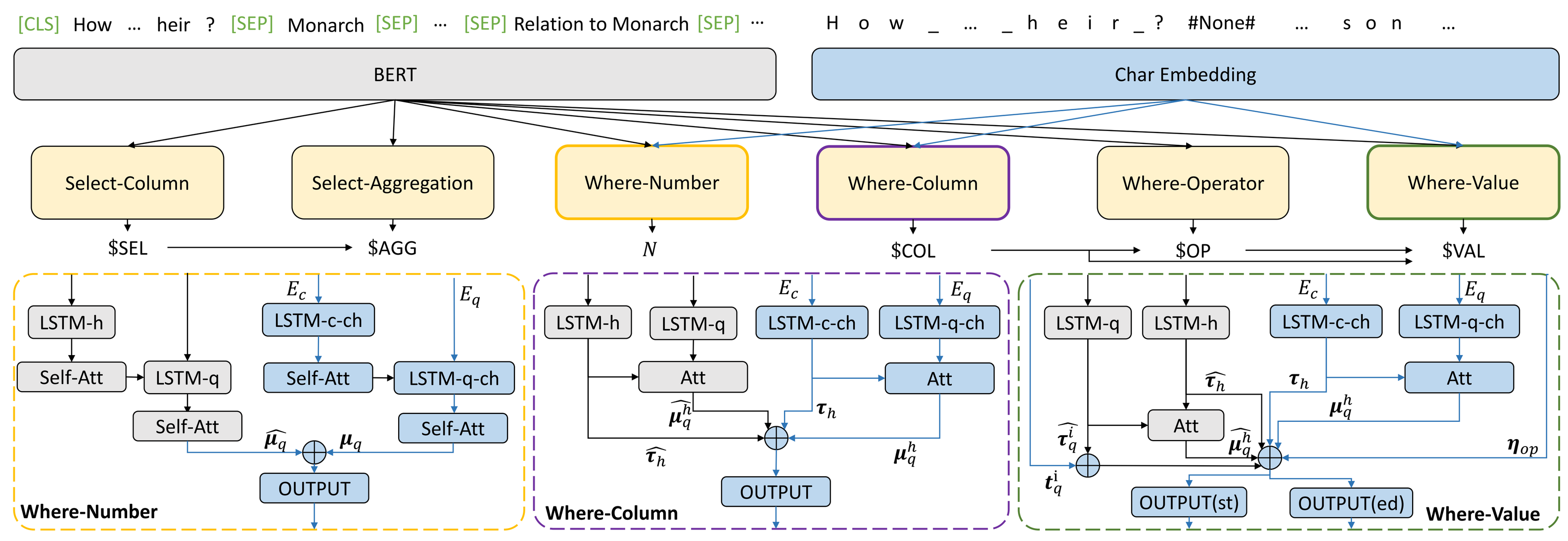

Figure 3: Architecture of the table content-enhanced model. WN, WC, and WV are detailed in the orange, purple, and green dotted box, respectively. Blue indicates the processes for table content and gray indicates the processes for headers.

Zero-Shot Meta Learning Framework

To simulate the scenario of zero-shot tables, the table set of the support set is disjoint with that of the query set. According to the split, the model experienced a two-stage gradient update during the training of each task. In the first stage, temporary parameters ϕ are obtained by calculating the loss of the support set $\mathcal{S}$ and perform the gradient updating on original parameters θ. In the second stage, the loss of the query set $\mathcal{Q}$ is first calculated with ϕ. Then, the losses of the support set and query set are jointed to calculate the gradient. Finally, original parameters θ are updated by the gradient. In addition, for sampling $\mathcal{S}$ and $\mathcal{Q}$, we follow the N-way K-shot setting, i.e, each set covers N tables and there are K samples for each table.

Although meta-learning has also been utilized in [1] on text-to-SQL, there are two key differences between our proposed approach and their method: First, [1] focuses on sampling support sets according to types of the questions (e.g., COUNT, MIN), but we sample according to different tables, so as to capture the potential relationship between questions and tables. Second, we ensure that the tables in the support set do not intersect with those in the query set to simulate a zero-shot environment. Following this setting, the model needs to learn the generic knowledge between two different sets of tables and perform a joint optimization, thus it can be forced to learn the generalization ability.

Results

The overall experimental results on WikiSQL are reported in Table 1. Except for TaBERT, where we use official API, all the other comparison results are directly taken from the original paper. On LF accuracy, our approach achieves state-of-the-art results on the development set and ranks second only to HydratNet (-0.1%) on the test set. On EX accuracy, our approach achieves state-of-the-art results on both the sets. Notably, our results are achieved by only utilizing the *base* version of BERT. After ignoring the baselines that use larger pre-trained models ("(*)" in Table 1), our approach achieves significant improvements on both LF (4.3%) and EX (4.2%) accuracy when testing. In addition, compared with the table-specific pre-trained model, our model still has advantages without pre-training on table corpus.

Approach	Dev LF	Dev EX	Test LF	Test EX
Seq2SQL	49.5	60.8	48.3	59.4
Coarse2Fine	72.5	79.0	71.7	78.5
Auxiliary Mapping	76.0	82.3	75.0	81.7
SQLova (-)	80.3	85.8	79.4	85.2
SQLova (*)	81.6	87.2	80.7	86.2
X-SQL (*)	83.8	89.5	83.3	88.7
HydratNet (*)	83.6	89.1	**83.8**	89.2
TaBERT-k1 (-)	83.1	88.9	83.1	88.4
TaBERT-k3 (-)	84.0	89.6	83.7	89.1
MC-SQL (-)	**84.1**	**89.7**	83.7	**89.4**

Table 1: Average candidate number(N_c), precision (P), recall (R) and F1-scores (F1) for different candidate query generation methods.

Conclusions

In this paper, we propose a new single-table text-to-SQL approach MC-SQL, which focuses on handling zero-shot tables. On the one hand, our approach takes advantage of table content to enhance the model. On the other hand, our approach learns the generalization capability from different tables by meta-learning.

References

[1] Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. Natural language to structured query generation via meta-learning. In *NAACL-HLT, 2018*, pages 732–738. Association for Computational Linguistics, 2018.