智慧无锡 2022
JSAI滨纷世界

Single-Table Text-to-SQL Technology for Few-Shot Scenarios

Yongrui Chen

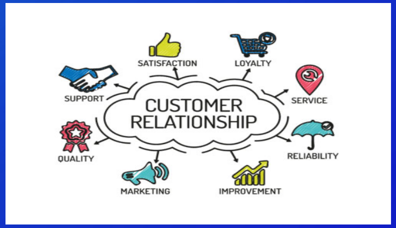
School of Computer Science and Engineering,
Southeast University, Nanjing, China

Outline

I. Background of Text-to-SQL

Background

Relational databases store a vast amount of today's information and provide the foundation of applications.

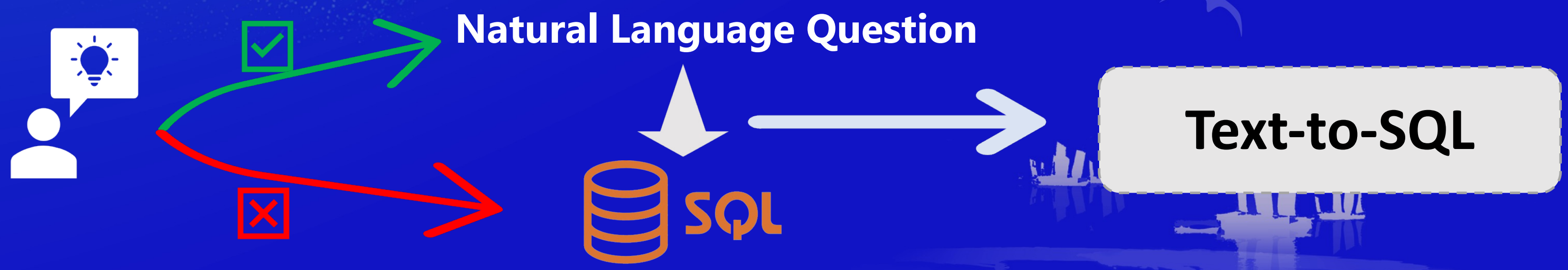


customer relations management

financial markets

medical records

- Structured Query Language (SQL)
  - Accessing relational databases
  - Machine understandable, Quick and efficient
  - Not user-friendly, requires deep understanding of the database and SQL syntax



Background

Single-Table Text-to-SQL

Task definition

$$y = M(q, T)$$

$$T = \{h_1, h_2, \ldots, h_l\}$$

Goal: learn model M

- q: NLQ
- y: SQL query
- T: table
- h_i: i-th header

WikiSQL

Table: CFLDraft

Pick #	CFL Team	Player	Position	College
27	Hamilton Tiger-Cats	Connor Healy	DB	Wilfrid Laurier
28	Calgary Stampeders	Anthony Forgone	OL	York
29	Ottawa Renegades	L.P. Ladouceur	DT	California
30	Toronto Argonauts	Frank Hoffman	DL	York
…	…	…	…	…

Question:

How many CFL teams are from York College?

SQL:

```
SELECT COUNT CFL Team FROM
CFLDraft WHERE College = "York"
```

Result:

2

Background

WikiSQL SQL skeleton

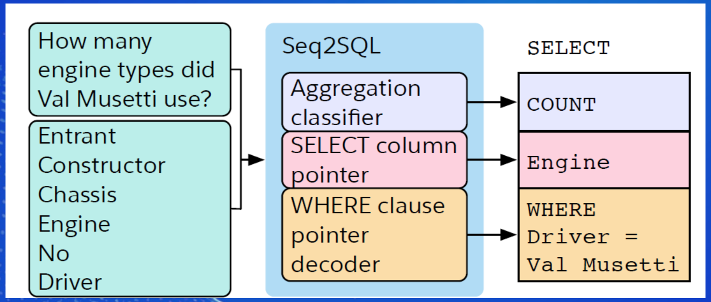
SELECT $AGG $SEL
(WHERE $COL $OP $VAL)
(AND $COL $OP $VAL)*

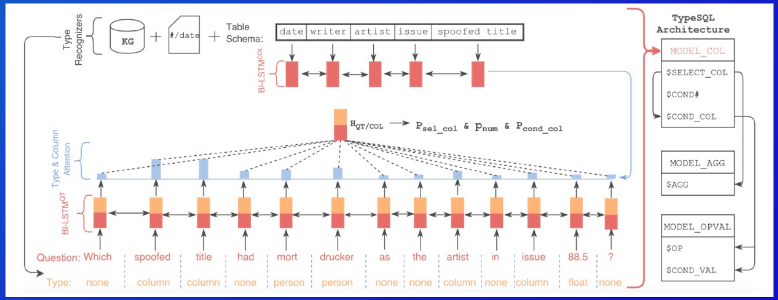
- $: slots
- ∗: zero or more AND clauses

1. **Select-Column(SC):**
   find the column $SEL in the SELECT clause from T.
2. **Select-Aggregation(SA)**:
   find the aggregation function $AGG (∈{NONE, MAX, MIN, COUNT, SUM, AVG}) of the column in the SELECT clause.
3. **Where-Number(WN):**
   find the number of where conditions, denoted by N.
4. **Where-Column(WC)**:
   find the column (header) $COL of each WHERE condition from T .
5. **Where-Operator(WO)**:
   find the operator $OP (∈ {=; >; < }) of each $COL in the WHERE clause.
6. **Where-Value(WV)**:
   find the value $VAL for each condition from the question, specifically, locating the starting position of the value in q.

Background

Seq2Seq-based Methods

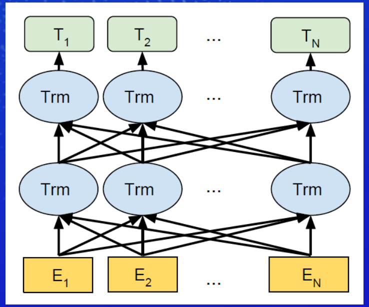




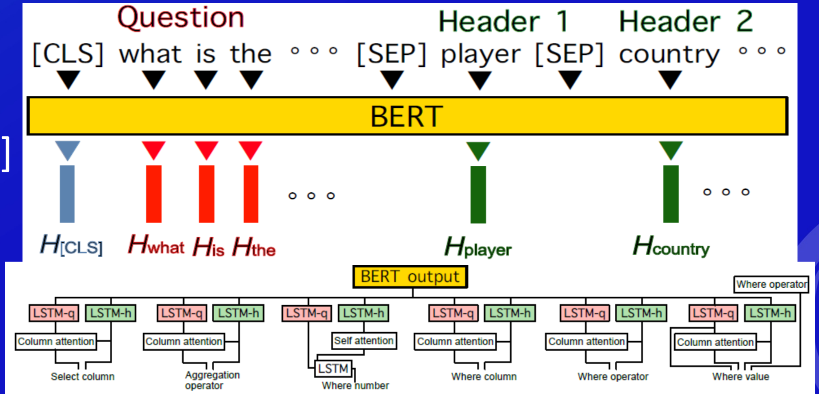
[Zhong, Xiong, and Socher 2017]; [Xu, Liu, and Song 2017]; [Yu et al. 2018]; [Dong and Lapata 2018]; [Chang et al. 2020]

- *Without Pre-trained model*
- *The SQL skeleton is not effectively used.*

Multi task learning



BERT [Devlin et al. 2019]



[Hwang et al. 2019];[He et al. 2019]; [Lyu et al. 2020]; [Chen et al. 2021] ;[Guo et al. 2022]

- *Pre-trained model Enhancement*
- *Multi-submodule framework*

智慧无锡 2022 JSAI滨纷世界

Background

customer relations management

- New product categories
- Target customer change

⋮

financial markets

- Business expansion
- New Stock Issuance
- Policy Adjustment

⋮

medical records

- New Diseases (Covid 19)
- Section upgrade

⋮

smart speakers

- New actions, skills, …
- Import knowledge

⋮

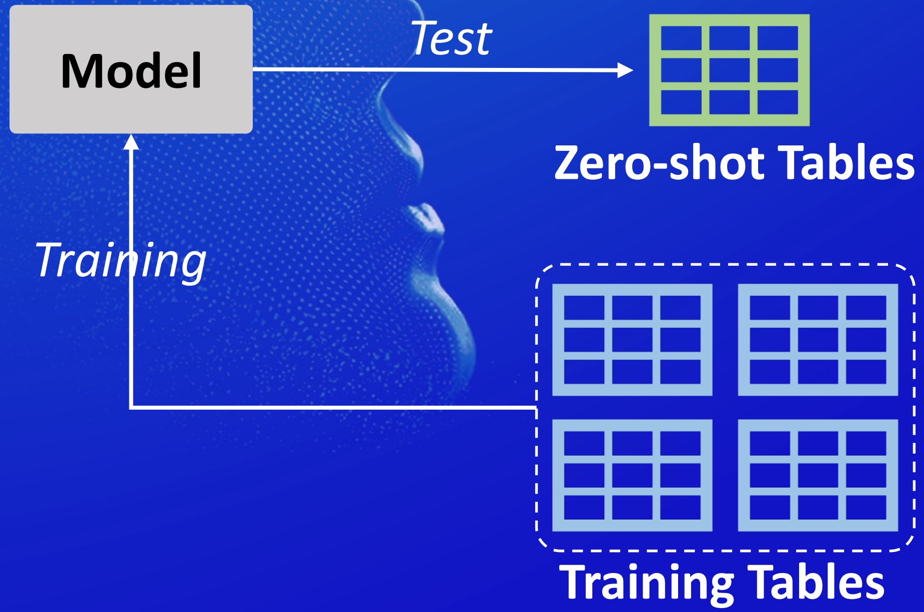
NLQ-SQL pairs that are difficult to collect annotations in a short period of time! → ***Few-shot Tables/ Zero-shot Tables***

II. Zero-shot Text-to-SQL

Zero-shot Text-to-SQL

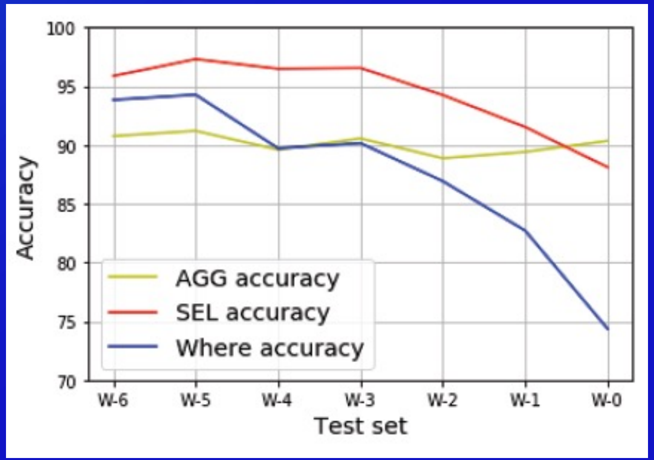
Challenge: *Zero-shot Tables*

The tables whose schema are not visible in the training set.

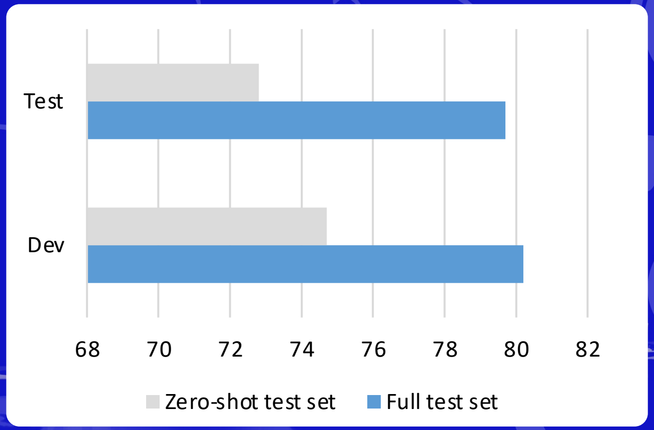


Few-shot/ Zero-shot performance

[Dong and Lapata 2018]

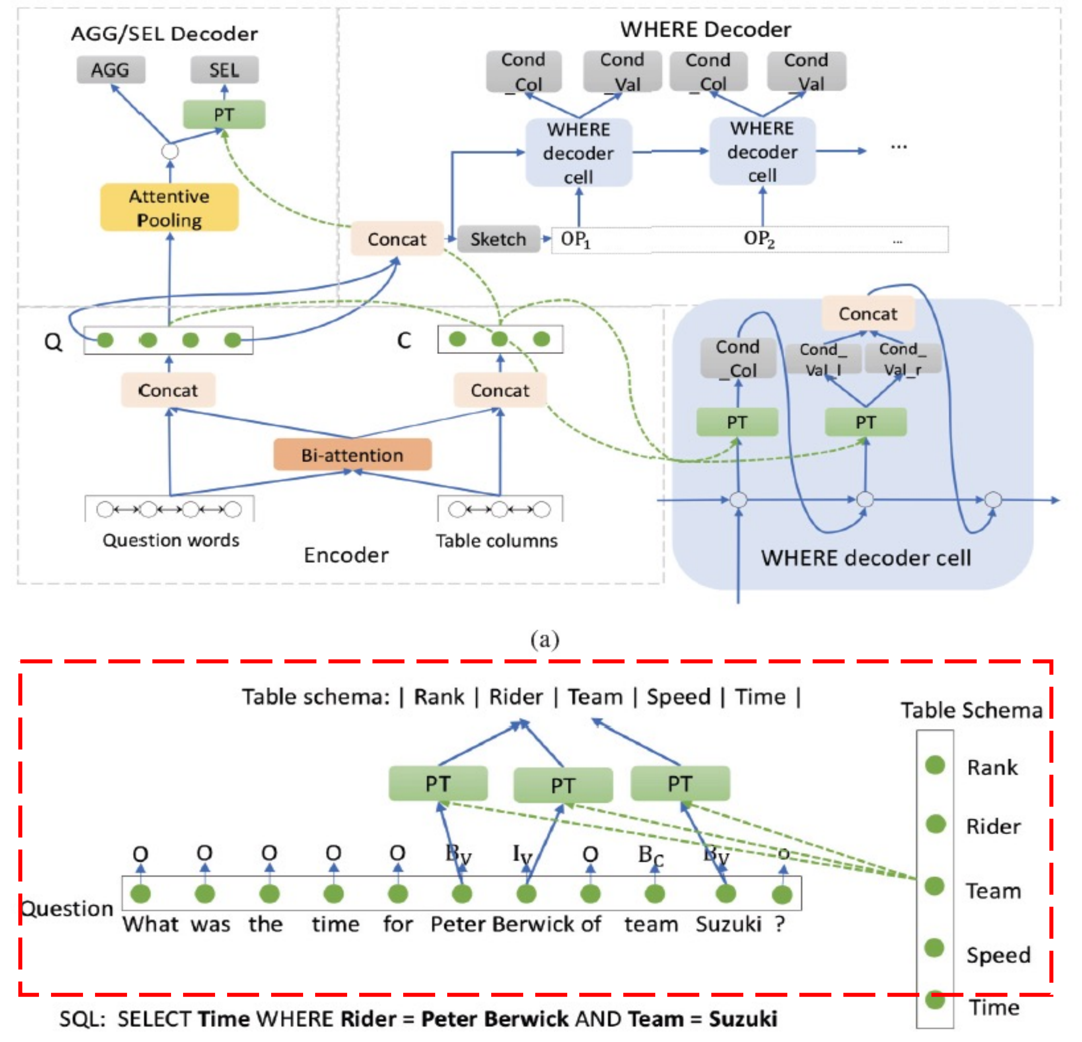


[Hwang et al. 2018]



Zero-shot Text-to-SQL

智慧无锡 2022
JSAI滨约世界



[Chang et al., 2020]

An auxiliary task to model the mapping from the NLQ to headers

Annotations need to be built manually, which is expensive.

Chang, S.; Liu, P.; Tang, Y.; Huang, J.; He, X.; and Zhou, B. 2020. Zero-Shot Text-to-SQL Learning with Auxiliary Task. In AAAI 2020

Zero-shot Text-to-SQL

Motivation

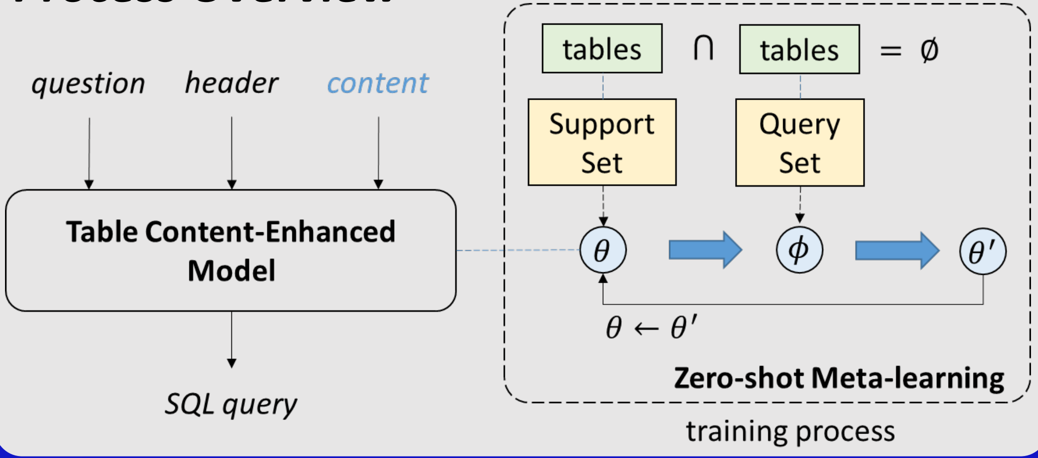
1. **Table content** can provide abundant information for predicting headers.
2. **Meta-learning** can help the model learn the generalization ability between different tables from the training data.

How many reasons did the son and heir Kale Kyetaungnyo has when he ceased to be heir?

SELECT COUNT(Ceased to be heir; reason)
WHERE Relationship to Monarch = "*son*"
AND Heir = "*Kale Kyetaungnyo* "

Monarch	*Heir*	*Relationship to Monarch*	*Ceased to be heir; reason*	...
...	...	...	...	...
Minkhang I	*Kale Kyetaungnyo*	*son*	*May 1400 father succeeded*	...
...	...	...	...	...

Process Overview



Preprocess

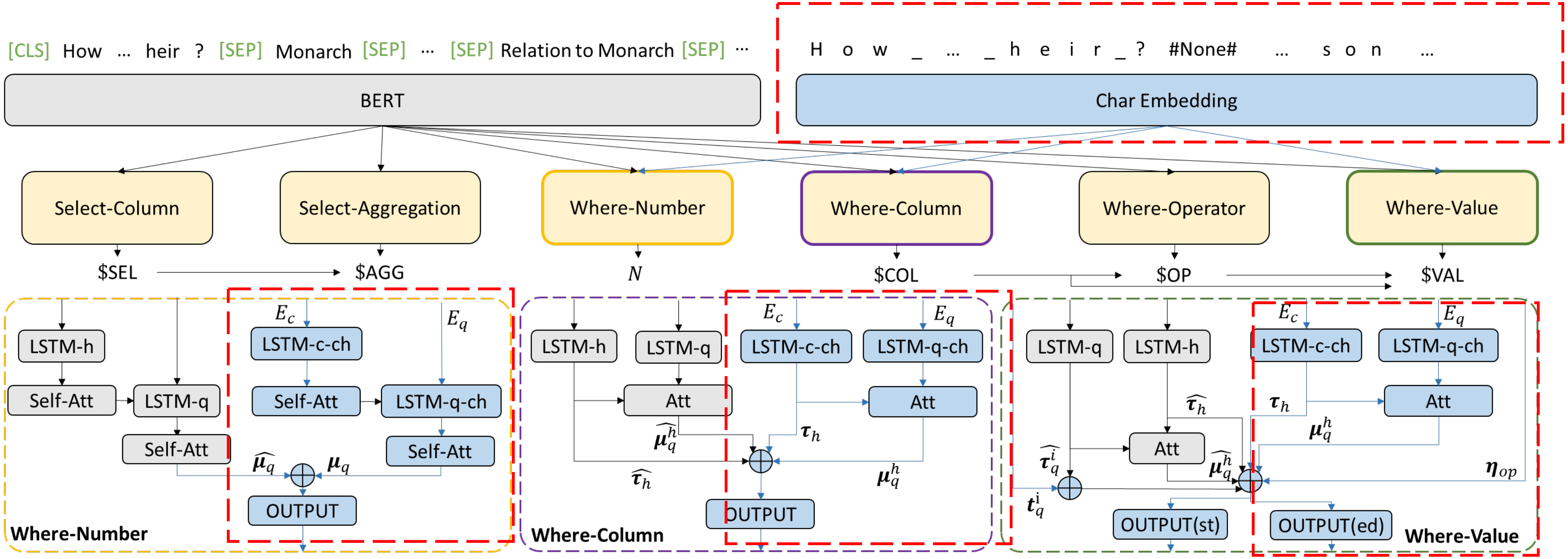
coarse-grained filtering

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

- $n(q)$: n-gram of q
- $|x|$: the length of string x
- $lcs(zx, y)$ Longest Consecutive Common Subsequence

Zero-shot Text-to-SQL

Content-Enhanced Model



Encoding Module

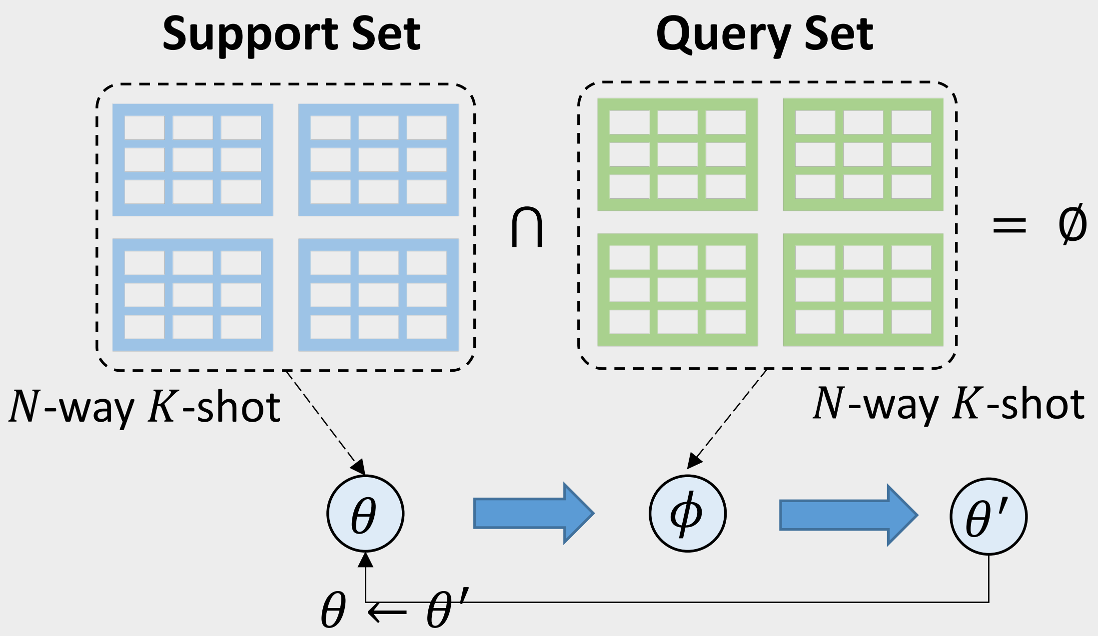
- BERT [Devlin et al. 2019] for *header*
- Char-embedding for *content*

Where-Number, Where-Column, Where-Value Sub-Module

- LSTM (gray) for obtaining information of header.
- LSTM-c (blue) for obtaining information of content.

Zero-shot Text-to-SQL

Zero-Shot Meta Learning Framework



- Simulate a zero-shot environment
- Get a possible gradient on the Support set and correct the gradient on the Query set

Algorithm 1 Zero-Shot Meta-Learning Framework

Require: A set of training samples $\mathcal{D} = \{(q^i, \mathcal{T}^i, y^i)\}$, where q^i is the i-th input question, t^i is the table which q^i relies on, and y^i is the gold SQL query of q^i. A model $\mathcal{M}(q, \theta)$, where θ is its parameters. Hyperparameters α, β and γ

```
while not done do
    for all task do
        Sample a support set S = {(q^j, T^j, y^j)} ⊆ D
        Evaluate ∇_θ L_S = ∇_θ Σ_j L(M(q^j, T^j, θ), y^j)
        Update parameters with gradient descent:
        φ = θ − α∇_θ L_S
        Sample a query set Q = {(q^k, T^k, y^k)} ⊆ D,
        where {T^j} ∩ {T^k} = ∅
        Evaluate ∇_θ L_{S←Q} = ∇_θ Σ_k L(M(q^k, T^k, φ), y^k)
        Update θ to minimum L using Adam optimizer
        with learning rate β,
        where L = γL_S + (1 − γ)L_{S←Q}
    end for
end while
```

Zero-shot Text-to-SQL

智慧无锡 2022 JSAI滨约世界

Experiments

Datasets

- *WikiSQL* [Zhong, Xiong, and Socher 2017]
  - English open-domain
  - 20K tables
  - 56,355 train; 8,421 dev; 15,878 test questions
- ESQL
  - Chinese domain-specific
  - 17 tables
  - 10,000 train; 1,000 dev; 2,000 test questions

- 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.

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: Overall results on WikiSQL. "x(-)" denotes the model x with BERT-base. "x(*)" denotes the model x with BERT-large or larger pre-trained model, such as MTDNN in X-SQL or tabular-specified TaBERT.

Compared with the table-specific pre-trained model (TaBERT), our model still has advantages **without pre-training on table corpus**.

Zero-shot Text-to-SQL

Ablation Test

- **w/o table content(TC)** Remove all the processes in WN, WC, and WV.
- **w/o value linking(VL)** Retain the processes related to TC but remove the value linking in WV
- **w/o meta-learning(ML)** Replace the meta-learning strategy with the traditional mini-batch strategy.

Dataset	Model	SC	SA	WN	WC	WO	WV	LF
WikiSQL	SQLova	96.7 / 96.3	90.1 / 90.3	98.4 / 98.2	94.1 / 93.6	97.1 / 96.8	94.8 / 94.3	80.2 / 79.7
	TaBERT-k1	97.2 / **97.1**	90.5 / 90.6	98.9 / 98.8	96.1 / 96.1	**97.9 / 97.8**	**96.7** / 96.6	83.1 / 83.1
	TaBERT-k3	**97.3 / 97.1**	**91.1 / 91.2**	98.8 / 98.7	96.6 / 96.4	97.5 / 97.5	96.6 / 96.2	83.9 / **83.7**
	MC-SQL	96.9 / 96.4	90.5 / 90.6	**99.1 / 98.8**	97.9 / **97.8**	97.5 / **97.8**	**96.7 / 96.9**	**84.1 / 83.7**
	w/o TC	97.0 / 96.5	89.8 / 90.0	98.6 / 98.3	94.5 / 93.7	97.2 / 97.0	94.7 / 94.7	79.9 / 79.2
	w/o VL	97.0 / 96.7	90.4 / **90.8**	99.0 / 98.7	**98.0** / 97.6	97.5 / 97.2	95.6 / 95.5	82.9 / 83.0
	w/o ML	96.5 / 96.2	90.4 / 90.4	98.9 / 98.7	97.8 / 97.4	97.5 / 97.4	96.5 / 96.1	83.2 / 82.9
ESQL	SQLova	96.2 / 95.9	98.9 / 99.0	98.5 / 98.4	84.6 / 84.1	96.5 / 95.8	89.9 / 89.6	72.0 / 71.5
	MC-SQL	**97.2 / 97.3**	99.1 / **99.2**	**98.9 / 98.9**	**93.6** / 93.3	**97.5** / 96.8	**92.9 / 92.6**	**82.8 / 82.7**
	w/o TC	95.9 / 96.1	99.2 / 99.1	98.8 / 98.3	84.5 / 84.4	96.7 / 96.2	90.5 / 90.3	72.9 / 72.1
	w/o VL	96.5 / 96.7	**99.3** / 98.9	**98.9** / 98.8	93.5 / **93.5**	97.4 / **96.9**	92.0 / 91.8	82.1 / 81.9
	w/o ML	96.2 / 96.0	98.8 / 98.9	**98.9** / 98.8	92.4 / 92.7	**97.5** / 96.7	92.7 / 92.3	82.3 / 81.9

Table 2: Results of sub-tasks on WikiSQL and ESQL.

- Total MC-SQL achieves the **optimal** results on **LF accuracy** and **most** sub-tasks.
- the contribution of TC is mainly reflected in the three sub-tasks of WN, WC, and WV.
- Meta-learning is helpful for **all** sub-tasks and has the most significant improvements on SC and SA.

Zero-shot Text-to-SQL

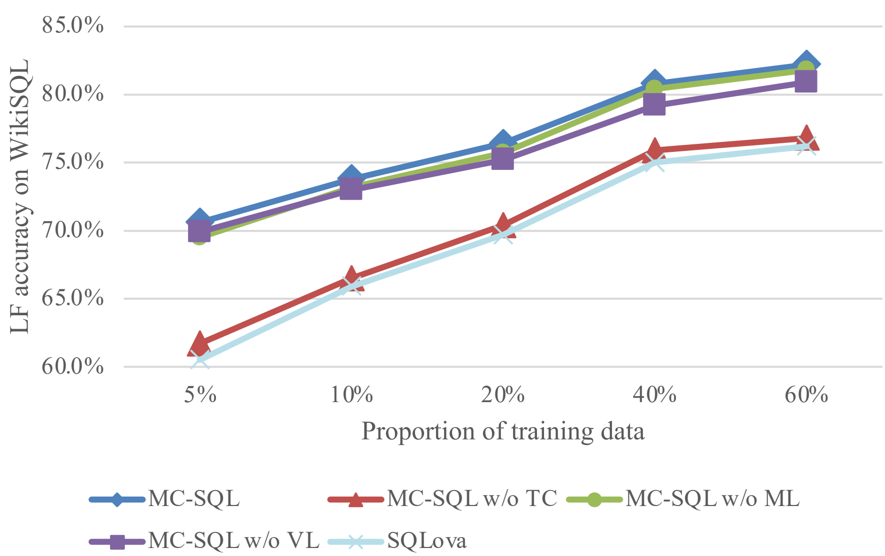
Zero-shot Test

Dataset	Model	SC	SA	WN	WC	WO	WV	LF
WikiSQL (zero-shot)	SQLova	95.8 / 95.2	89.7 / 89.3	97.6 / 97.4	91.1 / 90.4	95.9 / 95.7	90.1 / 90.5	74.7 / 72.8
	TaBERT-k1	96.6 / **96.4**	91.0 / 91.0	98.6 / **98.4**	94.8 / 94.6	**97.7 / 97.5**	**95.3 / 94.6**	81.3 / 80.5
	TaBERT-k3	**96.7 / 96.4**	**91.6 / 91.5**	98.2 / 98.2	95.1 / 95.0	96.8 / 97.0	94.9 / 94.2	82.0 / **81.2**
	MC-SQL	**96.4** / 95.5	91.1 / 91.0	**98.7** / 98.1	96.6 / **96.3**	97.1 / 96.7	94.8 / 94.2	**82.4** / 80.5
	w/o TC	96.2 / **95.7**	91.0 / 90.5	97.6 / 97.7	91.5 / 90.7	96.2 / 96.1	90.5 / 90.8	75.8 / 73.6
	w/o VL	96.2 / 95.8	90.6 / 90.9	**98.7** / 98.0	**97.1 / 96.3**	97.1 / 96.3	91.7 / 92.1	79.0 / 79.1
	w/o ML	95.7 / 95.0	90.4 / 90.2	98.5 / 98.2	96.0 / 95.8	96.8 / 96.7	94.0 / 93.5	81.2 / 79.4
ESQL (zero-shot)	SQLova	94.3 / 94.0	97.8 / 97.9	97.3 / 97.0	80.5 / 80.7	95.9 / 94.6	87.8 / 86.7	62.9 / 61.2
	MC-SQL	**94.6 / 94.2**	98.0 / 98.0	**97.5 / 97.3**	**93.7 / 92.0**	**96.2 / 94.8**	**91.9 / 90.5**	**76.7 / 74.8**
	w/o TC	94.4 / 94.1	**98.1 / 98.2**	97.1 / 97.2	80.7 / 80.6	95.5 / 94.2	88.4 / 87.6	64.7 / 63.3
	w/o VL	93.8 / 94.0	98.0 / 98.1	97.4 / 97.2	92.6 / 91.1	95.1 / **94.8**	90.9 / 90.1	75.7 / 73.7
	w/o ML	93.5 / 93.0	97.7 / 97.9	97.4 / 96.9	93.2 / 91.8	96.0 / 94.3	91.2 / 90.2	75.2 / 72.9

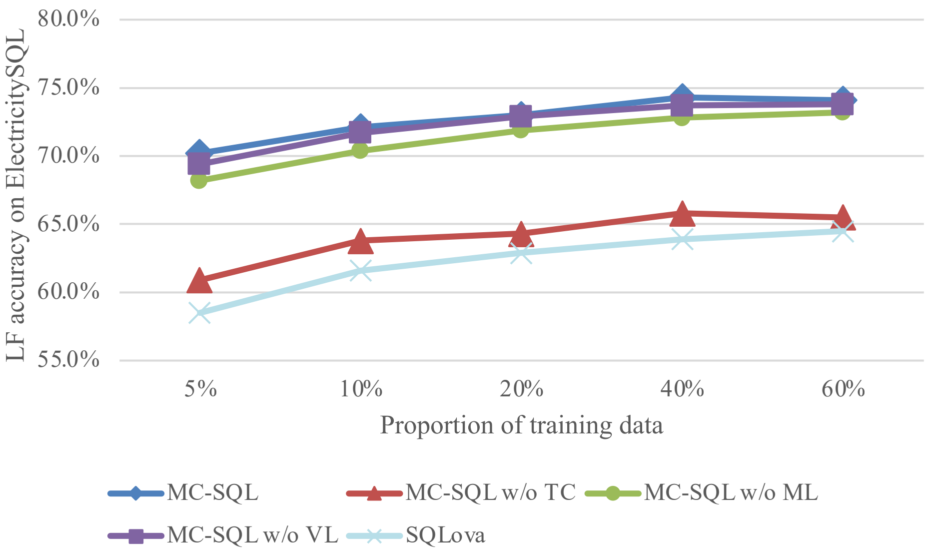
Table 3: Results of zero-shot subset on WikiSQL and ESQL.

- MC-SQL achieves **greater** improvements over SQLova on the zero-shot subsets of both WikiSQL (7.7% vs 4.0%) and ESQL (13.6% vs 10.2%).
- The contribution of table content is **greater** on zero-shot tables
- Meta-learning is also contributing to the WHERE clause when handling zero-shot tables.

Zero-shot Text-to-SQL



LF on WikiSQL with proportions of training data.



LF on ESQL with proportions of training data.

- The MC-SQL equipped with all components always maintains **optimal** performance with different sizes of training data.
- When the training data is **small**, the improvement achieved by MC-SQL over SQLova is **more significant**, especially on WikiSQL.
- The less training data, the **more significant** the improvement brought by meta-learning.

III. Few-shot Text-to-SQL

Few-shot Text-to-SQL

智慧无锡2022
JSAI滨约世界

Motivation

Readily available from previous user records

- Unannotated quetions provides the information to the few-shot tables.
- To learn the generic knowledge, the optimization object should focus on the common columns.

Table 1

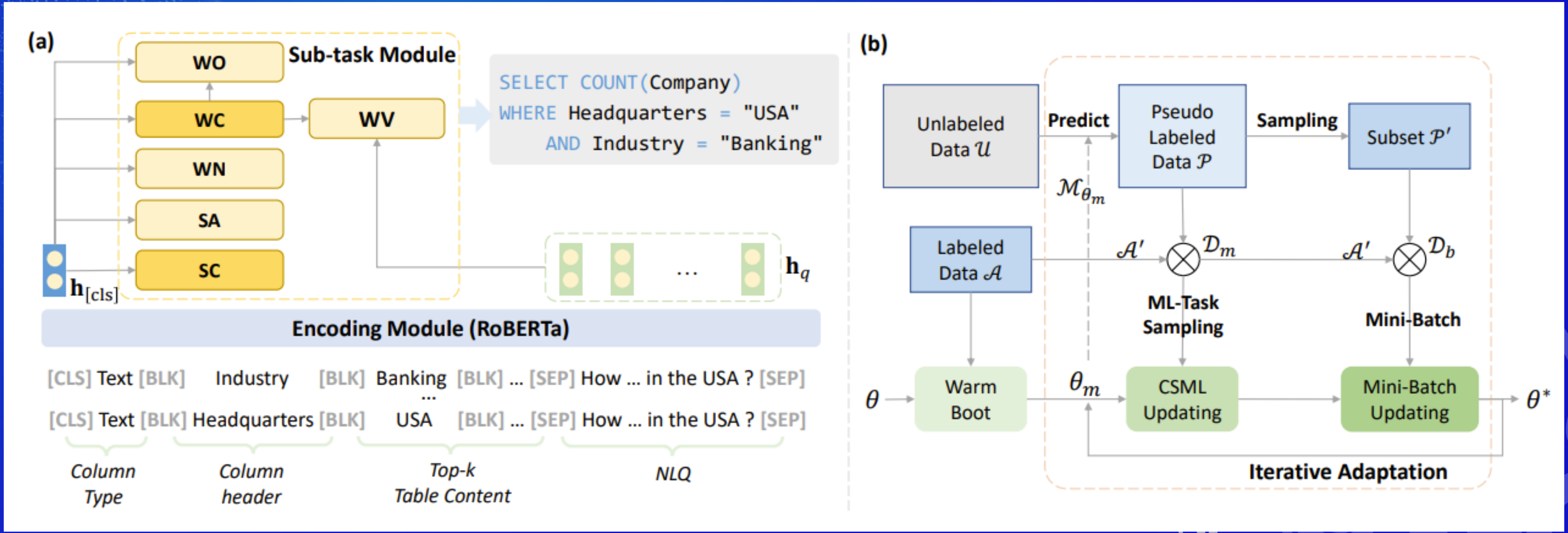
Company	*Headquarters*	*Industry*	…	*Profits*	*Market Value*
Citigroup	USA	Banking	…	21.54	247.42
…	…	…	…	…	…

Table 2

Title	*Author*	*Company*	…	*Format*	*Release Date*
Doctor Who and the Cave Monsters	Malcolm Hulke	BBC	…	4-CD	2007-09-03
…	…	…	…	…	…

Few-shot Text-to-SQL

Overview: MST-SQL

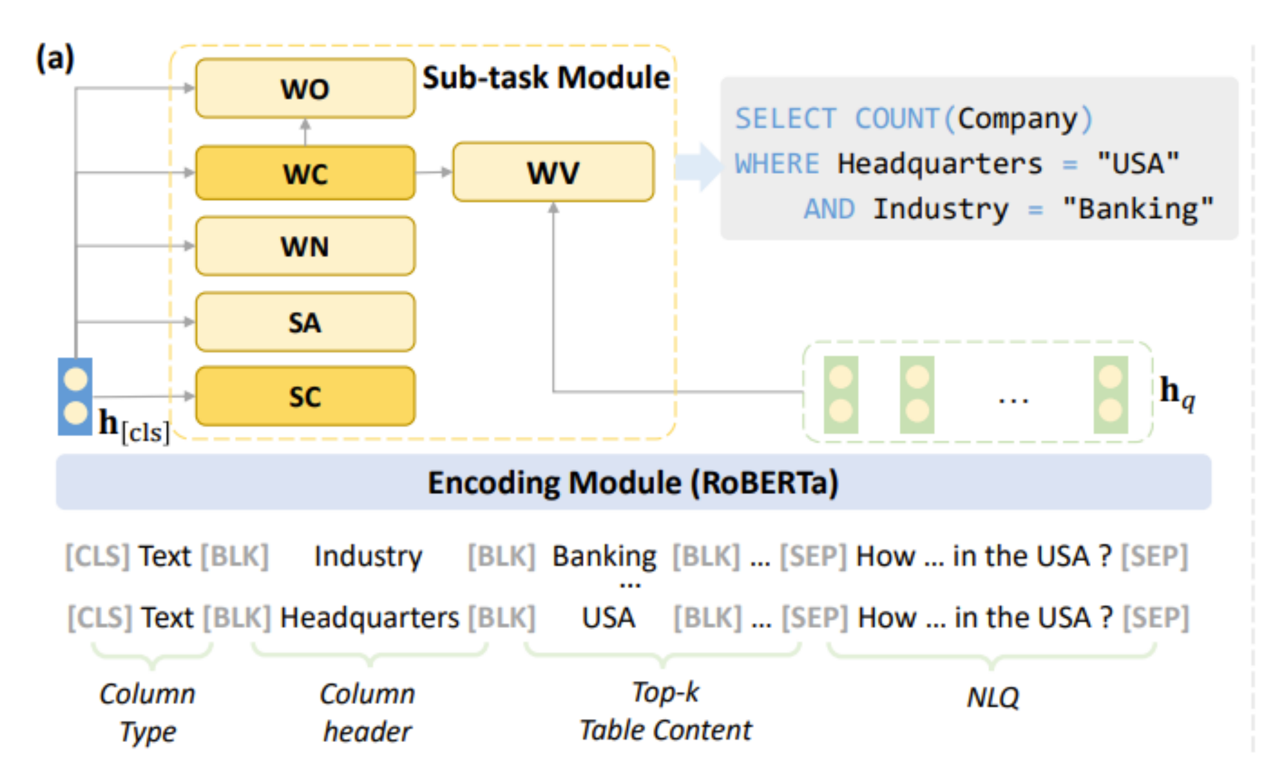


The architecture of our basic model. The procedure of our meta self-training framework.

Few-shot Text-to-SQL

Basic Model

We adopt an encoder-subtask architecture to generate SQL, which refers to HydraNet [Lyu et al., 2020].



Encoder Module

- An original input (q, T) is decomposed into m column-input (q, h^i).
- Each column-input includes column type, column header, filtered content, and NLQ.

Sub-task Module

- SQL generation is divided into six sub-tasks:
  - SC: Predicting the column in the SELECT clause.
  - SA: Predicting the aggregation function in the SELECT clause.
  - WN: Predicting the number of conditions in the WHERE clause.
  - WC: Predicting the column of the condition in the WHERE clause.
  - WO: Predicting the operator of the condition in the WHERE clause.
  - WV: Extract the value of the condition in the WHERE clause.

Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, Zheng Chen: Hybrid Ranking Network for Text-to-SQL. CoRR abs/2008.04759 (2020)

Few-shot Text-to-SQL

智慧无锡 2022 JSAI滨纷世界

Meta Self-Training

Labeled Data: $\mathcal{A} = \{a^1, a^2, \ldots a^{|\mathcal{A}|}\}$, where $a^i = (q^i, \mathcal{T}^i, y^i)$

Unlabeled Data: $\mathcal{U} = \{u^1, u^2, \ldots, u^{|\mathcal{U}|}\}$, where $u^i = (q^i, \mathcal{T}^i)$

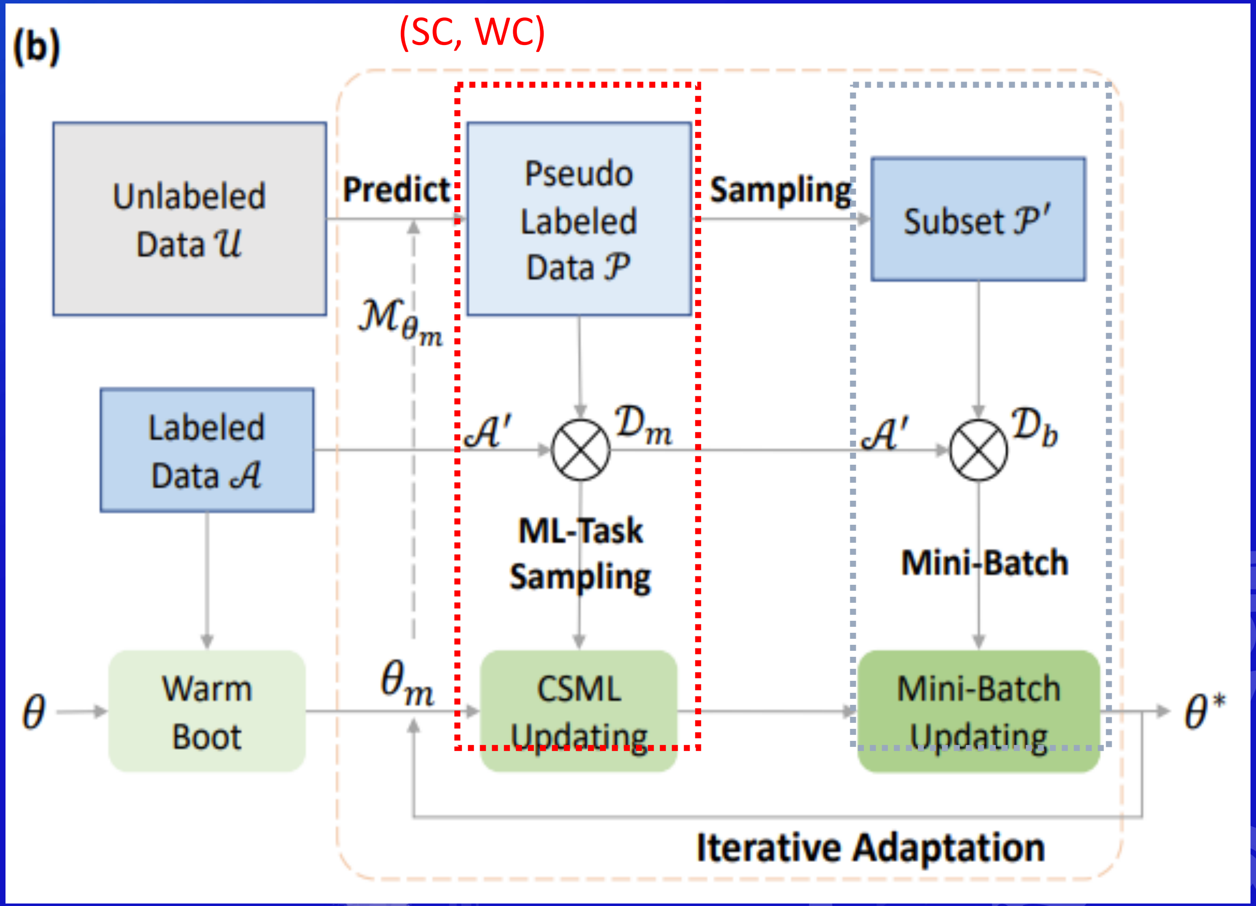
Iterative Adaption

a. We propose a Column Specificity Meta-Learning (CSML) algorithm to improve the Column Selection Task(SC, WC) since they are typically table-sensitive tasks.

b. The parameters are updated by optimizing the total loss of all the sub-tasks with the mini-batch training to further adopt semi-supervised learning.

Meta-Learning for table-sensitive tasks (SC, WC)

Mini-Batch training for all the sub-tasks



Few-shot Text-to-SQL

Column Specificity Meta-Learning(CSML)

New objects:

Each original sample $(q, \mathcal{T}, y)$ is broken into column-samples (q, h^i, y_{sc}, y_{wc}) and all of them are shuffled and sampled to form n ML-tasks. The loss of each column-sample is defined as

$$\mathcal{L}^i = H(P_{sc}(h^i|q), y_{sc}) + H(P_{wc}(h^i|q), y_{wc})$$

SC and WC are logically transformed into classification tasks:

- h^i is irrelevant to q.
- h^i is the query target of q.
- h^i is one of the query conditions of q.

Column Specificity:

We define column specificity as

$$\mu^i = \frac{N_{distinct} \cdot N^i}{N_{total}}$$

Table 1

Company	*Headquarters*	*Industry*	…	*Profits*	*Market Value*
Citigroup	USA	Banking	…	21.54	247.42
…	…	…	…	…	…

Table 2

Title	*Author*	*Company*	…	*Format*	*Release Date*
Doctor Who and the Cave Monsters	Malcolm Hulke	BBC	…	4-CD	2007-09-03
…	…	…	…	…	…

Few-shot Text-to-SQL

Results in Standard Text-to-SQL Setting

Method	Dev.LF	Dev.EX	Test.LF	Test.EX
SQLOVA	81.6	87.2	80.7	86.2
X-SQL	83.8	89.5	83.3	88.7
HydraNet	83.6	89.1	83.8	89.2
MC-SQL*	84.1	89.7	83.7	89.4
IE-SQL+	84.6	88.7	84.6	88.8
SeaD	84.9	90.2	84.7	90.1
SDSQL+	86.0	91.8	85.6	91.4
BRIDGE*	86.2	91.7	85.7	91.1
TAPAS*+	85.1	-	83.6	-
TABERT*+	84.0	89.6	83.7	89.1
GRAPPA*+	85.9	-	84.7	-
MST-SQL*+	85.7	90.8	85.4	90.3
Basic*	85.6	91.2	85.0	90.8
ST-only*+	**86.4**	**91.9**	**85.8**	**91.6**

Strong End-to-End Model

Tabular Pre-trained Model

Results on original WikiSQL."*" denotes using table contents or accessing databases during SQL generation. "+" denotes using extra knowledge (e.g., tabular pre-training).

- The ST-only setting surprisingly achieves state-of-the-art results. It proves that self-training with unlabeled data can also improve the model in rich-resource scenarios.
- A possible reason for the drop brought by CSML is that mini-batch can optimize parameters more stably than meta-learning with sufficient training data.

Few-shot Text-to-SQL

Results in Few-Shot Tests

Strong End-to-End Model

Tabular Pre-trained Model

	WikiSQL				ESQL			
Method	1-SHOT	2-SHOT	3-SHOT	4-SHOT	5-SHOT	10-SHOT	15-SHOT	20-SHOT
SQLova	23.3	40.8	48.7	55.3	22.3	39.6	52.0	54.7
MC-SQL	52.0	62.9	71.0	73.8	36.5	53.2	60.5	67.4
HydraNet	64.2	69.9	72.9	74.3	43.6	58.1	70.5	76.7
BRIDGE	53.6	68.9	73.1	77.3	-	-	-	-
TaBERT	57.5	67.4	71.2	72.5	-	-	-	-
GraPPa	72.8	76.8	78.0	78.1	-	-	-	-
MST-SQL	**78.4**	**80.5**	**82.1**	**83.2**	**55.3**	**67.4**	**76.7**	**80.5**
Basic	69.6	74.3	76.3	77.3	45.3	60.2	72.2	77.7
ST-only	75.8	78.6	79.4	81.1	51.2	64.5	74.2	80.2

For each dataset, we built four training sets by the shot number (WikiSQL: {1, 2, 3, 4}, ESQL:{5, 10, 15, 20}), and construct validation and test sets with the same setting(4 / 100 samples each table for WikiSQL / ESQL).

Few-shot Text-to-SQL

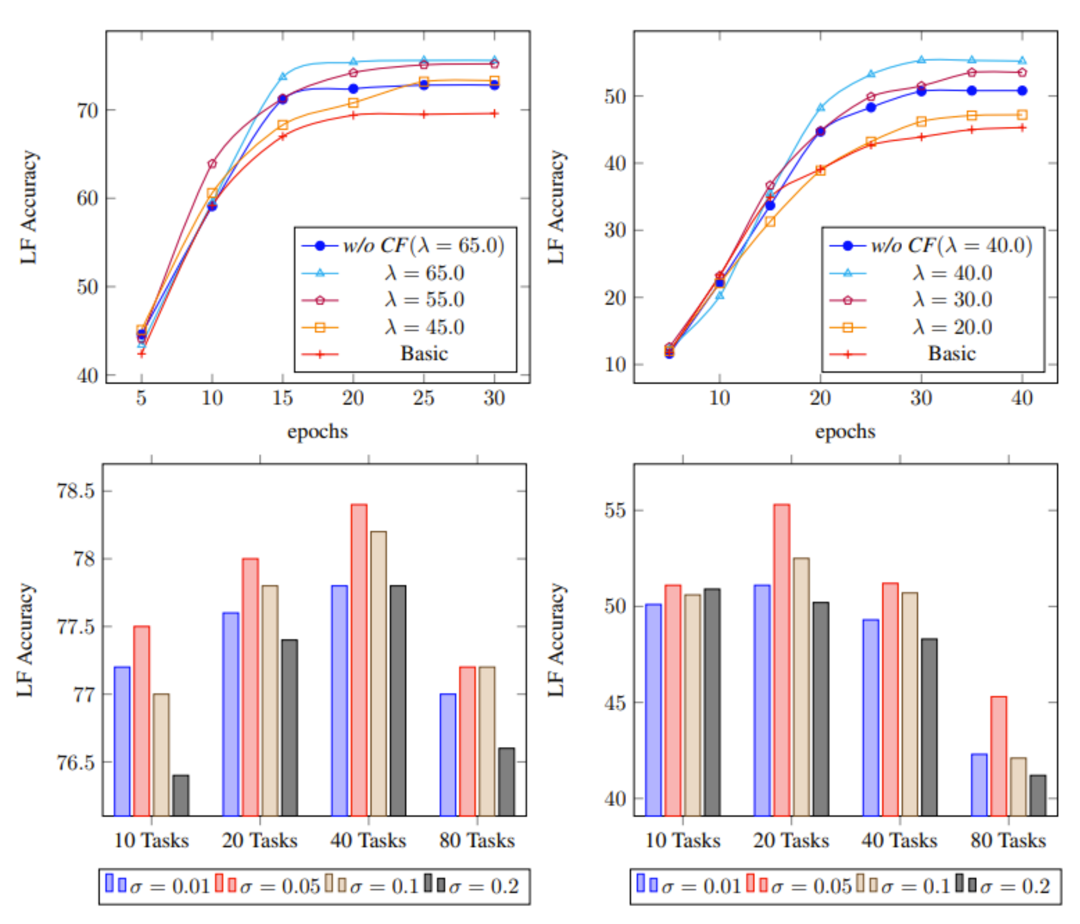
The Impact of Unlabeled Data Source

Method	1-SHOT	2-SHOT	3-SHOT	4-SHOT
Basic	69.6	74.3	76.3	77.3
ST-only$_{wtq}$	72.2	75.5	76.8	77.9
ST-only$_{wiki}$	75.8	78.6	79.4	81.1
MST-SQL$_{wtq}$	72.9	77.1	77.4	78.2
MST-SQL$_{wiki}$	**78.4**	**80.5**	**82.1**	**83.2**

Ablation Study of CSML

	Method	SC	SA	WN	WC	WO	WV	LF
WikiSQL	MST-SQL	**98.2**	86.7	95.5	**93.7**	**94.4**	**93.5**	**78.4**
	w/o OB	97.8	87.2	**95.8**	93.0	94.0	92.5	77.3
	w/o CS	97.4	**88.2**	93.4	91.6	92.9	92.8	77.2
	w/o OB & CS	96.9	87.5	94.9	92.7	93.4	92.5	76.4
	ST-only	96.4	86.1	95.7	91.5	93.0	92.4	75.8
ESQL	MST-SQL	**93.3**	82.9	**86.5**	**80.7**	**90.0**	**87.1**	**55.3**
	w/o OB	91.5	82.2	85.5	77.1	88.1	85.7	53.1
	w/o CS	91.2	80.4	86.3	76.9	88.8	86.2	52.8
	w/o OB & CS	91.5	80.5	85.1	77.3	87.8	86.5	52.4
	ST-only	91.2	**83.9**	86.1	74.2	86.5	84.8	51.2

Ablation Study of Self-Training



IV. Conclusion

Conclusion

智慧无锡 2022 JSAI滨约世界

- Table content works well for dealing with zero-shot tables and potential headers can be inferred from the semantic relevance of the questions and content.
- Meta-learning can improve the generalization ability of text-to-SQL models, which helps to handle not only few-shot tables but also zero-shot tables.
- Self-training can be used to handle few-shot text-to-SQL using unlabeled NLQs.
- The generic knowledge of common columns are more useful for text-to-SQL models to improve generalization capabilities.

智慧无锡 2022
JSAI滨纷世界

中国無錫 Wuxi China 無錫濱湖 Binhu Wuxi

Thank you for your listening !