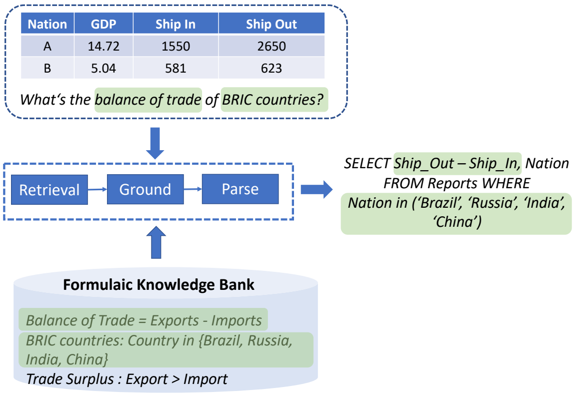
Towards Knowledge-Intensive Text-to-SQL Semantic Parsing with Formulaic Knowledge

Longxu Dou[1]*, **Yan Gao**[2], **Xuqi Liu**[1], **Mingyang Pan**[1], **Dingzirui Wang**[1],
Wanxiang Che[1], **Min-Yen Kan**[3], **Dechen Zhan**[1], **Jian-Guang Lou**[2]
[1] Harbin Institute of Technology [2] Microsoft Research Asia
[3] National University of Singapore
{lxdou, xqliu, mypan, dzrwang, car}@ir.hit.edu.cn, dechen@hit.edu.cn,
{yan.gao, jlou}@microsoft.com, kanmy@comp.nus.edu.sg

Abstract

In this paper, we study the problem of ***knowledge-intensive text-to-SQL***, in which domain knowledge is necessary to parse expert questions into SQL queries over domain-specific tables. We formalize this scenario by building a new Chinese benchmark KNOWSQL consisting of domain-specific questions covering various domains. We then address this problem by presenting ***formulaic knowledge***, rather than by annotating additional data examples. More concretely, we construct a formulaic knowledge bank as a domain knowledge base and propose a framework (REGROUP) to leverage this formulaic knowledge during parsing. Experiments using REGROUP demonstrate a significant 28.2% improvement overall on KNOWSQL.



Nation	GDP	Ship In	Ship Out
A	14.72	1550	2650
B	5.04	581	623

What's the balance of trade of BRIC countries?

SELECT Ship_Out – Ship_In, Nation FROM Reports WHERE Nation in ('Brazil', 'Russia', 'India', 'China')

Retrieval
Ground
Parse
Formulaic Knowledge Bank
Balance of Trade = Exports - Imports
BRIC countries: Country in {Brazil, Russia, India, China}
Trade Surplus : Export > Import

Figure 1: Harnessing REGROUP with formulaic knowledge for knowledge-intensive text-to-SQL with three steps: (1) **Ret**rieval the formulaic knowledge; (2) **Grou**nd the concept of formulaic knowledge; (3) **P**arse the question.

1 Introduction

Text-to-SQL translates user queries into executable SQL, greatly facilitating interactions between users and relational databases. Along with the release of large-scale benchmarks (Zhong et al., 2017; Yu et al., 2018, 2019a,b) and developments in model design (Wang et al., 2020a; Cao et al., 2021), text-to-SQL works are now achieving promising results in both research and practical applications (Zeng et al., 2020).

However, in the professional application of text-to-SQL, such as in the data analysis of financial reports, models require external knowledge to map the expert query with the domain-specific database. Take the financial query for example: *What's the EBIT*[1] *of Walmart?*, where the underlying database has component columns that can be used to calculate the EBIT. We treat this problem as ***knowledge-intensive text-to-SQL***, where domain knowledge is highly necessary to parse expert questions over domain-specific tables. This problem prevents text-to-SQL techniques from being fielded in novel, professional applications to assist the experts in processing data.

Traditional approaches would address this problem by annotating specific question/SQL pairs on a target domain (Wang et al., 2015; Herzig and Berant, 2019). Then such mappings are induced during the training process. This approach does work but has the drawback that any induced information is both *fragile* and *expertise-heavy*: such knowledge does not port across domains and requires expert knowledge to craft.

We propose to solve this problem by modeling how a non-expert person might tackle this problem. When meeting unseen examples (as in the EBIT case above), they may first search for the related mathematical formulas from public resources, then ground the concepts referenced in the formulas with schema elements presented in their particular databases. This process leverages common, encoded ***formulaic knowledge*** that are already described in publicly-available resources such as tu-

*Contribution during the internship at Microsoft Research Asia.

[1] EBIT is Earnings Before Interest and Tax, and is calculated as *Revenue – Cost of Goods Sold – Operating Expenses.*

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 5240–5253
December 7-11, 2022 ©2022 Association for Computational Linguistics

torials, textbooks, encyclopedias, and references.

Inspired by this, we propose to address the knowledge-intensive text-to-SQL through ***formulaic knowledge*** which provides the evidence of mapping from domain-specific phrases presented in questions to actual SQL operations over schema elements. More concretely, we define a taxonomy of three types of formulaic knowledge: *calculation*, *union*, and *condition*, each corresponding to a particular snippet of SQL. Then we propose REGROUP, a text-to-SQL framework (Fig. 1), consisting of three stages: (1) **Re**trieve the formulaic knowledge from formulaic knowledge bank as an external knowledge source; (2) **Grou**nd the concept of formulaic knowledge to the schema elements (*e.g.*, *Exports* to *Ship_Out*); (3) **P**arse the results with the question, schema, and grounded formulaic knowledge. The external formulaic knowledge bank imbues REGROUP with formulaic knowledge, making it *knowledgeable*. REGROUP is also *extensible* because updating the formulaic knowledge bank does not require retraining any modules.

Moreover, we construct a Chinese benchmark KNOWSQL, to examine the effectiveness of REGROUP framework. It advances the existing knowledge-intensive text-to-SQL beyond the previous work (Wang et al., 2020b; Zhao et al., 2022) by considering more SQL operations and challenging domains. Experimental results demonstrate the REGROUP with formulaic knowledge would improve the performance by 23.4% overall. Furthermore, we classify error cases into three classes, which are resolvable by advancing the corresponding module of REGROUP. Finally, we discuss the potential future work such as expanding the scope of knowledge and advancing REGROUP model design.

Our contributions are summarised as follows:

- To the best of our knowledge, we are the first to explore knowledge-intensive text-to-SQL and propose a challenging Chinese benchmark KNOWSQL, which requires domain-specific knowledge.
- We propose a novel framework REGROUP to address knowledge-intensive text-to-SQL by retrieving and grounding formulaic knowledge, which is knowledge-extensible.
- Experimental results demonstrate the effectiveness of REGROUP with formulaic knowledge which achieves 28.2% overall improvement on KNOWSQL.

2 Knowledge-Intensive Text-to-SQL

2.1 Problem Analysis

After studying the real cases in professional data analysis, we roughly categorize the required knowledge for knowledge-intensive text-to-SQL into three classes : (1) *linguistic knowledge* enables the model to adapt to linguistic diversity; (2) *domain knowledge* allows the model to perceive domain-specific sayings and concepts; (3) *mathematical knowledge* yields the specific SQL operations (*e.g.*, *Density* phrase to *division* operation). These three sets of knowledge jointly provide the evidence of *mapping from domain-specific phrases of questions to actual SQL operations over schema elements*.

However, most text-to-SQL researches focus on general scenario (Yu et al., 2018; Zhong et al., 2017), where linguistic knowledge is mainly required. Recently, Wang et al. (2020b) and Zhao et al. (2022) promote text-to-SQL to more challenging scenarios via involving the calculation questions. In this paper, we further explore the knowledge-intensive text-to-SQL by considering more operations (*e.g.*, calculation, union, and condition) with more challenging domains which require all these three classes of knowledge.

2.2 Challenge

Despite that pre-trained language models contain linguistic knowledge, they lack domain knowledge and mathematical knowledge. Therefore, the model would meet two problems: (1) ***don't know which operations to use***: if an operation (*e.g.*, *density = total number / space*) has never occurred in training data, the model rarely employ this unseen operation during the inference; (2) ***don't know how to adapt operations***: the model would fail to generalize the operation across domains. For instance, the model cannot generalize the calculation of *Population Density (number of people / land area)* to *Car Density (number of cars / parking lot area)*.

Accordingly, we consider that the vanilla pre-trained language model is (1) ***narrow*** since it only supports the limited operation and (2) ***inefficient*** since it can't generalize the operation across domains. However, it's time-consuming and expertise-heavy to directly increase the amount of annotated data examples. In contrast, we address this challenge from the view of formulaic knowledge in Sec 3, which is more knowledge-extensible.

2.3 KNOWSQL Benchmark

	#DB	#Question	#Formulaic
Train	160	$23,157$	328
Dev	40	$2,731$	122
Finance	217	$1,392$	219
Estate	35	749	79
Transportation	36	439	82

Table 1: The dataset statistic of KNOWSQL.

To uncover the knowledge-intensive text-to-SQL problem and advance the research, we construct a challenging Chinese text-to-SQL benchmark named KNOWSQL. Roughly, it consists of two parts: training/dev sets built on the existing DuSQL (Wang et al., 2020b) dataset and a newly constructed test set on three professional domains with discovered knowledge in DuSQL.

2.3.1 Building Training/Dev Set on DuSQL

We build the training/dev set of KNOWSQL based on the existing DuSQL, a Chinese multi-table text-to-SQL benchmark. We categorize its 200 databases into 16 domains like sports, energy, health care, foods, *etc*. Given the high quality of DuSQL schema and broad domain coverage, it's a satisfactory start-point to build a challenging knowledge-intensive text-to-SQL benchmark. However, the domain-specific question is not well included in DuSQL, where most of the questions could be answered easily without relying on external knowledge and only considers one SQL operation (*i.e.*, calculation). Given that, we extend the original DuSQL by adding more domain-specific questions and involving more operations in both the train set and the dev set. Eventually, KNOWSQL expands the size of DuSQL train set from 22,521 to 23,157 and the dev set from 2,482 to 2,731.

2.3.2 Building Test Set from Scratch

To simulate the professional data analysis scenario, we create a challenging test set covering three domains (finance, estate, and transportation). These three domains have high data analysis requirements in real life. Different from the train/dev sets, we construct the test set from the scratch by: (1) collecting the domain-specific tables, and (2) annotating the domain-specific questions and corresponding SQL queries.

Table Collection. For collecting table schema, we collect the tables from the following source: (1) the public annual reports of the company (2) the industry reports (3) academic papers (4) the statistical reports released by the government. To ensure the table quality, we conduct several preprocessing procedures. Firstly, we convert matrix tables (present in annual reports) into relational tables to make the question SQL-answerable. Next, to ensure the table data quality, we conduct data cleaning (*e.g.*, filtering out the irrelevant columns to simplify the table structure, and normalizing the headers to reduce the noise). Finally, to avoid data privacy issues, we conduct value anonymization (*e.g.*, removing direct identifiers and anonymizing geo-related data).

Question Annotation. It's challenging for annotators to propose the domain-specific questions without background knowledge [2]. Thus, we train the annotators first about the domain-specific knowledge via (1) collecting the jargon (*i.e.*, abbreviation, terminology) from the domain-specific open resources, which are widely adopted by domain experts (e.g., EBIT for finance) but unusual for a layperson; (2) to mimic the domain expert by asking questions using the jargon with the above materials.

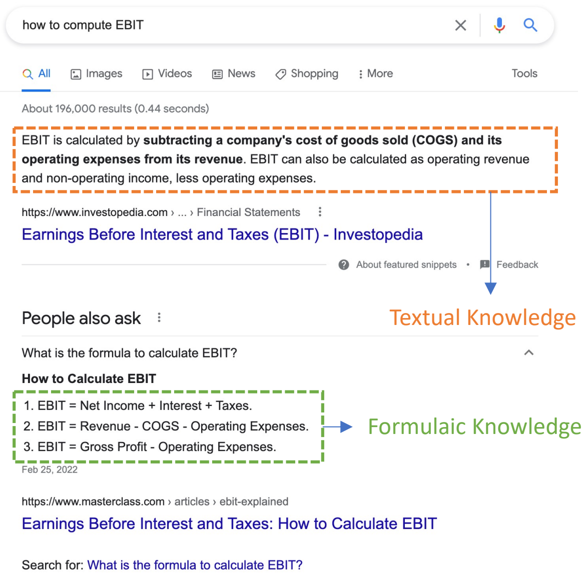
After that, the annotators would annotate the questions and SQL with the following criteria: (1) be faithful to the given table (*i.e.*, don't exceed the scope of table columns and table content); (2) not be directly answerable by the single element of the table but could be answered by the operation over existing columns; (3) limited to first-order operation (*i.e.*, excludes multi-hop questions like 'What is the gross profit?', where the table only contains 'Sales', 'Average Price' and 'Cost of Goods Sold' so that model needs to compute the 'revenue' first).

2.3.3 Dataset Quality and Data Statistic

To guarantee the data quality, we conduct a multi-rounds check. Finally, the inter-agreement of annotators reaches 94.7% [3]. During each round, we ask each annotator to review others' annotations based on the criteria (stated above), then ask them to further improve annotations that do not meet the criteria. As shown in Tab.1, the test set contains ***288*** databases and ***2,580*** questions. Notably, all these challenging data examples in the test set could be covered by ***380*** formulaic knowledge, which will be discussed in Sec. 3.

[2] See Sec. 8 for annotator payment and profile.

[3] The inter-annotator agreement is calculated as the percentage of overlapping votes about whether it's a correct and domain-specific question.



About 196,000 results (0.44 seconds)

EBIT is calculated by **subtracting a company's cost of goods sold (COGS) and its operating expenses from its revenue**. EBIT can also be calculated as operating revenue and non-operating income, less operating expenses.

https://www.investopedia.com › ... › Financial Statements

Earnings Before Interest and Taxes (EBIT) - Investopedia

What is the formula to calculate EBIT?

How to Calculate EBIT

1. EBIT = Net Income + Interest + Taxes.
2. EBIT = Revenue - COGS - Operating Expenses.
3. EBIT = Gross Profit - Operating Expenses.

Feb 25, 2022

https://www.masterclass.com › articles › ebit-explained

Earnings Before Interest and Taxes: How to Calculate EBIT

Search for: What is the formula to calculate EBIT?

Figure 2: Two types of knowledge in expressing *the calculation of EBIT*: ***textual knowledge*** and ***formulaic knowledge***.

3 Approach: Formulaic Knowledge

3.1 Motivation

When meeting unseen examples, the human may first search the related mathematical knowledge or domain knowledge from textbooks or encyclopedias. As shown in Fig. 2, the information of calculation of EBIT is returned in both textual and formulaic format. Intuitively, the formulaic format is preferred because it's (1) ***more concise and precise***: for instance, *a adds b times c* is more ambiguous than *a+b*c* or *(a+b)*c*; (2) ***easy to obtain***: most description of calculation is stored in a formulaic format in the textbook, tutorials, and academic paper; (3) ***SQL parser friendly***: the formulaic format is closed to the snippet of SQL then easily for the parser to generate[4].

3.2 Formulaic Knowledge for Text-to-SQL

Following this idea, we focus on three categories of operations (Fig. 3): calculation, union, and condition. Besides the popular *calculation knowledge*, we also consider the taxonomy information as *union knowledge* and the judgment standard as *condition knowledge*. The design insight here is that the left part is the name of the knowledge item, and the right part expresses its semantic meaning represented by operations over concepts. Note that all operations are consistent with SQL grammar, making it closer to SQL query. Besides the entity, the left part of formulaic knowledge might also be the SQL function (*e.g.*, NOW()) or constant (*e.g.*, threshold of Real Estate Bubble).

3.3 Formulaic Knowledge Bank

We further build a formulaic knowledge bank with 1,954 formulaic knowledge items, which supports 19 domains involved in KNOWSQL. Importantly, the bank covers all these examples of KNOWSQL as shown in Tab. 2. Note that this bank is a domain-related resource, not one tied to the specific database. Thus, this bank is more general and could be utilized in other applications natural language applications (*e.g.*, question answering) [5].

Criteria The design of the formulaic knowledge follows three criteria: (1) Only the first-order (flat) formulaic knowledge is considered (*i.e.*, the concept in the formulaic item should be align-able to the schema elements rather than another formulaic item) ; (2) The stored formulaic knowledge should be both faithful (*i.e.*, acknowledged by the expert) and standardized (*i.e.*, shared at the domain level); (3) The formulaic knowledge should be domain-level (*i.e.*, not tied to the specific schema elements).

Collection We collect the formulaic knowledge from the following public resource: (1) Baidu Wenku, the platform where the domain experts usually share the domain knowledge of various domain[6]; (2) CNKI, China's largest academic website[7]; (3) the data analysis websites of a specific domain, like ESPN for sports and Yahoo for finance. We also collect some knowledge from the English resource and let annotators translate this domain knowledge into Chinese.

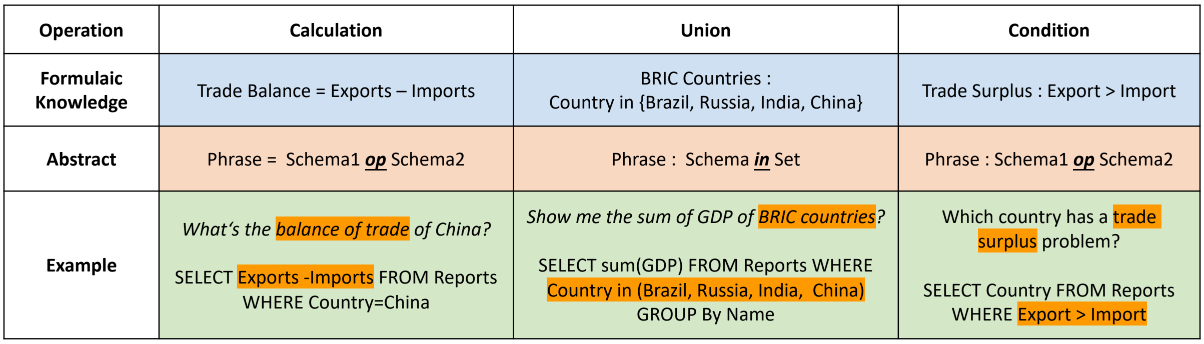
Abstraction To make the formulaic knowledge more generic, we propose to accumulate the formulaic knowledge at the domain level instead of database-specific. Specifically, we abstract the concept of formulaic knowledge before storing them in the knowledge bank, which indicates the operation over concept rather than specific schema. For example, we would extract the formulaic knowledge from 'People Density in China 2020 = total number of Chinese in 2020 / Chinese Land Area' to 'People Density = total number of People / Area'.

[4] In Sec. 6, experimental results prove that formulaic format receives better performance than textual format.

[5] See fine-grained statistic of bank in Appendix A.1.

[6] https://wenku.baidu.com/

[7] https://oversea.cnki.net/index/



Operation	Calculation	Union	Condition
Formulaic Knowledge	Trade Balance = Exports – Imports	BRIC Countries : Country in {Brazil, Russia, India, China}	Trade Surplus : Export > Import
Abstract	Phrase = Schema1 ***op*** Schema2	Phrase : Schema ***in*** Set	Phrase : Schema1 ***op*** Schema2
Example	*What's the balance of trade of China?* SELECT Exports -Imports FROM Reports WHERE Country=China	*Show me the sum of GDP of BRIC countries?* SELECT sum(GDP) FROM Reports WHERE Country in (Brazil, Russia, India, China) GROUP By Name	Which country has a trade surplus problem? SELECT Country FROM Reports WHERE Export > Import

Figure 3: We consider three types of formulaic knowledge to address knowledge-intensive text-to-SQL.

	#Formulaic	#Calculation	#Union	#Condition
Formulaic Knowledge Bank	1,954	1,102	346	506
KnowSQL involved	891	656	52	183

Table 2: The dataset statistic of formulaic knowledge bank and its overlap with KnowSQL.

Consequently, only ***ONE*** formulaic knowledge is required to address ***MANY*** schema elements to calculate the density of animals/cars/shops.

Mapping within KnowSQL We further examine the overlap between formulaic knowledge bank and KnowSQL benchmark. As stated in Sec 2.3.3, all questions from KnowSQL are covered by formulaic knowledge banks. Specifically, there are 1,954 knowledge items in the bank, and 891 items are used for answering the KnowSQL questions as shown in Table 2. Especially, there are extra 1,063 knowledge items beyond KnowSQL which could support future work in applying formulaic knowledge.

4 ReGroup Framework

To address the knowledge-intensive text-to-SQL problem, we propose a novel framework named ReGrouP, consisting of three stages: (1) **Re**trieve the formulaic knowledge from the formulaic knowledge bank as an external knowledge source; (2) **Grou**nd the concept of formulaic knowledge to the schema elements (*e.g.*, *Exports* to *Ship_Out*); (3) **P**arse the results with the question, schema, and grounded formulaic knowledge. As shown in Fig. 4, ReGroup consists of three models: retriever, grounding model, and parser. We will give a brief introduction of each model in the following[8].

4.1 Retriever Model

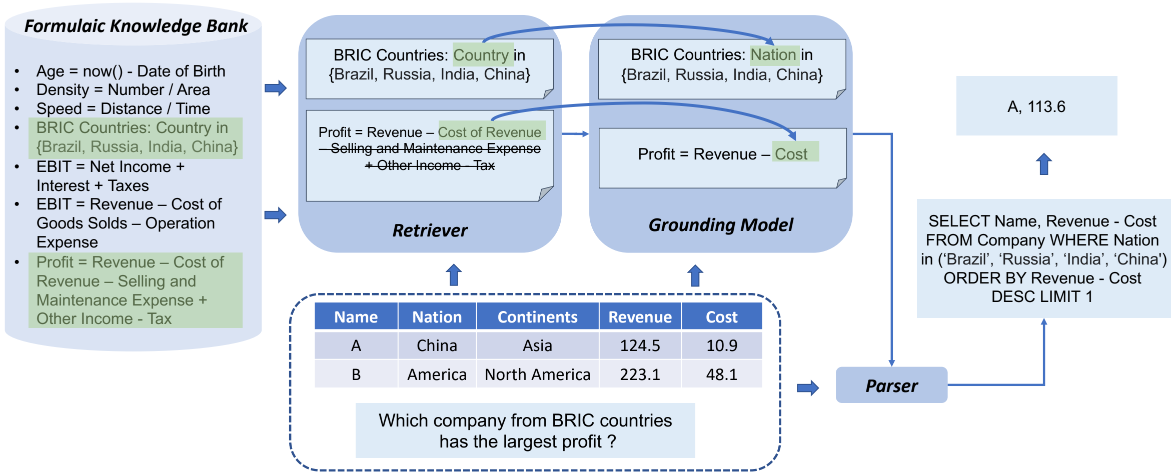
The goal of the retriever is to *extract the relevant formulaic knowledge items from the formulaic knowledge bank* (Fig. 4). The challenge is the fine-grained modeling of the formulaic knowledge to disambiguate the ones with the same intent but differing in operation over concepts, such as calculating EBIT in different ways. We directly utilize the off-the-shelf Dense Passage Retriever (DPR) (Karpukhin et al., 2020) which was originally designed for open-domain QA. It employs a bi-encoder architecture to learn the dense representation of sentences and passages, then it computes the dot-product between the representations as the similarity score.

To adapt the DPR in the formulaic knowledge retrieval task, we treat the formula knowledge bank as the passage candidate and concatenate the question with flattened schema (separated by special token '|') to enrich the semantics of the question. Then we follow the standard DPR training procedure to optimize the bi-encoder. Specifically, during the training process, we derive the positive knowledge items from KnowSQL annotation and sample five negative examples from the formulaic knowledge bank. During the inference process, we first cache the embedding of formulaic knowledge items, then leverage the FAISS algorithm (Johnson et al., 2017) to rank each formulaic knowledge item.

4.2 Grounding Model

Given the retrieved knowledge items, the goal of the grounding model is to *edit the formulaic knowl-*

[8] More details of the model implementation could be found in Appendix B.


Formulaic Knowledge Bank
Age = now() - Date of Birth
Density = Number / Area
Speed = Distance / Time
BRIC Countries: Country in {Brazil, Russia, India, China}
EBIT = Net Income + Interest + Taxes
EBIT = Revenue – Cost of Goods Solds – Operation Expense
Profit = Revenue – Cost of Revenue – Selling and Maintenance Expense + Other Income - Tax
BRIC Countries: Country in {Brazil, Russia, India, China}
Profit = Revenue – Cost of Revenue ~~– Selling and Maintenance Expense + Other Income – Tax~~
Retriever
BRIC Countries: Nation in {Brazil, Russia, India, China}
Profit = Revenue – Cost
Grounding Model
Parser
A, 113.6

Name	Nation	Continents	Revenue	Cost
A	China	Asia	124.5	10.9
B	America	North America	223.1	48.1

Which company from BRIC countries has the largest profit ?

```
SELECT Name, Revenue - Cost
FROM Company WHERE Nation
in ('Brazil', 'Russia', 'India', 'China')
ORDER BY Revenue - Cost
DESC LIMIT 1
```

Figure 4: Pipeline of REGROUP: (1) **Re**trieve the formulaic knowledge item from the bank; (2) **Grou**nd the concept of formulaic knowledge into schema elements; (3) **P**arse the question with grounded formulaic knowledge into SQL.

edge items w.r.t specific schema through (1) removing the irrelevant concept and (2) instantiating the concept with the schema elements. The main challenge is the expensive annotations of grounding (*i.e. supervision*). Therefore, the weakly supervised grounding approaches would be more suitable. Specifically, we leverage the Erasing-then-Awakening (ETA) model proposed by (Liu et al., 2021), which was originally designed for grounding the entity from the knowledge base to the entity mentioned in the question. The output of ETA is a confidence matrix, indicating the possible grounding relations between entity mentions and entities.

To adapt the ETA in the formulaic knowledge grounding task, we treat each knowledge item as the 'question' and attempt to figure out which specific schema elements are grounded in the knowledge item. Specifically, it's determined by a hyperparameter H to indicate the threshold of confidence (whether it's grounded and which one it's grounded). As shown in Fig. 4, we filter the concept (cross outed parts) under the confidence threshold H and replace the concept with aligned elements (green parts).

4.3 Parser Model

The goal of the parser is to *predict the executable SQL according to question and database schema.* The main challenge is how to model the database structure to infer the implicit schema mentioned, and how to make use of the grounded knowledge (*i.e.*, knowledge-fusion) to leverage grounded knowledge. We are inspired by the recent progress in adopting the large pre-trained language model in semantic parsing problems. For instance, (Scholak et al., 2021; Shin et al., 2021; Dou et al., 2022; Xie et al., 2022) achieves excellent performance on several semantic parsing tasks under the simple pretrained language model framework, such as BART (Lewis et al., 2020) and T5 (Raffel et al., 2020).

Given that, we propose to adopt UniSAr (Dou et al., 2022) as the base parser in this work. It improves the vanilla BART with three non-invasive extensions and achieves SOTA or competitive performance on seven text-to-SQL benchmarks. Concretely, the input of the model is the concatenation of the question, serialized schema, and retrieved formulaic knowledge. We propose that the parser should correctly adopt the grounded formulaic knowledge during SQL generation.

5 Experimental Results and Analysis

To evaluate our approach: REGROUP with external formulaic knowledge bank, we conduct several experiments on KNOWSQL benchmark. We report both the overall results of the pipeline and the fine-grained results of each module. We also conduct error analysis and categorize the bad cases into three main classes. Note that we report the average experimental results of each setting during three runs[9].

[9] Code and data are available at link.

Model	Dev	Finance	Estate	Transportation	Average
Vanilla	69.3	8.7	5.7	6.9	22.7
ReGroup (w/o Grounding)	71.7	38.1	25.1	32.7	41.9
ReGroup	74.6	43.7	46.1	39.1	50.9
ReGroup (Oracle)	78.4	71.4	84.8	64.7	74.8

Table 3: Overall results on different KnowSQL splits. Oracle refers to the use of the oracle formulaic knowledge. The evaluation metric is SQL exact set match. Average indicates the micro-average score of the first four columns.

Data	Model	R@1	R@3	R@10
Dev	BM-25	67.9	89.1	96.5
	ReGroup	73.0	89.8	96.5
Finance	BM-25	39.4	66.5	85.9
	ReGroup	46.0	68.1	86.1

Table 4: Results of ReGroup retriever compared with BM-25 on KnowSQL dev and finance splits. The evaluation metric is the Recall.

Data	Model	Precision	Recall	F1
Dev	FuzzyMatch	69.3	62.5	65.7
	ReGroup	71.3	70.4	70.8
Finance	FuzzyMatch	35.3	31.5	33.2
	ReGroup	42.9	44.7	43.8

Table 5: Results of ReGroup grounding model compared with fuzzy string match on KnowSQL dev and finance splits.

5.1 Experimental Setup

The retriever returns the top-3 retrieved formulaic knowledge items from the bank. The grounding model further aligns the concept in formulaic knowledge into schema elements and the decision threshold H is 0.6 which is decided empirically. The parser receives the grounded knowledge, table schema, and user query as the input and then outputs the SQL. For the parsing baseline, we adopt UniSAr (Dou et al., 2022) as the vanilla parser[10].

5.2 Overall Results

As shown in Tab. 3, we could observe that: (1) ReGroup exceeds the vanilla model by 28.2%, which indicates the effectiveness of using formulaic knowledge; (2) grounding the formulaic knowledge improves the ReGroup by 9.0%; (3) the oracle formulaic knowledge (retrieve correctly and grounding correctly) reaches the upper bound of ReGroup 74.8%, which implies the potential improvement room for KnowSQL.

5.3 Fine-grained Results

We compare the retriever and grounding model with other baselines, on both the dev set and the test set of KnowSQL in the finance domain, to examine the performance in general and domain-specific scenarios.

Retriever We compare the retriever of ReGroup (bi-encoder) with BM-25 (Robertson and Zaragoza, 2009). The evaluation metric is the recall score over retrieved results. We observe that the finance domain is more challenging than the general domain (dev split) since it contains many homogeneous formulaic knowledge items that express the same intention in the left part but with different computation ways in the right part. For example, there are two ways to compute the 'EBIT' in Fig. 4.

Grounding We compare the grounding model of ReGroup with the fuzzy string match-based method [11]. Following the previous work (Lei et al., 2020; Liu et al., 2021), we report the micro-average precision, recall, and F1-score. We could observe that: (1) the model-based grounding improves the performance by 5.1% and 10.6% respectively; (2) the domain-specific data like Finance poses more challenging cases than the general domain, where finance is behind the dev by about 27.0%.

5.4 Error Analysis

We sample 300 cases from the dev split and 100 cases from finance/estate/transportation in the test split respectively (600 in total) for error analysis.

Vanilla Model Error We first compare the correct case of ReGroup while predicted incorrectly by the vanilla model. As the example in the first part of Fig. 5, the vanilla model is unable to predict the unseen operation during training. In contrast, the grounded formulaic knowledge enables

[10] More implementation details could be found in Appendix B

[11] It enumerates all n-gram ($n \leq 5$) of the concepts in formulaic knowledge, and links each of them to schema element by fuzzy string matching.

<table>
<tr><th><em>Vanilla Model Error</em></th><th><em>Formulaic Knowledge</em></th></tr>
<tr><td><strong>Question: 东三省每省的一胎出生率是多少?</strong>
(What is the first birth rate in each of the three northeastern provinces in China?)
Schema : 省份 ｜ 婴儿出生率 | 二胎出生率 | 人口
(Province | Birth Rate | Second Birth Rate | Population)

Vanilla: SELECT 婴儿出生率 FROM 各省人口出生及死亡率 WHERE 省份 = "辽宁"
ReGrouP: SELECT 婴儿出生率 - 二胎出生率 FROM 各省人口出生及死亡率 WHERE 省份 IN ("辽宁" , "吉林" , "黑龙江")</td><td><strong>Grounded Formulaic Knowledge:</strong>
东三省 : { 辽宁 , 吉林 , 黑龙江 }
(Three Northeastern Provinces: { Liaoning , Jilin , Heilongjiang })

一胎出生率 = 婴儿出生率 - 二胎出生率
(First birth rate = Birth rate - Second Birth Rate)</td></tr>
<tr><th><em>Retriever Error (43%)</em></th><th><em>Retrieval Knowledge</em></th></tr>
<tr><td><strong>Question:息税前利润是多少?</strong>
(Please return the Earnings Before Interest and Taxes)
<strong>Schema</strong>: 收入 | 净收入 | 销售费用 |营业费用 | 销售额
(Revenue| Net Income | Cost of Goods Sold Expenses | Operating Expenses | Sales)

<strong>Gold SQL</strong>: SELECT 收入 - 销售费用 - 营业费用 FROM 报表
<strong>Pred SQL</strong>: SELECT 净收入 + 销售额 FROM 报表</td><td><strong>Oracle Formulaic Knowledge:</strong>
息税前利润 = 收入 - 销售成本 - 营业费用
(Earnings Before Interest and Taxes = Revenue – Cost of Goods Sold – Operating Expenses)
<strong>Retrieved Formulaic Knowledge</strong>:
息税前利润 = 净收入 + 利息 + 税
(Earnings Before Interest and Taxes = Net Income + Interest + Taxes)</td></tr>
<tr><th><em>Grounding Error (41%)</em></th><th><em>Grounded Knowledge</em></th></tr>
<tr><td><strong>Question: A公司的流动资产是多少?</strong>
(What is company A's current assets?)
<strong>Schema</strong>:现金 | 应收款项 | 可销售证券|商品成本| 运营费用
(Cash | Trade Receivables | Marketable Securities | Cost of Goods | Operating Expenses)

<strong>Gold SQL</strong>: SELECT 应收款项 + 可销售证券 +现金 FROM 报表
<strong>Pred SQL</strong>: SELECT 应收款项 + 现金 FROM 报表</td><td><strong>Undergrounded Formulaic Knowledge:</strong>
流动资产 = 短期资本 + 应收帐款 + 股票 + 存款余额
(Current Assets = Short Term Capital + Debtors + Stock + Cash and bank)
<strong>Correct Grounded Formulaic Knowledge</strong>:
流动资产 = 应收款项 + 可销售证券 + 现金
(Current Assets = Trade Receivables + Marketable Securities + Cash)
<strong>Prediced Grounded Formulaic Knowledge</strong>:
流动资产 = 应收款项 + 现金
(Current Assets = Trade Receivables + Cash)</td></tr>
<tr><th><em>Parser Error (12%)</em></th><th><em>Leveraging Knowledge</em></th></tr>
<tr><td><strong>Question: 哪个城市的房地产市场发展合理?</strong>
(Which city's real estate market is developing reasonably?)
<strong>Schema</strong>: 城市 | 吸纳率 | 空置率
(City | Commercial Housing Absorption Rate | Commercial Housing Vacancy Rate)

<strong>Gold SQL</strong>: SELECT 城市 FROM 报表 where 空置率 > 15% and 空置率 < 30%
<strong>Pred SQL</strong>: SELECT 城市 FROM 报表 where 空置率 > 15%</td><td><strong>Grounded Formulaic Knowledge:</strong>
房地产市场良性发展 : 空置率 > 15% AND 空置率 < 30%
(Good development of real estate market: Commercial Housing Vacancy Rate > 15% AND Commercial Housing Vacancy Rate < 30%)</td></tr>
</table>

Figure 5: Case studies of ReGroup. We first compare it with the vanilla parsing model. Then we classify the bad cases of ReGroup into three categories: (1) Retriever Error: *not getting the knowledge from bank*; (2) Grounding Error: *not learning the knowledge by alignment*; (3) Parser Error: *not using the grounded knowledge in generation*.

ReGroup to predict the operation over schema elements correctly.

Then we categorize the error of ReGroup into three main classes and list their percentage in Fig. 5. Finally, we discuss the potential future work in improving each part of ReGroup. An advantage of ReGroup is the decoupled framework could track each type of bad case individually, avoiding the catastrophic forgetting problem.

Retrieval Error About 43% errors are attributed to the retriever where the model *doesn't get the correct knowledge from bank* since it can't distinguish the semantic difference between the closed formulaic knowledge items. Future work should improve its distinguishing ability by fine-grained modelings, like attention mechanism Huang et al. (2019).

Grounding Error About 41% errors are caused by incorrect grounded knowledge where the model *doesn't learn the knowledge by alignment* since it can't correctly align the concept to schema elements. Future work should focus on how to derive the grounding information under weak supervision or even without supervision. It would greatly alleviate the severe annotation effort in specific domains.

Parsing Error There are still 8% error cases caused by parsing, where the formulaic knowledge is correctly retrieved and grounded but the parser still *doesn't use the grounded knowledge in generation* well. Future work should improve it by explicitly modeling the copy process of knowledge from the input to the SQL snippet position, such as implementing the additional gate mechanism.

Other Error The remaining 8% errors are about the SQL generation, such as the GROUP-BY or nested SQL. Since it's not our main focus, we ignore these cases in Fig. 5 for brevity.

6 Discussion

Is formulaic knowledge better than textual knowledge for text-to-SQL? In Sec.3.1, we argue that formulaic knowledge is preferred over textual knowledge intuitively. Empirically, we conduct the experiments by the following steps: (1) transforming the formulaic knowledge to textual knowledge through annotators; (2) training the retriever and parser with textual knowledge under the same experiment setting as formulaic knowledge. Experimental results reveal that textual knowledge receives an overall performance degradation of 13.6% compared with Table 3. We conclude that REGROUP prefers formulaic knowledge since it's more close to the SQL snippets or schema representation. Moreover, formulaic knowledge is both precise and concise. In contrast, textual knowledge is redundant and much more diverse in expressing the equivalent meaning.

What's the cost of collecting formulaic knowledge? During the collection process of formulaic knowledge bank (19 domains), we found most domains have the public knowledge resource. Moreover, the effort spent on collection formulaic knowledge is also acceptable compared with annotating data examples. For example, we spent ***4 hours*** in collecting ***219*** formulaic knowledge in the finance domain, which is far more effective than annotating the equivalent data examples. Eventually, formulaic knowledge improves the performance by 35.0% without retraining the model as shown in Table 3 (from 8.7% to 43.7%).

How to expand the scope of formulaic knowledge further? In this paper, we mainly focus on domain knowledge and mathematical knowledge and transfer them into formulaic knowledge format for model learning. Other types of knowledge would improve the knowledge-intensive text-to-SQL further, such as the commonsense knowledge (*e.g.*, water freezing point: temperature=0 °C) or personalized information (*e.g.*, *favourite food: Tiramisu*). Thus, we could package these types of knowledge into a formulaic format in future work.

7 Related Work

7.1 Domain Generalization of Text-to-SQL

To be applicable in real scenarios, a text-to-SQL model should generalize to new domains without relying on expensive domain-specific labeled data. Previous work has shown that current text-to-SQL usually fails on domain generalization scenarios (Finegan-Dollak et al., 2018). Recent approaches track this problem including data synthesis (Yin et al., 2021), meta-learning (Wang et al., 2021) and encoder pretraining (Yin et al., 2020; Herzig et al., 2020). Most recently, Zhao et al. (2022) proposed to adopt schema expansion and scheme pruning to preprocess the table schemas.

We highlight that compared with the schema-expansion approach, the advantage of our approach (REGROUP with formulaic knowledge) is the broad knowledge scope: we not only consider the calculation knowledge but also union knowledge and condition knowledge. Moreover, our approach is extensible with an external and maintainable formulaic knowledge bank.

7.2 Retrieval Enhanced Semantic Parsing

There has been a recent trend toward leveraging retrieval-enhanced methods in various NLP tasks such as machine translation (Cai et al., 2019) and question answering (Karpukhin et al., 2020). Similar with REGROUP, previous work (Gupta et al., 2022; Pasupat et al., 2021) leverage a retrieval step to provides examples as the context of input for seq2seq model learning.

However, our approach differs in two ways: (1) our retrieval object is grounded formulaic knowledge which contains more condensed information than data example; (2) prior work directly leverage the retrieved results. We leverage the grounding model to edit the retrieved formulaic knowledge to make it more relevant to the question and schema.

8 Conclusion and Future Work

This paper explores formulaic knowledge to address the knowledge-intensive text-to-SQL problem, which would advance the professional application of text-to-SQL such as data analysis for domain experts. First, we analyze the challenge of knowledge-intensive text-to-SQL and construct a new challenging benchmark KNOWSQL. Then we propose to address this problem from the view of formulaic knowledge. Concretely, we propose a simple framework REGROUP to leverage an external formulaic knowledge bank. Experimental results reveal that REGROUP with formulaic knowledge achieves the 28.2% improvements overall.

We further discuss three directions in improving the REGROUP via analyzing different types of bad cases: (1) iterative filling in the blank of formulaic knowledge bank; (2) mitigating the gap between formulaic knowledge and specific schema via improving the grounding model; (3) driving the parser to fully make use of more complicated (*e.g.*, commonsense) formulaic knowledge.

Ethical Considerations

This work presents KNOWSQL, a free and open dataset for the research community to study the knowledge-intensive text-to-SQL problem. Data in KNOWSQL are constructed based on DuSQL (Wang et al., 2020b) , a free and open cross-database Chinese text-to-SQL dataset. We also collect formulaic and table data from CNKI[12] and Baidu Wenku[13], which are also free and open for academic usage. The content of the table is anonymized to address the privacy issue. To annotate the KNOWSQL, we recruit 3 Chinese college students (1 female and 2 males). Each student is paid 4 yuan ($0.6 USD) for annotating the (SQL, question) pairs and 2 yuan ($0.3USD) for collecting the formulaic knowledge items. This compensation is determined according to the prior similar dataset construction (Guo et al., 2021). Since all question sequences are collected against open-access databases or public tables, there is no privacy issue.

Limitations

(1) KNOWSQL is built based on DuSQL, a Chinese large-scale text-to-SQL dataset. Thus the language coverage of this paper is limited to Chinese. We leave the extension to other languages for future work. (2) For the scope of formulaic knowledge, we mainly address three types of knowledge to associate with each SQL phrase: calculation, union, and condition. Some types of knowledge are under-explored such as commonsense knowledge. (3) For the model design of REGROUP, we build it from improving many existing works. Despite achieving promising evaluation results, the case studies reveal that many challenging remains during the retrieval, grounding, or parsing.

Acknowledgement

We thank all anonymous reviewers for their constructive comments. Wanxiang Che was supported via the grant 2020AAA0106501 and NSFC grants 62236004 and 61976072. Dechen Zhan is the corresponding author.

References

Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, and Shuming Shi. 2019. Retrieval-guided dialogue response generation via a matching-to-generation framework. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1866–1875. Association for Computational Linguistics.

Ruisheng Cao, Lu Chen, Zhi Chen, Yanbin Zhao, Su Zhu, and Kai Yu. 2021. LGESQL: Line graph enhanced text-to-SQL model with mixed local and non-local relations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2541–2555. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Longxu Dou, Yan Gao, Mingyang Pan, Dingzirui Wang, Wanxiang Che, Dechen Zhan, and Jian-Guang Lou. 2022. Unisar: A unified structure-aware autoregressive language model for text-to-sql. arXiv.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360. Association for Computational Linguistics.

Jiaqi Guo, Ziliang Si, Yu Wang, Qian Liu, Ming Fan, Jian-Guang Lou, Zijiang Yang, and Ting Liu. 2021. Chase: A large-scale and pragmatic Chinese dataset for cross-database context-dependent text-to-SQL. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2316–2331, Online. Association for Computational Linguistics.

Vivek Gupta, Akshat Shrivastava, Adithya Sagar, Armen Aghajanyan, and Denis Savenkov. 2022. RetroNLU: Retrieval augmented task-oriented semantic parsing. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 184–196. Association for Computational Linguistics.

Jonathan Herzig and Jonathan Berant. 2019. Don't paraphrase, detect! rapid and effective data collection for semantic parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3810–3820, Hong Kong, China. Association for Computational Linguistics.

[12] https://oversea.cnki.net/index/

[13] https://wenku.baidu.com/

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333. Association for Computational Linguistics.

Hsin-Yuan Huang, Eunsol Choi, and Wen-tau Yih. 2019. Flowqa: Grasping flow in history for conversational machine comprehension. In *7th International Conference on Learning Representations, ICLR*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. arXiv.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan, and Tat-Seng Chua. 2020. Re-examining the role of schema linking in text-to-SQL. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6943–6954, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Qian Liu, Dejian Yang, Jiahui Zhang, Jiaqi Guo, Bin Zhou, and Jian-Guang Lou. 2021. Awakening latent grounding from pretrained language models for semantic parsing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1174–1189.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Panupong Pasupat, Yuan Zhang, and Kelvin Guu. 2021. Controllable semantic parsing via retrieval augmentation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7683–7698.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Stephen Robertson and Hugo Zaragoza. 2009. *The Probabilistic Relevance Framework: BM25 and Beyond*. Foundations and Trends in Information Retrieval.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901.

Richard Shin, Christopher H. Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. *CoRR*, abs/2104.08768.

Bailin Wang, Mirella Lapata, and Ivan Titov. 2021. Meta-learning for domain generalization in semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–379. Association for Computational Linguistics.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Lijie Wang, Ao Zhang, Kun Wu, Ke Sun, Zhenghua Li, Hua Wu, Min Zhang, and Haifeng Wang. 2020b. DuSQL: A large-scale and pragmatic Chinese text-to-SQL dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6923–6935, Online. Association for Computational Linguistics.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342, Beijing, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir Radev, Caiming Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *arXiv preprint arXiv:2201.05966*.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426. Association for Computational Linguistics.

Pengcheng Yin, John Wieting, Avirup Sil, and Graham Neubig. 2021. On the ingredients of an effective zero-shot semantic parser. arXiv.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019a. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1962–1979, Hong Kong, China. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. 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*, pages 3911–3921.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523, Florence, Italy. Association for Computational Linguistics.

Jichuan Zeng, Xi Victoria Lin, Steven C.H. Hoi, Richard Socher, Caiming Xiong, Michael Lyu, and Irwin King. 2020. Photon: A robust cross-domain text-to-SQL system. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 204–214. Association for Computational Linguistics.

Chen Zhao, Yu Su, Adam Pauls, and Emmanouil Antonios Platanios. 2022. Bridging the generalization gap in text-to-SQL parsing with schema expansion. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5568–5578. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A Details of Formulaic Knowledge Bank

A.1 Knowledge Source

We construct the formulaic knowledge bank across 19 domains following KnowSQL and 1 misc domain. The misc domain stores the infrequent or general knowledge items in KnowSQL, such as the calculation of density, and speed. In the following, we will briefly analyze the collected bank.

A.2 Statistic Across Domain

Different domains have different amounts of publicly available data online. As shown in Fig. 6, not unsurprisingly, finance and estate share the most plentiful publicly available resource.

A.3 Distribution within Domain

We also observe the different distribution of knowledge across domains. If the domain focus on *calculation* (*e.g.*, finance report and fund), we assume the data analysis tends to be more ***objective***, which is easier for model learning. If the domain focus on *condition* (*e.g.*, estate and awards), we assume the data analysis tends to be more ***subjective*** since it's more challenging in learning semantics.

B Implementation Details of ReGroup

Retriever We implement the retriever based on the code of Karpukhin et al. (2020)[14]. We adopt the Chinese BERT-wwm-ext (Cui et al., 2021) as pretrained encoder[15]. It would return the top-3 retrieved formulaic knowledge. Future work could improve the negative sampling by in-batch sampling or BM25-based sampling following Karpukhin et al. (2020).

Grounding Model The code of ETA[16] is not released at the time of submission of this paper. We re-implement the ETA model based on the paper using pytorch (Paszke et al., 2019). We evaluate our implemented model with the original model on Spider-L (Lei et al., 2020) to examine whether the re-implemented model works. Our model achieves 82.1% column F1 score where Liu et al. (2021) reported 82.5%. The experiments on KnowSQL also employ the Chinese BERT.

Parser We build the paper based on the code of Dou et al. (2022)[17] We choose the mBART-CC25[18] as the base model to fine-tune. Following the vanilla model, we build the input of parser as follows: *[schema] | [grounded formulaic knowledge] | [question]*, where '|' is the delimiter across different parts.

Resource and Tools For tokenization, we employ Stanza (Qi et al., 2020) considering its excellent performance. For the retriever and grounding model, we import the BERT model with Transformer library (Wolf et al., 2020). For parser mode, we preprocess the data and fine-tune the mBART with fairseq framework (Ott et al., 2019)

Device and Training Time We conduct all these experiments on one NVIDIA TESLA V100-32GB GPU. The training of the retriever, grounding model, and parser takes about 4 hours, 3 hours, and 8 hours respectively. The minimum device requirement is NVIDIA TESLA P100-16G to fine-tune mBART.

Hyper-parameters All the hyper-parameters are kept the same as cited paper of each model. The only difference is the *batch size* of the retriever and grounding model, we turn it into the maximum number to fit in the NVIDIA TESLA V100-32G GPU.

[14] Code of DPR Retrieval Model

[15] Chinese-BERT-wwm Model

[16] Code of ETA Grounding Model

[17] Code of UniSAr Parser

[18] mBART Model

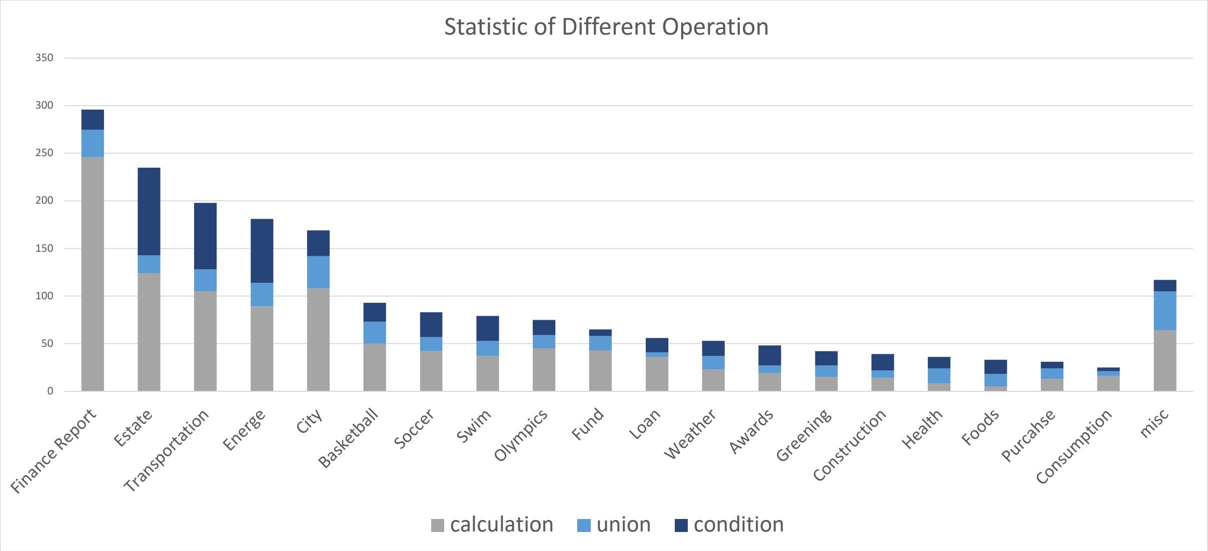


Figure 6: Statistic of three operations in different domains.

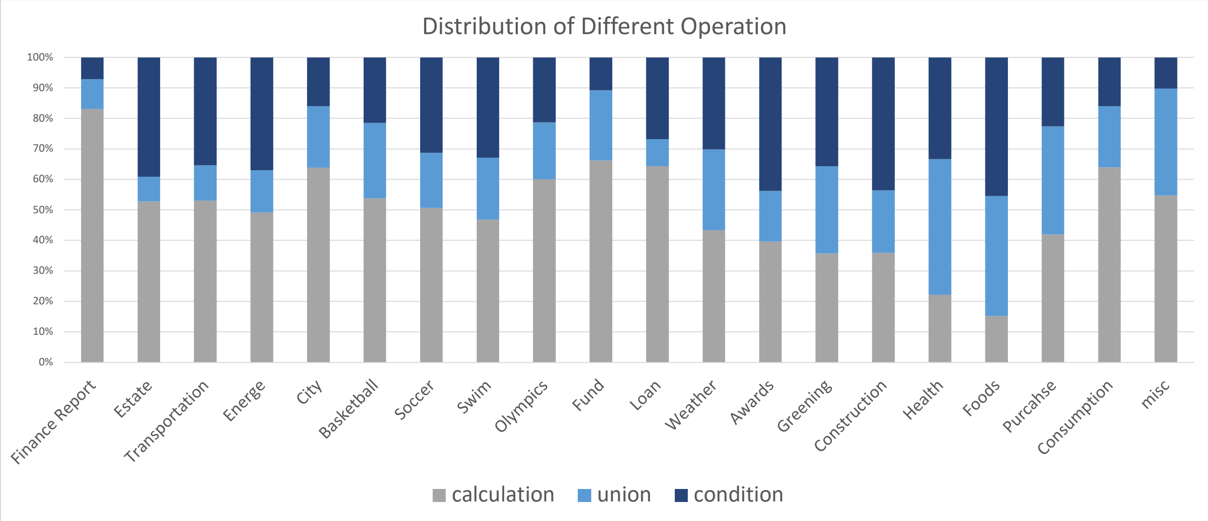


Figure 7: Distribution of three operations in different domains.