EMNLP 2022

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

Work done during the internship of Microsoft Research Asia

Overview

[New Task]
We define the **knowledge-intensive text-to-SQL** task for professional applications.

[New Methodology]
We explore to address this problem from **knowledge-centric** rather than data-centric.

[New Framework]
We propose the ReGroup framework to be **knowledge-extensible** without retraining.

Motivation of Task

Apple Inc. (AAPL)
NasdaqGS - NasdaqGS Real-time price. Currency in USD

Add to watchlist

148.79 -1.25 (-0.83%) At close: 04:00PM EST

148.98 +0.19 (+0.13%) After hours: 07:59PM EST

Summary | Chart | Statistics | Historical data | Profile | **Financials** | Analysis | Options | Holders | Sustaina

Show: **Income statement** | Balance sheet | Cash flow

Income statement All numbers in thousands

Breakdown	TTM	29/09/2022	29/09/2021	29/09/2020	29/09/2019
Total revenue	394,328,000	394,328,000	365,817,000	274,515,000	260,174,000
Cost of revenue	223,546,000	223,546,000	212,981,000	169,559,000	161,782,000
Gross profit	170,782,000	170,782,000	152,836,000	104,956,000	98,392,000
Operating expenses					
Research development	26,251,000	26,251,000	21,914,000	18,752,000	16,217,000
Selling general and administrative	25,094,000	25,094,000	21,973,000	19,916,000	18,245,000
Total operating expenses	51,345,000	51,345,000	43,887,000	38,668,000	34,462,000
Operating income or loss	119,437,000	119,437,000	108,949,000	66,288,000	63,930,000
Interest expense	2,931,000	2,931,000	2,645,000	2,873,000	3,576,000
Total other income/expenses net	-228,000	-228,000	60,000	-87,000	422,000
Income before tax	119,103,000	119,103,000	109,207,000	67,091,000	65,737,000
Income tax expense	19,300,000	19,300,000	14,527,000	9,680,000	10,481,000
Income from continuing operations	99,803,000	99,803,000	94,680,000	57,411,000	55,256,000
Net income	99,803,000	99,803,000	94,680,000	57,411,000	55,256,000

Formula and Calculation for Earnings Before Interest and Taxes (EBIT)

$$\text{EBIT} = \text{Revenue} - \text{COGS} - \text{Operating Expenses}$$

Or

$$\text{EBIT} = \text{Net Income} + \text{Interest} + \text{Taxes}$$

where:

$$\text{COGS} = \text{Cost of goods sold}$$

Credits to Yahoo!Finance and Investopedia.com

*Q: What's the **EBIT** of Apple in **Q3**?*

It's useful for assisting data analyst and advancing business intelligence.

However, existing general-domain QA system can't support this domain-specific question.

In **professional** data analysis applications, models require **external knowledge**.

We formulate it as **Knowledge-Intensive Text-to-SQL.**

Task Definition

Nation	GDP	Population	Import	Export
A	1472000	130	1550	2650
B	5040000	200	581	623

General-domain Question

What's the maximum GDP?

SELECT MAX(GDP) FROM Reports

What about the A's Population?

SELECT Population FROM Reports WHERE Nation='A'

Domain-specific Question

*What's the **balance of trade** of **BRIC countries**?*

*SELECT **Export – Import**, Nation FROM Reports WHERE **Nation in ('Brazil', 'Russia', 'India', 'China')***

*Show me the import of all **developed countries**?*

*SELECT Nation, Import FROM Reports WHERE **GDP/Population > 20000***

Input

- Database
  - Table + Headers + Values
  - Single/Multi-Table
- Question
  - Terminalogy

Output

- SQL
  - Grammaly correct
  - Be faithful to schema

Resource for Task: KnowSQL Benchmark

	#DB	#Question
Train	160	23,157
Dev	40	2,731
Finance	217	1,392
Estate	35	749
Transportation	36	439

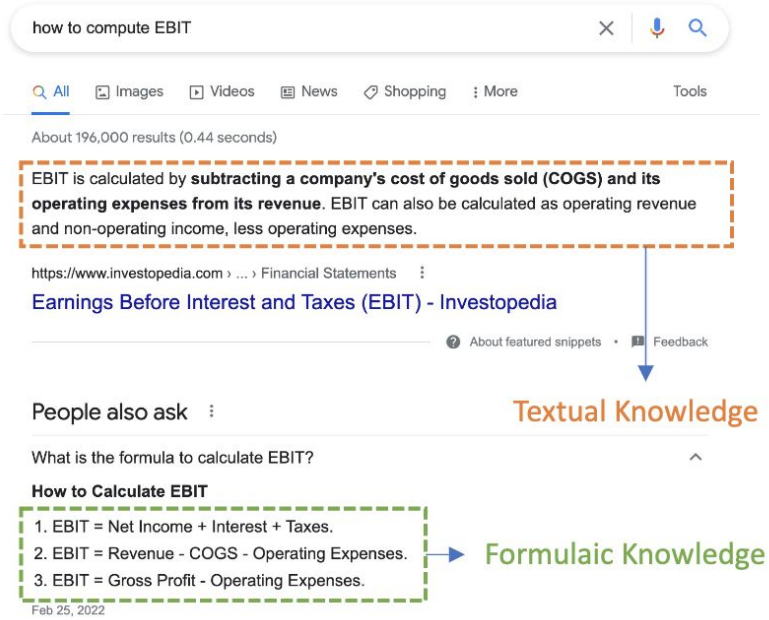
Train/Dev are built on the top of DuSQL(Chinese).

We annotate the **challenging test-set** covering

- Finance
- Estate
- Transportation

Motivation of Methodology

*Q: What's the **EBIT** of Apple in **Q3**?*



When we(as human) meets unknown terminology, we will adopt the search engine to find the necessary domain knowledge

Compared with textual knowledge, formulaic knowledge is preferred:

- Concise and precise
- SQL-closed

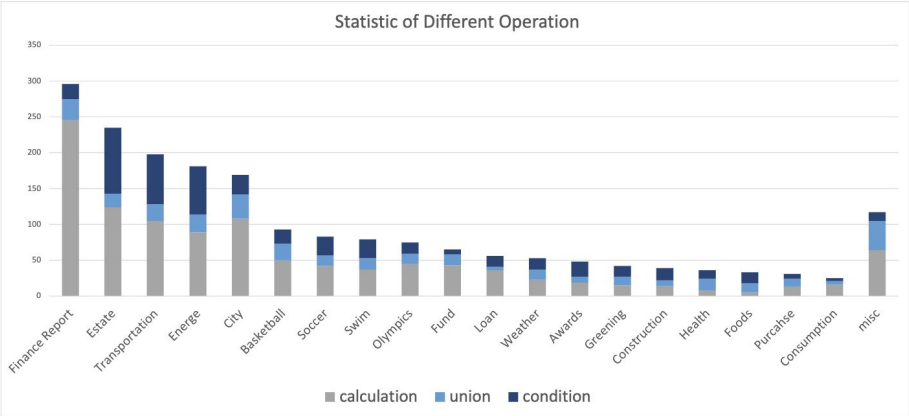
Approache: Formulaic Knowledge

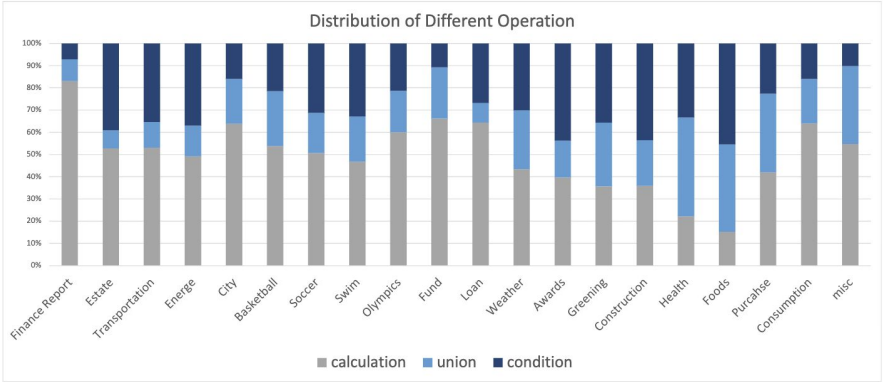
Operation	Calculation	Union	Condition
Formulaic Knowledge	Trade Balance = Exports – Imports	BRIC Countries : Country in {Brazil, Russia, India, China}	Trade Surplus : Export > Import
Abstract	Phrase = Schema1 ***op*** Schema2	Phrase : Schema ***in*** Set	Phrase : Schema1 ***op*** Schema2
Example	*What's 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

Abstract the formulaic knowledge to make it more **generlizaliable**:

- Grounded Formula: People Density in China 2020 = total number of Chinese in 2020 / Chinese Land Area
- Generic Formula: People Density = total number of People / Area

Resource: Formulaic Knowledge Bank





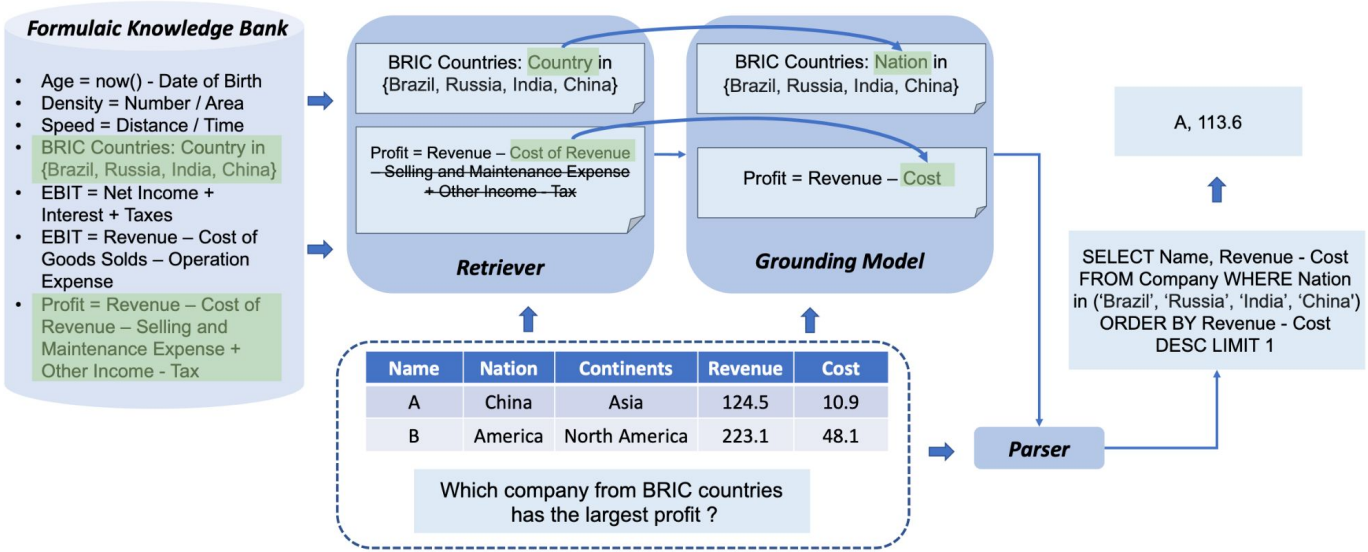
	#Formulaic	#Calculation	#Union	#Condition
Formulaic Knowledge Bank	1,954	1,102	346	506
KNOWSQL involved	891	656	52	183

Finance and estate share the most plentiful publicly available resource.

More objective: focus on calculation (finance and fund)

More subjective: focus on condition (estate and awards)

Framework: ReGrouP



(1) **Retrieve** the formulaic knowledge items from the bank;

(2) **Ground** the concepts of formulaic knowledge into schema elements;

(3) **Parse** the question with grounded formulaic knowledge into SQL.

Related Work

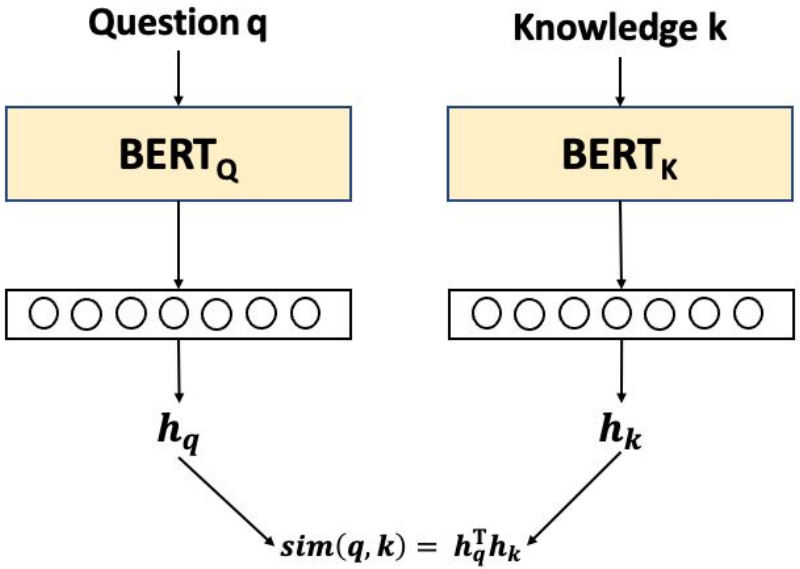
Domain-Generalization of Text-to-SQL

- **Data-Centric**: data-synthese, meta-learning, table-encoder pretraining
- Ours: **Knowledge-Centric**, benefit from broader knowledge scope

Retrieval-Enhanced Semantic Parsing

- Retrieval **data examples** as the context of input for model learning
- Ours: the retrieved **knowledge** would be futher **grounded** to the input

Framework: Knowledge Retriever



Dense retriever model is based on **bi-encoder** architecture.

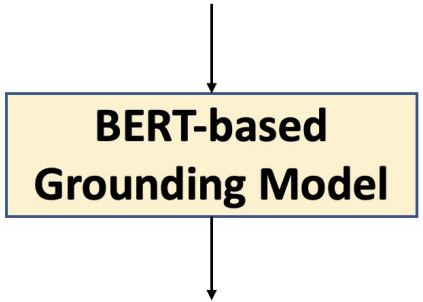
Dense retriever component for inference time logic is based on **FAISS** index.

https://github.com/facebookresearch/DPR

Framework: Knowledge Grounding Model

Profit = Revenue – Cost of Revenue – Selling and Maintenance Expense + Other Income - Tax

Company	Nation	Revenue	Cost
A	China	124.5	10.9
B	America	22.3	48.1

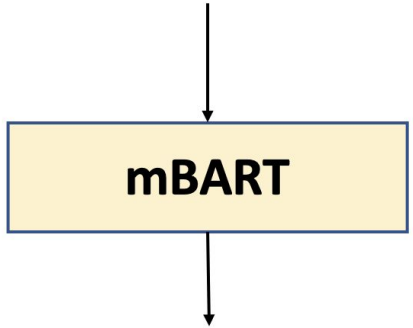


	Company	Nation	Revenue	Cost
Revenue	0.1	0.1	**0.7**	0.1
Cost of Revenue	0.1	0.1	0.3	**0.5**
~~Selling and …~~	~~0.2~~	~~0.3~~	~~0.3~~	~~0.2~~
~~Other Income~~	~~0.2~~	~~0.2~~	~~0.3~~	~~0.3~~
~~Tax~~	~~0.3~~	~~0.2~~	~~0.2~~	~~0.3~~

https://github.com/microsoft/ContextualSP/tree/master/awakening_latent_grounding

Framework: Text-to-SQL Parser with Knowledge

[Question] : Which company from BRIC countries has the largest profit?
[Schema]: Name | Nation | Revenue | Cost
[Knowledge]: BRIC Countries : Nation in {Brazil, Russia, India, China } | Profit = …..



SELECT Name, Revenue – Cost FROM Company WHERE ….

[Question], **[Schema]** and **[Knowledge]** are special tokens as the delimiters of the input.

https://github.com/microsoft/ContextualSP/tree/master/unified_parser_text_to_sql

Evaluation

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

Evaluation metric: exact match accuracy ; Oracle setting: ground-truth knowledge

ReGrouP exceeds the vanilla model (i.e., only parser without knowledge) by **28.2%**

Grounding the formulaic knowledge improves the model by **9.0%.**

Case Studies

Vanilla Model Error	*Formulaic Knowledge*
Question: 东三省每省的一胎出生率是多少? (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")	**Grounded Formulaic Knowledge:** 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)
Retriever Error (43%)	***Retrieval Knowledge***
Question:息税前利润是多少? (Please return the Earnings Before Interest and Taxes) **Schema**: 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) Gold SQL: SELECT 收入 - 销售费用 - 营业费用 FROM 报表 Pred SQL: SELECT 净收入 + 销售额 FROM 报表	Oracle Formulaic Knowledge: 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) Retrieved Formulaic Knowledge: 息税前利润 = 净收入 + 利息 + 税 (Earnings Before Interest and Taxes = Net Income + Interest + Taxes)
Grounding Error (41%)	***Grounded Knowledge***
Question: A公司的流动资产是多少? (What is company A's current assets?) **Schema**: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) Gold SQL: SELECT 应收款项 + 可销售证券 +现金 FROM 报表 Pred SQL: SELECT 应收款项 + 现金 FROM 报表	**Undergrounded Formulaic Knowledge:** 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) Correct Grounded Formulaic Knowledge: 流动资产 = 应收款项 + 可销售证券 + 现金 (Current Assets = Trade Receivables + Marketable Securities + Cash) Prediced Grounded Formulaic Knowledge: 流动资产 = 应收款项 + 现金 (Current Assets = Trade Receivables + Cash)
Parser Error (12%)	***Leveraging Knowledge***
Question: 哪个城市的房地产市场发展合理? (Which city's real estate market is developing reasonably?) **Schema**: 城市 | 吸纳率 | 空置率 (City | Commercial Housing Absorption Rate | Commercial Housing Vacancy Rate) Gold SQL: SELECT 城市 FROM 报表 where 空置率 > 15% and 空置率 < 30% Pred SQL: SELECT 城市 FROM 报表 where 空置率 > 15%	**Grounded Formulaic Knowledge:** 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%)

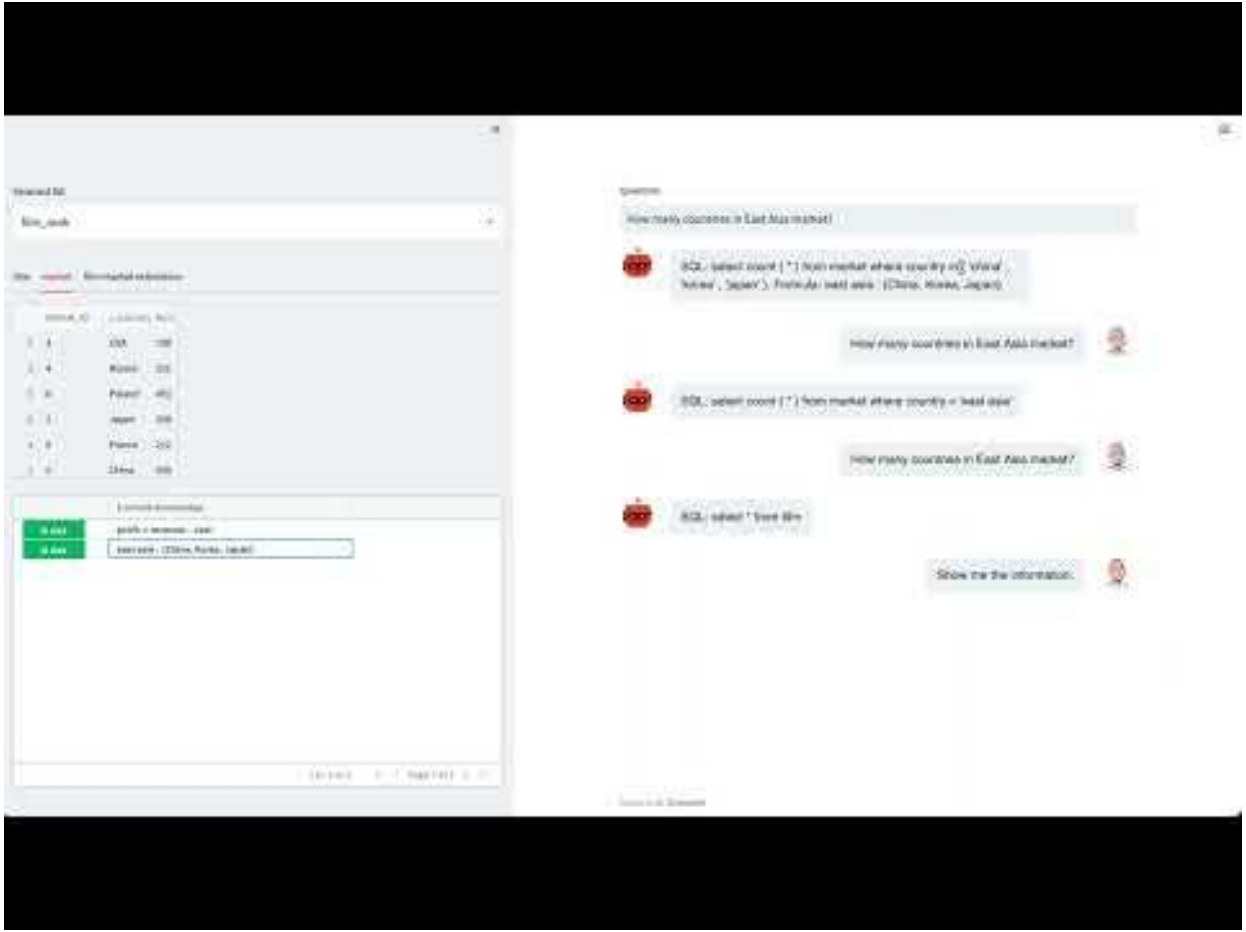
(0) ReGrouP really works better than vanilla model!

(1) improve the retriever by fine-grained modeling.

(2) derive the grounding information under weak supervision.

(3) explicitly modeling the copy process of knowledge.

Demonstration



Discussion

- How to collect the formulaic knowledge efficiently and what's the cost?
  - **[Source]** Google -> relevant enclyopediea and tutorials
  - **[Cost]** Fours hours for collecting 219 finance knowledge items

- Formulaic knowledge vs. textual knowledge: which one is preferred for BART parser?
  - Textual knowledge receives an overall performance degradation of **13.6%**

Future Work

(1) Iterative filling in the blank of formulaic knowledge bank (interactively or automatically);

(2) Improve the grounding model to close the gap between formulaic knowledge and specific schema;

(3) Extend the bank to more complicated (e.g., commonsense and personalized) formulaic knowledge. Such as “Favorite food: Tiramisu”.

Contribution

- **[New Task and Benchmark]**
  We define the task of knowledge-intensive text-to-SQL and propose KnowSQL.

- **[New Knowledge Resource]**
  We explore the usage of formulaic knowledge and build a knowledge bank.

- **[New Framework]**
  Out proposed ReGroup framework achieves the 28.2% improvements overall.

Paper/Slides/Code in https://longxudou.github.io/

EMNLP 2022

Thanks!

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**