



利用大模型提升学生工程实践能力

张静

中国人民大学



大模型时代

- 以Transformer为基础的语言大模型、代码大模型、多模态大模型，在自然语言、代码以及图像理解与生成方面展现出令人惊叹的能力



自然语言



```
1 <script>
2 (function(){
3     var bp = document.createElement('script');
4     var curProtocol = window.location.protocol.split(':')[0];
5     if (curProtocol === 'https'){
6         bp.src = 'https://zz.bdstatic.com/linksubmit/push.js';
7     }
8     else{
9         bp.src = 'http://push.zhanzhang.baidu.com/push.js';
10    }
11    var s = document.getElementsByTagName("script")[0];
12    s.parentNode.insertBefore(bp, s);
13 })();
14 </script>
```

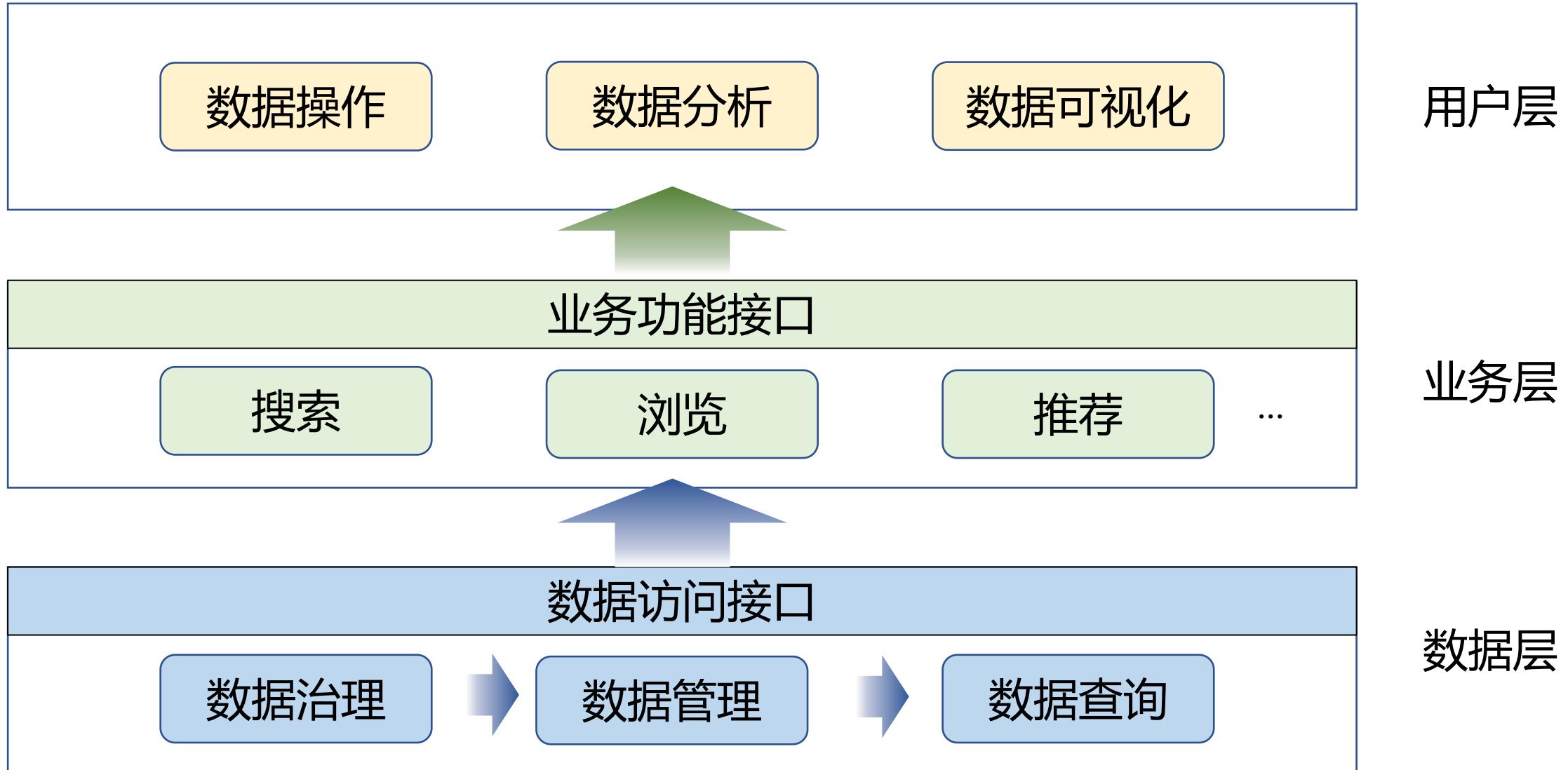
代码



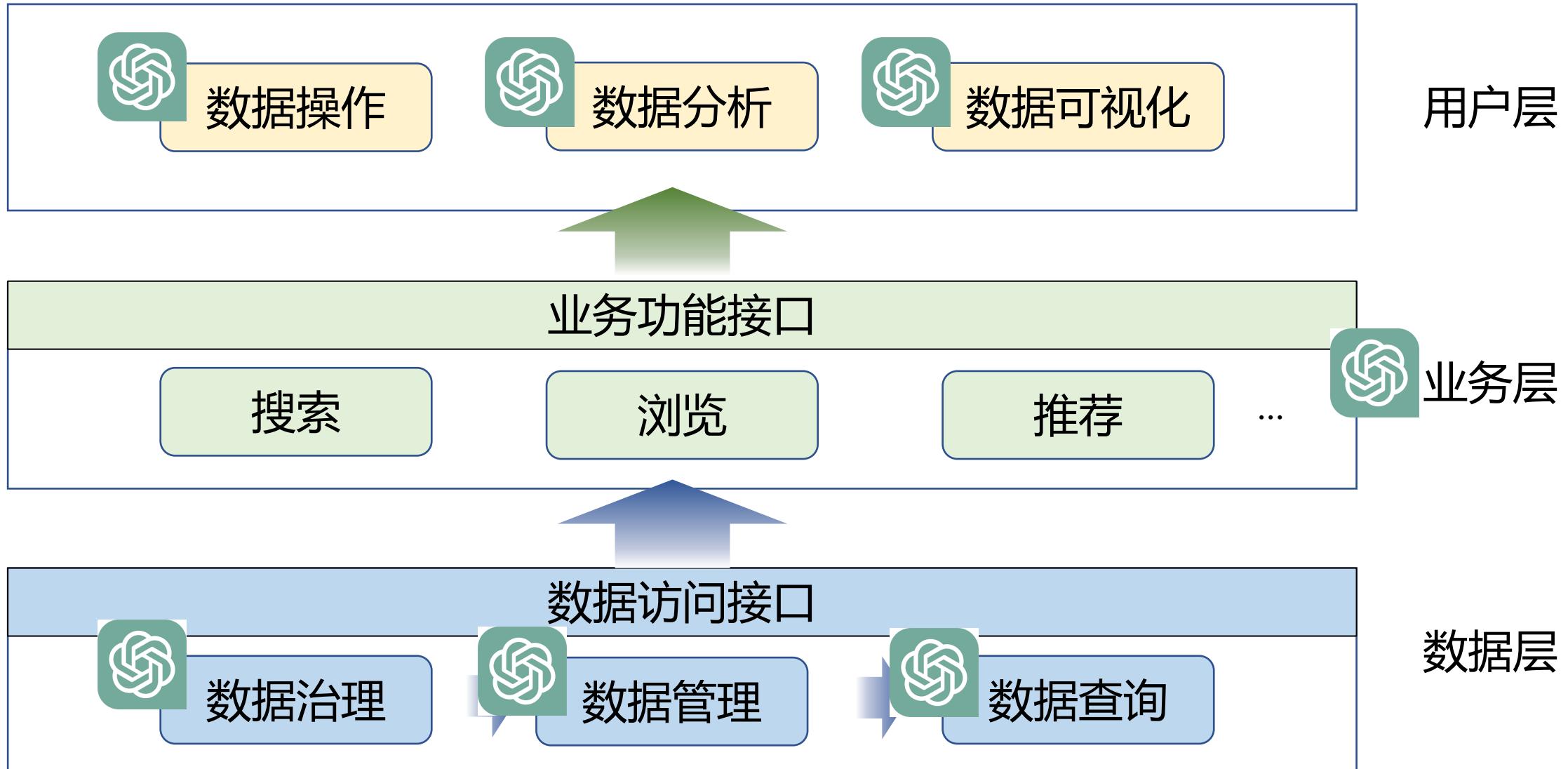
图像

- 大模型可以辅助教学资源开发、智能课堂助手、作业批改
 - 本报告探讨大模型对本科教育中提升学生工程实践能力可能发挥的作用

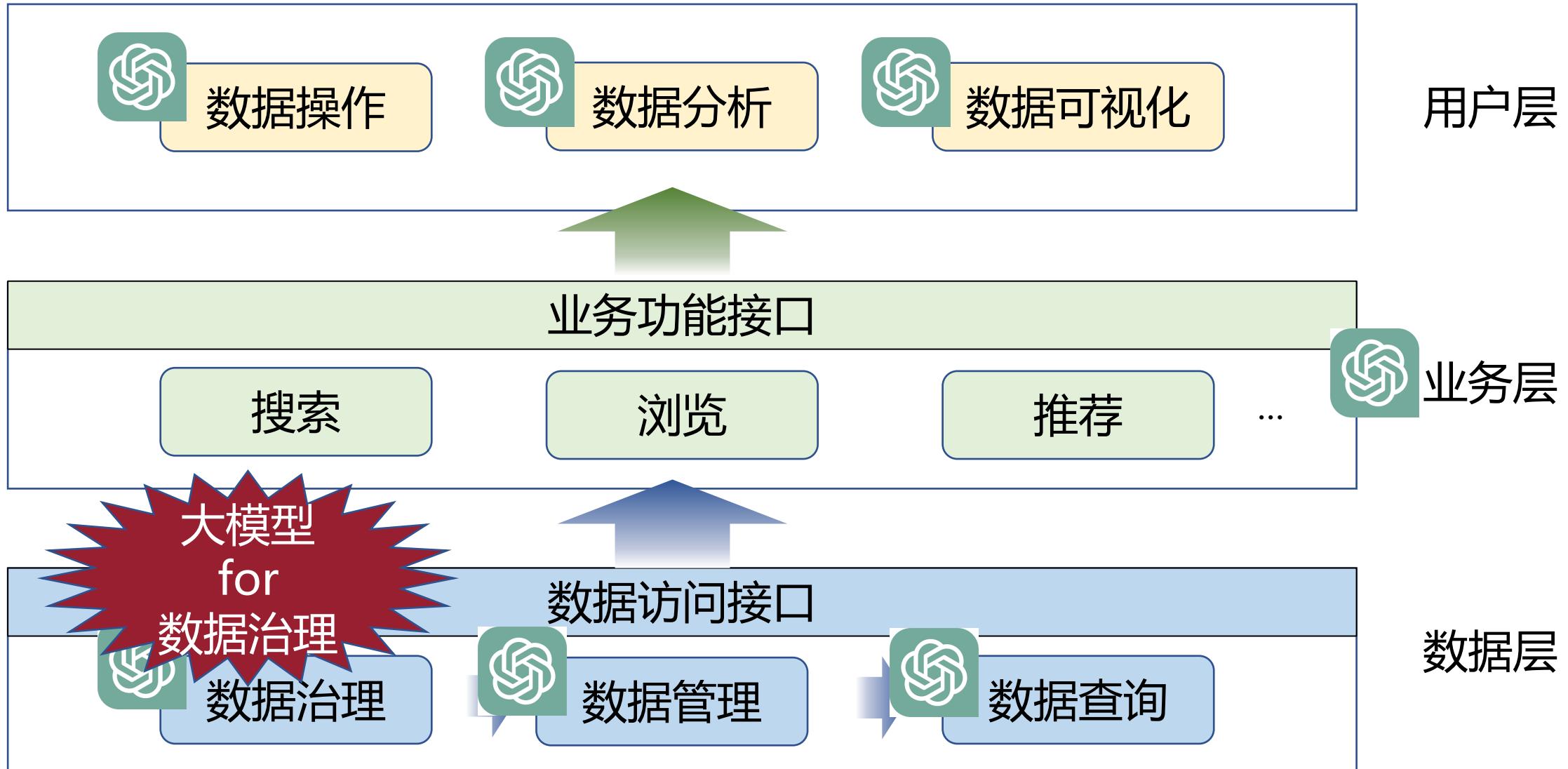
工程开发流程



工程开发流程

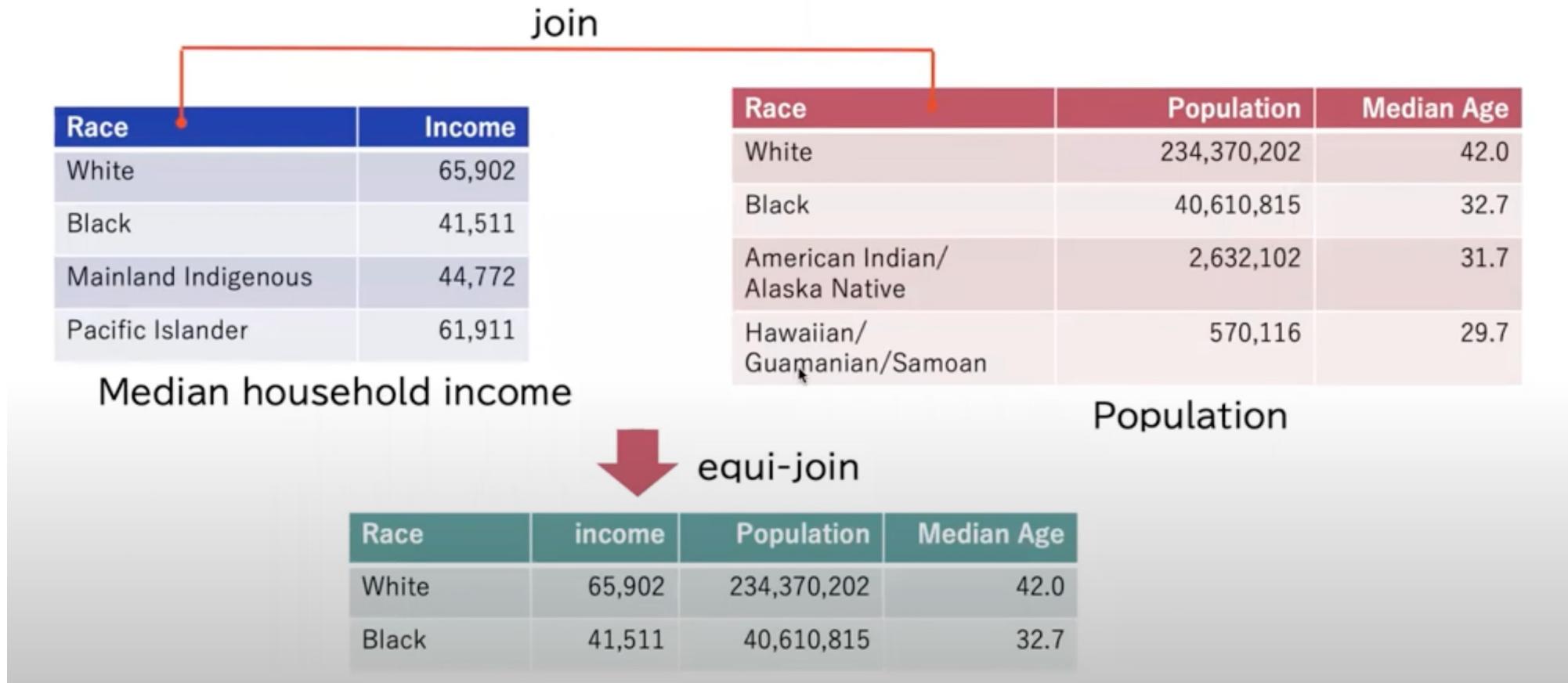


工程开发流程



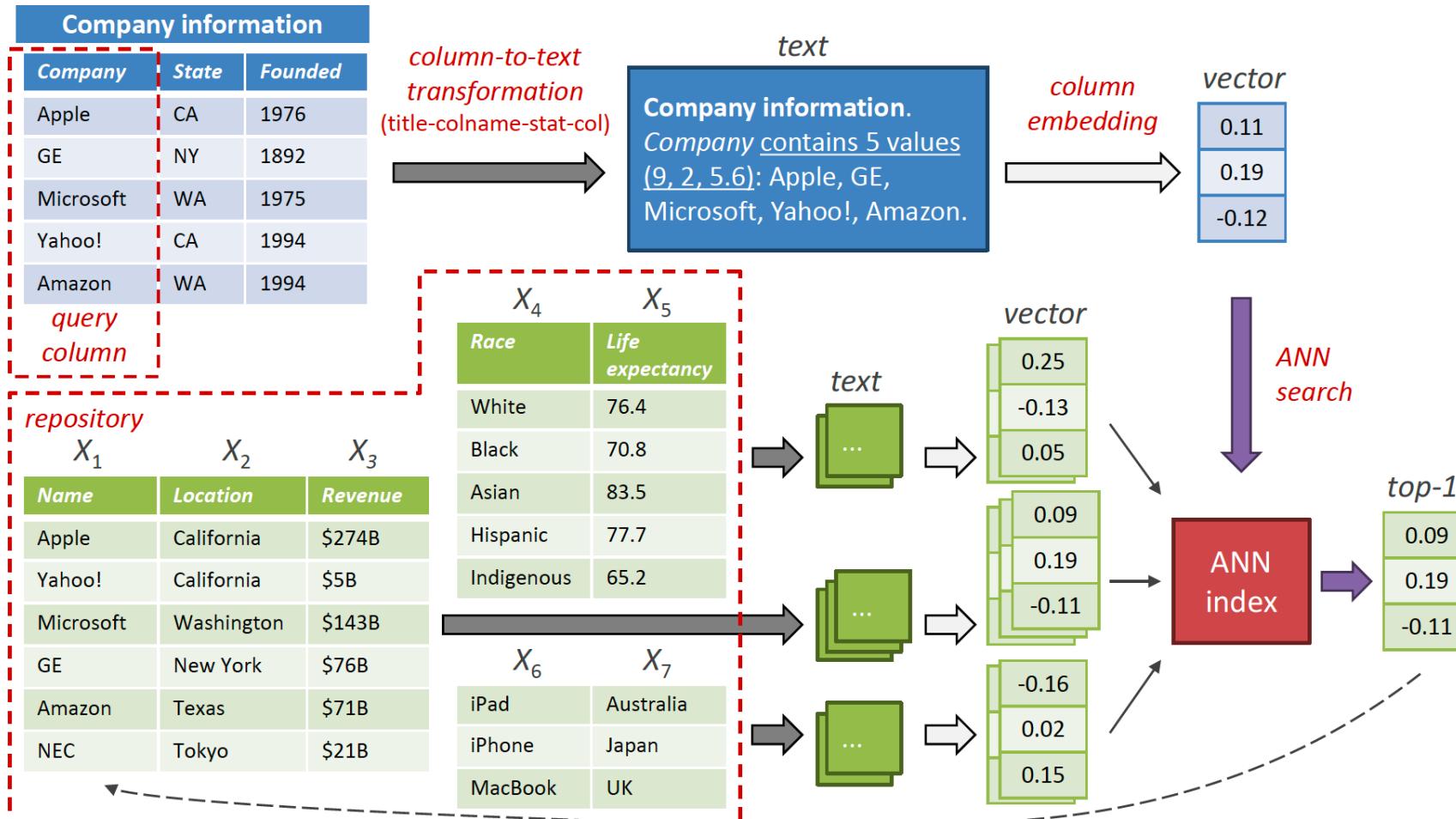
表格数据链接

任务: 多个关系型表格自动链接



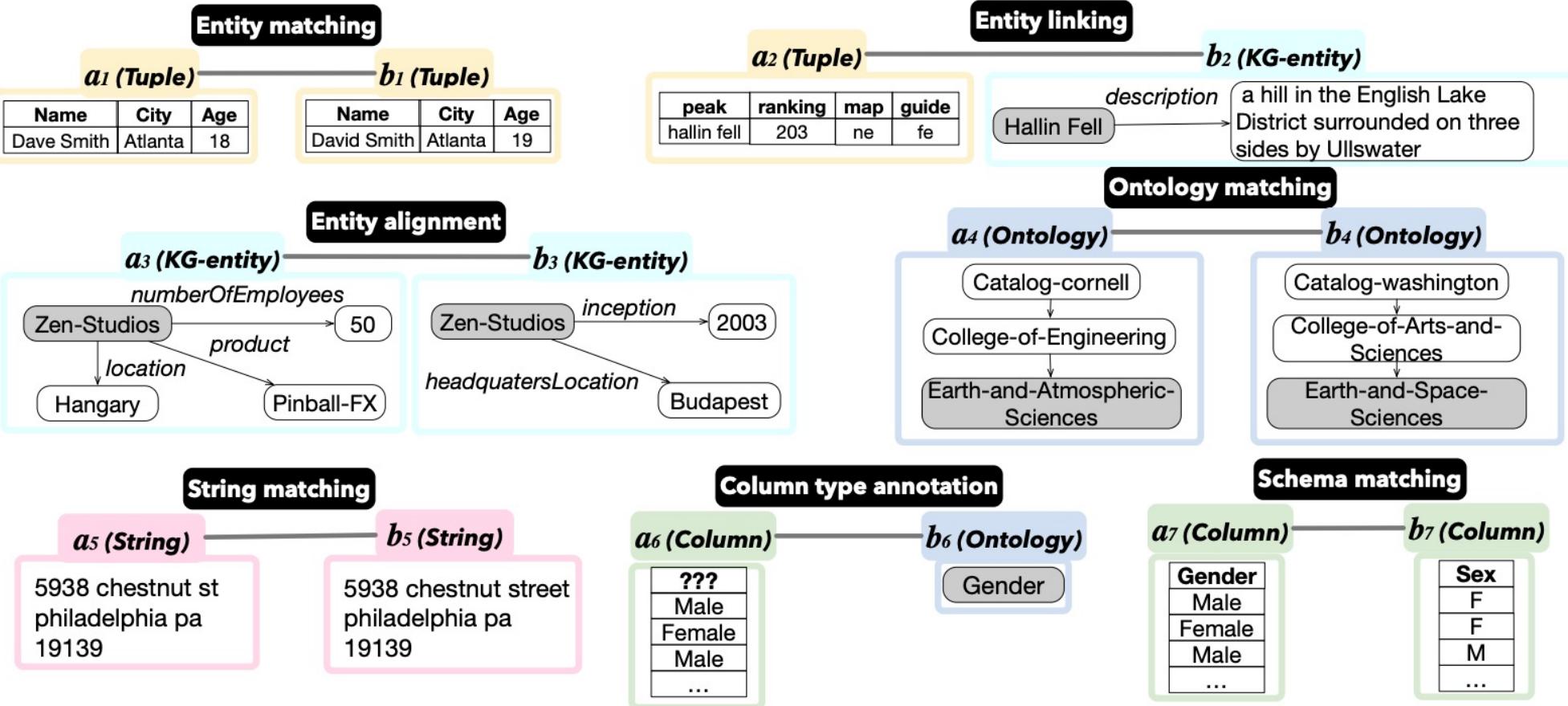
表格数据链接

解决方法: DeepJoin - 基于表格嵌入的检索



数据匹配多任务

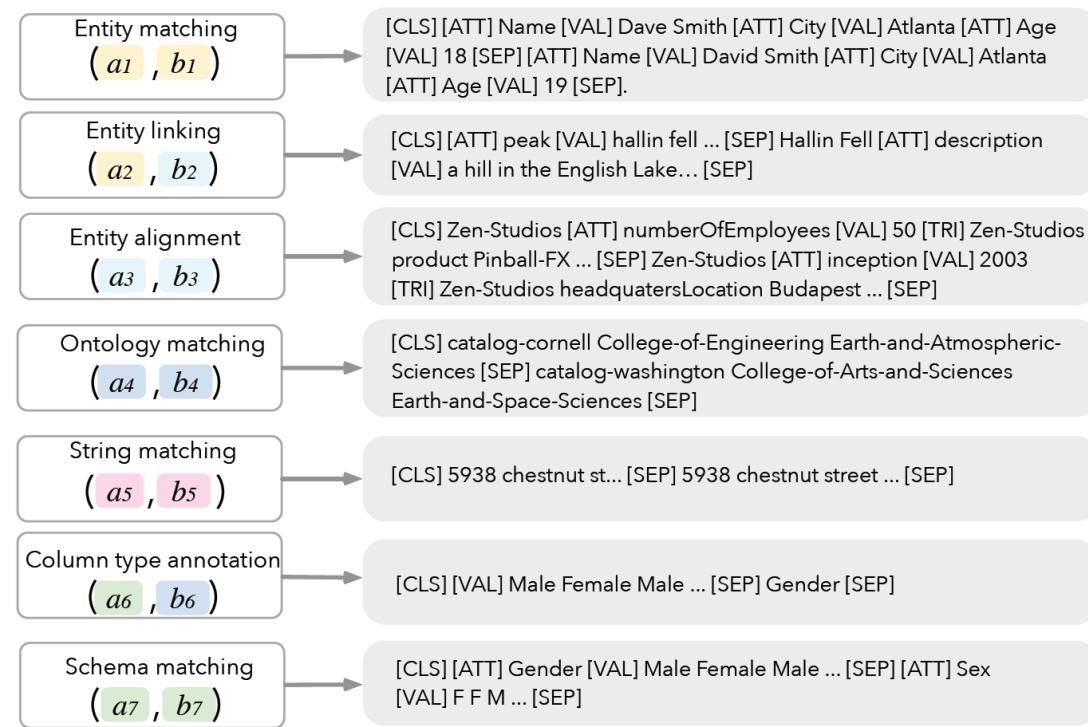
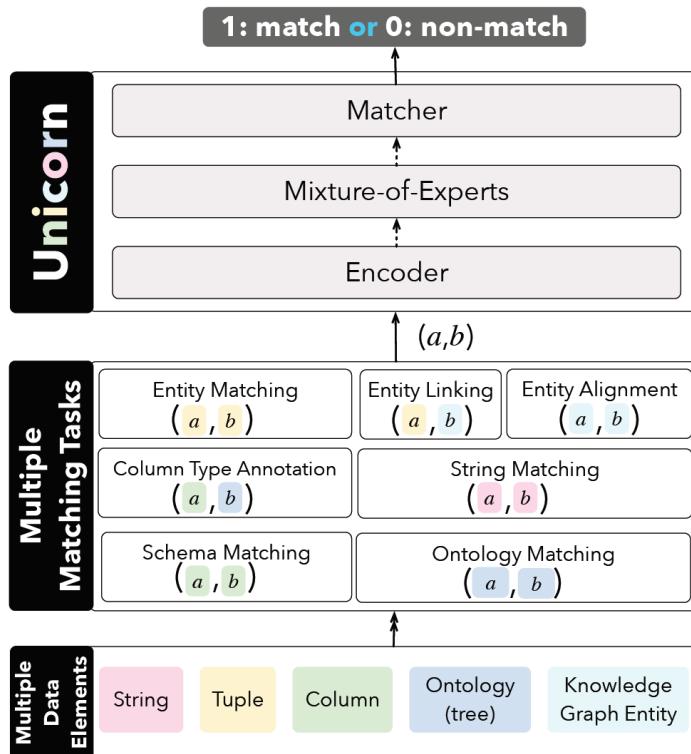
任务：判断两个元素（字符串、三元组、列、实体）是否一致



数据匹配多任务

解决方法: Unicorn 是一个用于数据集成领域**数据匹配任务的统一模型**

- Unicorn 由统一的编码器, 专家混合层, 匹配器组成
- 编码器将输入的数据对(a, b)编码为统一的表示向量。
- 专家混合层融合不同任务的专家知识, 匹配器判断(a, b)是否匹配



■ 将数据匹配任务中的数据对序列化为统一的文本序列

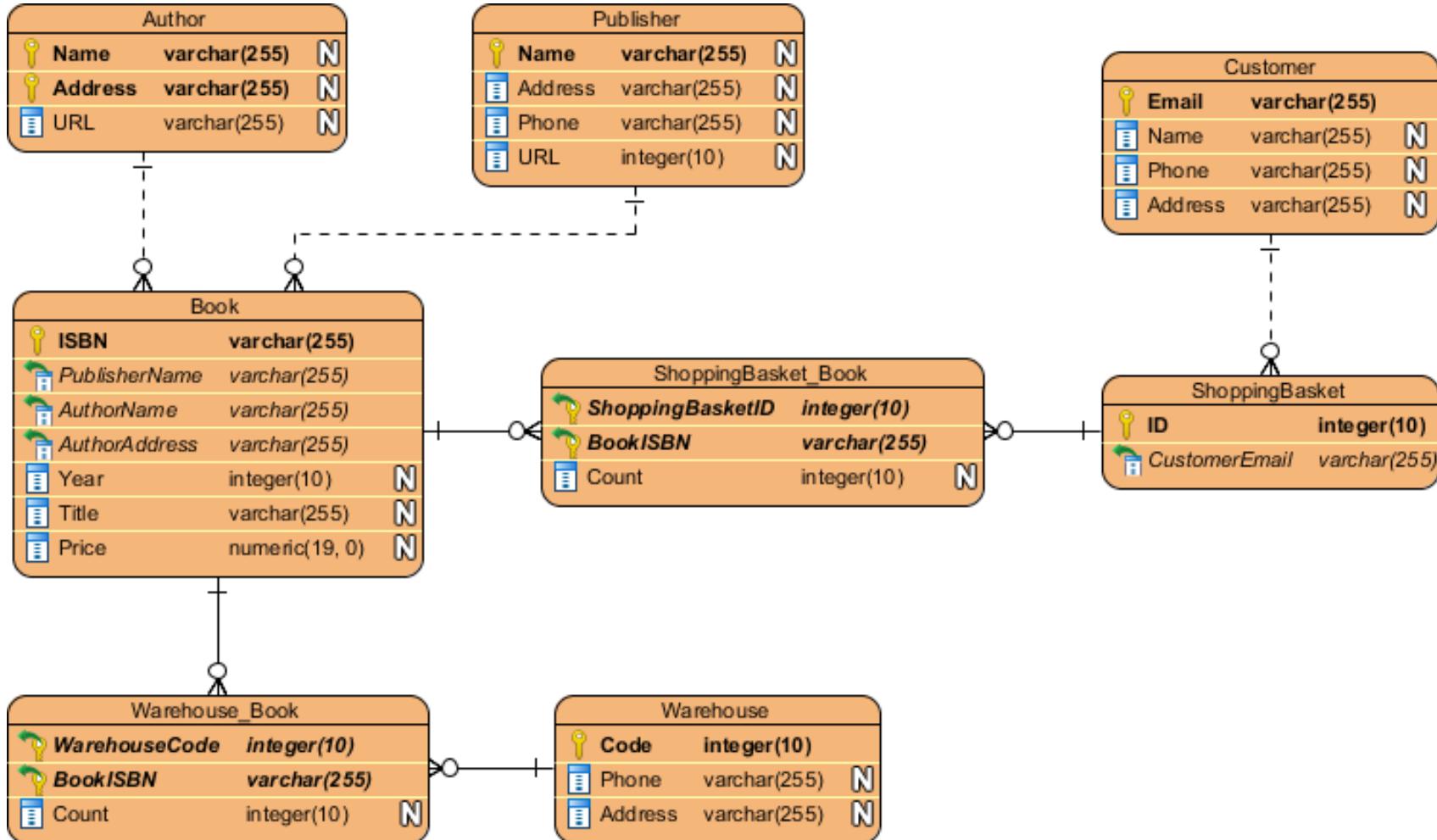
数据匹配多任务

实验结果：

Type	Task	Metric	Unicorn w/o MoE	Unicorn	Unicorn ++	Previous SOTA (Paper)
EM	Walmart-Amazon	F1	85.12	86.89	86.93	86.76 (Ditto [30])
	DBLP-Scholar	F1	95.38	95.64	96.22	95.6 (Ditto [30])
	Fodors-Zagats	F1	97.78	100	97.67	100 (Ditto [30])
	iTunes-Amazon	F1	94.74	96.43	98.18	97.06 (Ditto [30])
	Beer	F1	90.32	90.32	87.5	94.37 (Ditto [30])
CTA	Efthymiou	Acc.	98.08	98.42	98.44	90.4 (TURL [10])
	T2D	Acc.	98.81	99.14	99.21	96.6 (HNN+P2Vec [5])
	Limaye	Acc.	96.11	96.75	97.32	96.8 (HNN+P2Vec [5])
EL	T2D	F1	79.96	91.96	92.25	85 (Hybrid I [20])
	Limaye	F1	83.12	86.78	87.9	82 (Hybrid II [20])
StM	Address	F1	97.81	98.68	99.47	99.91 (Falcon [39])
	Names	F1	86.12	91.19	96.8	95.72 (Falcon [39])
	Researchers	F1	96.59	97.66	97.93	97.81 (Falcon [39])
	Product	F1	84.61	82.9	86.06	67.18 (Falcon [39])
	Citation	F1	96.34	96.27	96.64	90.98 (Falcon [39])
ScM	FabricatedDatasets	Recall	81.19	89.6	89.35	81 (Valentine [27])
	DeepMDatasets	Recall	66.67	96.3	96.3	100 (Valentine [27])
OM	Cornell-Washington	Acc.	90.64	92.34	90.21	80 (GLUE [15])
EA	SRPRS: DBP-YG	Hits@1	99.46	99.67	99.49	100 (BERT-INT [46])
	SRPRS: DBP-WD	Hits@1	97.11	97.22	97.28	99.6 (BERT-INT [46])
AVG			90.8	94.21	94.56	91.84
Model Size			139M	147M	147M	995.5M

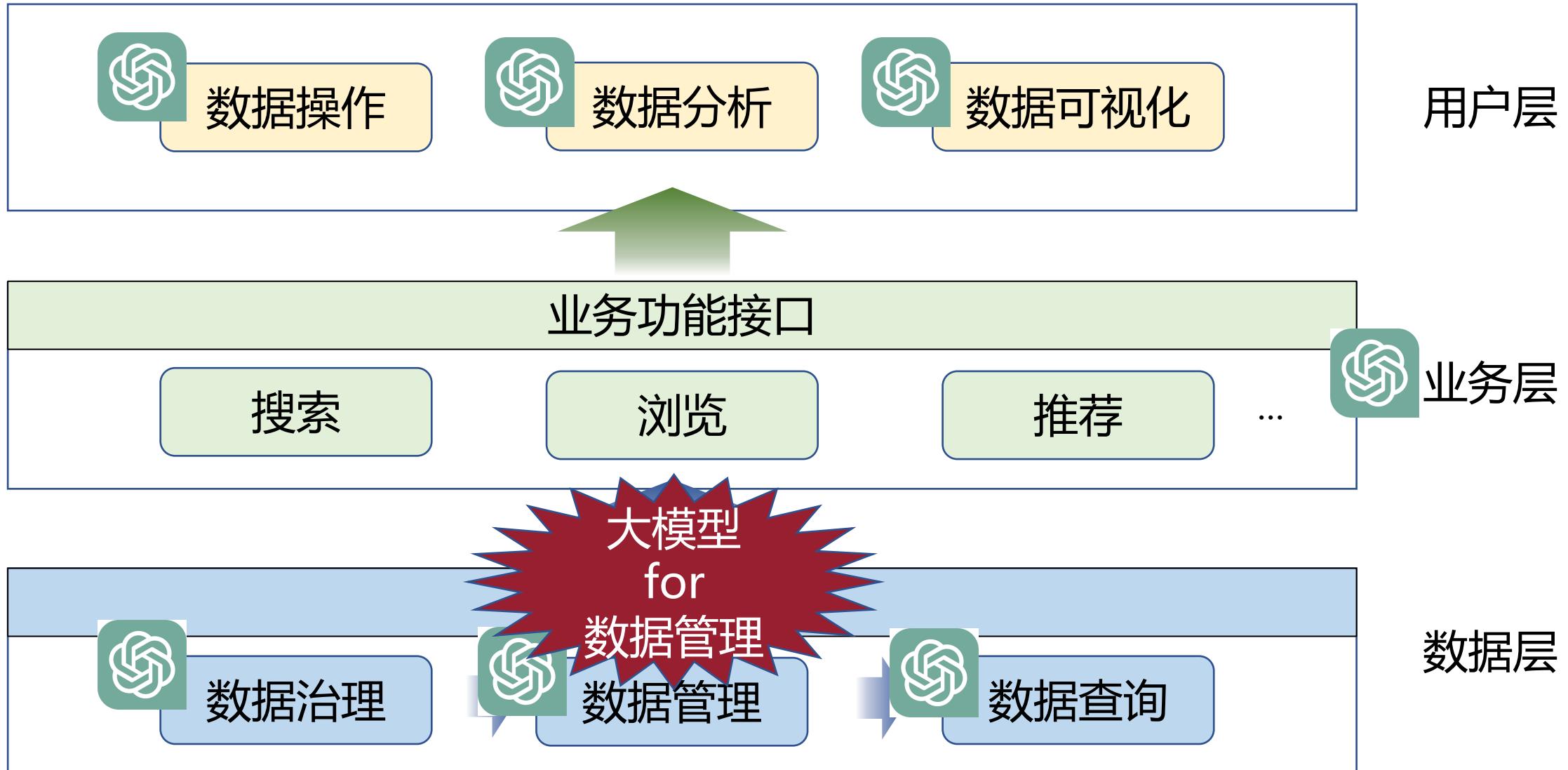
■ Unicorn与每个任务特定方法的效果持平，在多个任务上甚至优于特定方法

工程能力培养：辅助学习实体关系（ER）图



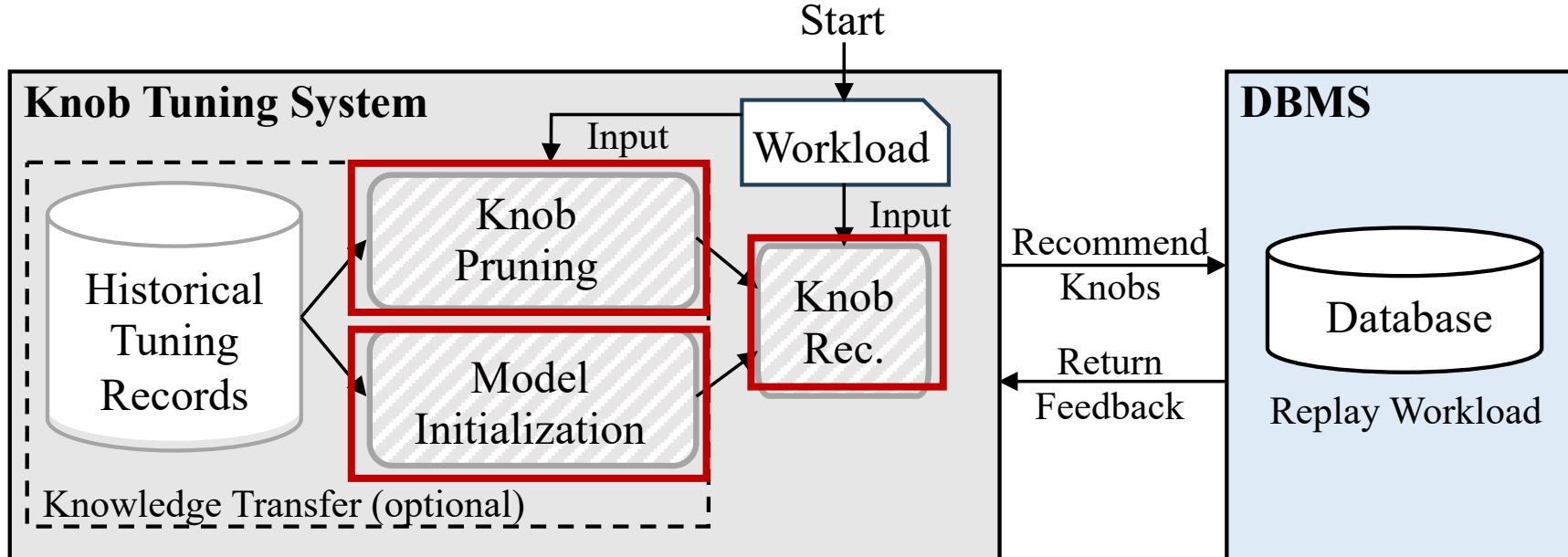
- ER图的作用
 - 数据库设计
 - 数据库调试
 - 数据库创建和修补
 - 帮助收集需求
- 手动设计实体关系
- 自动检测关系合理性

工程开发流程



数据管理

任务：数据库参数调优，对负载的执行效率进行优化



调参系统包含三个关键组件：参数修剪、模型初始化、参数推荐

- **参数修剪**旨在选择关键参数，并确定它们的合理取值范围
- **模型初始化**着眼于在参数推荐方法中初始化可学习的模型，加快收敛速度
- **参数推荐**是调参系统的核心，针对具体的工作负载提供合适的参数配置建议

数据管理

解决方法：探索每个组件的大模型提示学习方法的可能性

Knob Pruning

Task Description:

Select the 10 most important knobs from the provided and give their range of values for the current tuning task in order to optimize the throughput metric.

Candidate Knobs:

```
"innodb_thread_concurrency": {  
    "max": 1000,  
    "min": 0,  
    "type": "integer",  
    "description": "Defines the maximum number of threads permitted inside of  
InnoDB."  
},  
.....
```

Workload and Database Information:

- Workload: OLTP, SYSBENCH Read-Write Mixed Model, Read-Write Ratio = 50%, threads = 32.
- Data: 13 GB data contains 50 tables and each table contains 1,000,000 rows of record.
- Database Kernel: RDS MySQL 5.7.
- Hardware: 8 vCPUs and 16 GB RAM.

Output Format:

Knobs should be formatted as follows:

```
{  
    "knob_name": {  
        "max": MAX_Value,  
        "min": Min_Value,  
        "type": "integer"  
    },  
    or  
    "knob_name": {  
        "enum_values": [  
            "value1",  
            .....  
        ],  
        "type": "enum"  
    }  
}
```

- 1.任务描述:** LLM 的目标
- 2.候选参数:** 候选参数详细信息
- 3.工作负载和数据库信息:** 工作负载、数据特征、数据库及硬件环境的关键细节
- 4.输出格式:** LLM 的响应格式，列出被选中的参数名称、取值范围和参数类型

- 1.任务描述:** 概述了 LLM 的目标。
- 2.参数优化示范:** 作为提示中的示范案例
- 3.环境:** 数据库内核和硬件信息,还有每个内部指标和可调参数的文本描述
- 4.当前工作负载信息:** 工作负载类型、读写比例和数据统计等
- 5.输出格式:** LLM 响应的格式要求
- 6.当前配置:** 参数默认值,作为优化起点
- 7.数据库反馈:** 默认配置下执行工作负载的性能和内部指标

Model Initialization / Knob Recommendation

Task Description:

Recommend optimal knob configuration based on the inner metrics and workload characteristics in order to optimize the throughput metric.

Demonstration for Knob Refinement:

```
- Current Configuration (input):  
{ "thread_cache_size": 50, ..., "innodb_log_file_size": 512M }  
- Inner Metrics (input):  
{ "lock_timeouts": 0, ..., "dml_updates": 27356 }  
- Refined Configuration (output):  
{ "thread_cache_size": 75, ..., "innodb_log_file_size": 1024M }
```

Environments:

- Database Kernel: RDS MySQL 5.7.
- Hardware: 8 vCPUs and 16 GB RAM.
- Inner Metrics: 1.load_average. In Unix computing, the system load is a measure of the amount of computational work that a computer system performs.
.....
- Knobs: 1. innodb_thread_concurrency. Controls the maximum number of concurrent threads that InnoDB can use for executing queries.
.....

Information about Current Workload:

- Workload: OLTP, SYSBENCH Read-Write Mixed Model, Read-Write Ratio = 50%, threads = 32.
- Data: 13 GB data contains 50 tables and each table contains 1,000,000 rows of record.

Output Format:

Strictly utilize the aforementioned knobs, ensuring that the generated configuration are formatted as follows:

```
{ "knob_name": knob_value, ..... }
```

Current Configuration:

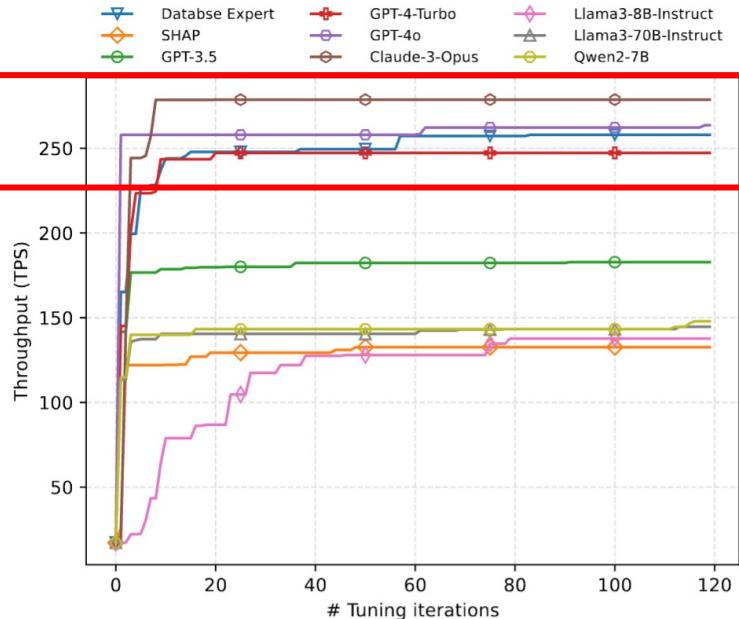
```
{ "tmp_table_size": 16777216, ..... }
```

Database Feedback:

- Throughput : 100
- Inner Metrics: lock_timeouts = 0, ..., dml_updates = 22692

数据管理

实验结果：



Type	Model	ODP	PE	TES	Speedup
Traditional Method	VBO	154.73	0%	316	0%
	VBO + Mapping	154.37	-0.23%	279	11.71%
	RGPE + Model Ensemble	158.02	0.42%	215	42.67%
	DS-DDPG + Pre-training	162.15	33.10%	313	-216.16%
Closed Source LLM	VBO + GPT-3.5	127.68	-17.48%	176	44.30%
	VBO + GPT-4-Turbo	152.01	-1.93%	84	73.41%
	VBO + GPT-4o	126.65	-18.29%	90	71.51%
	VBO + Claude-3-Opus	126.09	-18.65%	2	99.37%
Open Source LLM	VBO + Llama3-8B-Instruct	153.16	-1.19%	90	71.51%
	VBO + Llama3-70B-Instruct	154.68	-0.03%	90	71.51%
	VBO + Qwen2-7B	153.58	-0.74%	100	68.35%

Type	Method	IR TPS	ODP	TES
Traditional Method	Default	-	17.45	-
	DDPG	-	120.71	99
	SMAC	-	157.25	375
	VBO	-	155.10	316
Closed Source LLM	GPT-3.5	16.38	116.62	24
	GPT-4-Turbo	145.06	155.30	9
	GPT-4o	117.96	126.05	13
	Claude-3-Opus	26.70	148.51	23
Open Source LLM	Llama3-8B-Instruct	20.90	125.88	25
	LlaMa3-70B-Instruct	28.84	145.12	3
	Qwen2-7B	143.58	154.94	14

参数修剪：

Claude-3-Opus, GPT-4o, GPT-4-Turbo
超越数据库管理员

模型初始化：

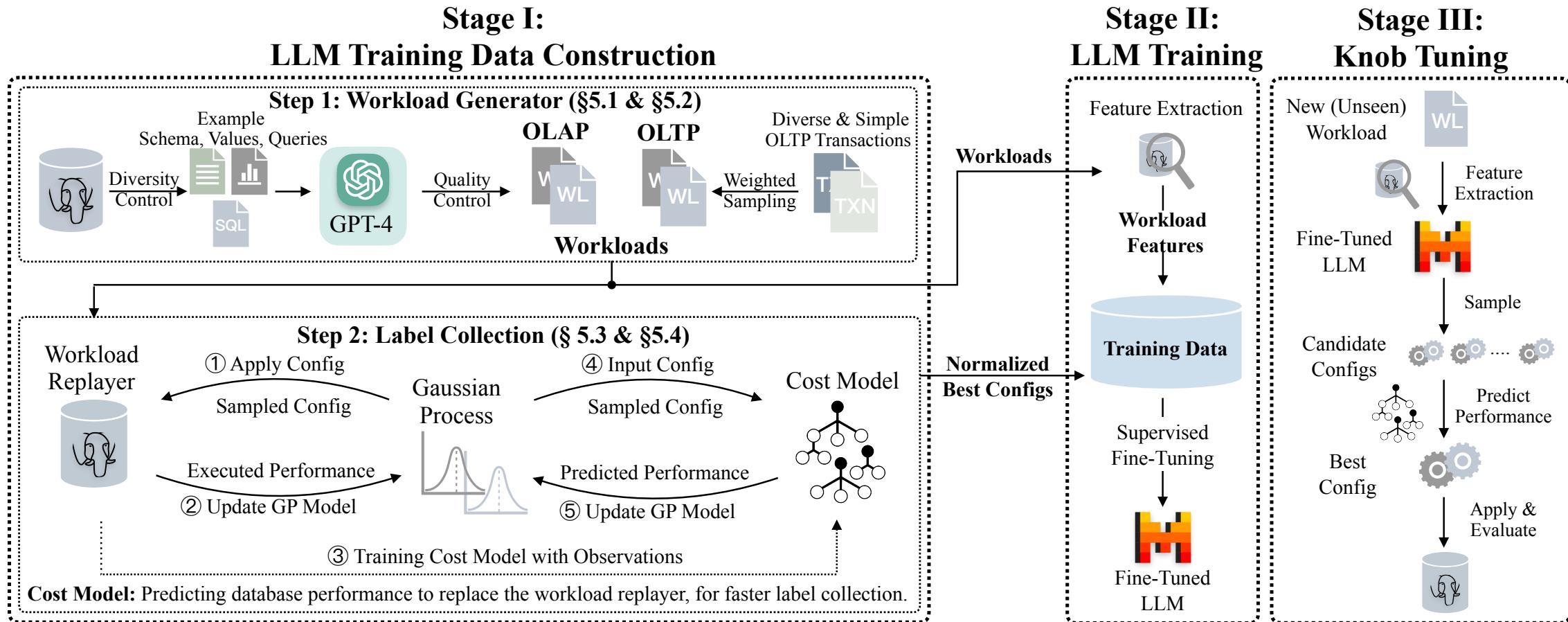
语言模型通过生成初始配置来大幅
加快贝叶斯优化方法的收敛能力

参数推荐：

语言模型能够在大幅减少的迭代
次数内找到优异的参数配置

数据管理

解决方法：参数调优的大模型微调方法



数据管理

实验结果

Methods	TPC-H [67]		JOB [35]		SSB		SSB-flat		TPC-DS	
	$\Delta (\%) \uparrow$	$T (\text{min}) \downarrow$								
HEBO	71.5/-	1381.7/-	51.8/-	766.4/-	41.7/-	732.4/-	46.3/-	823.8/-	39.2/-	749.9/-
+ Cost Model	68.8/58.2	91.2/99.8	49.2/33.6	82.3/91.5	39.2/34.6	83.8/89.3	43.1/29.2	84.5/92.4	37.6/29.2	88.9/98.7
+ Workload Mapping	72.5/70.2	328.7/739.7	51.3/50.7	308.2/428.2	42.3/42.0	280.3/439.4	46.6/44.3	266.1/547.6	40.9/40.4	254.8/563.1
+ Model Ensemble	72.3/ 71.5	234.6/662.6	52.1/ 51.9	241.4/491.0	43.5/42.2	199.0/345.2	47.3/46.9	213.7/490.2	39.6/ 40.0	321.8/462.4
SMAC	69.8/-	1182.5/-	50.0/-	724.7/-	40.0/-	634.2/-	43.9/-	794.3/-	38.7/-	712.9/-
+ Cost Model	67.3/43.1	67.3/78.6	48.8/29.6	50.2/ <u>67.8</u>	37.6/22.6	49.5/ <u>45.8</u>	41.2/33.8	48.9/55.1	37.0/29.8	<u>43.5/59.8</u>
+ Workload Mapping	70.4/67.7	374.4/896.8	52.3/51.5	283.2/567.3	40.9/40.1	289.8/413.8	46.2/45.9	315.2/515.8	39.7/38.6	223.4/632.4
+ Model Ensemble	70.3/70.5	289.3/658.4	51.9/ <u>51.7</u>	249.9/493.2	43.4/43.3	179.6/323.1	47.1/46.7	284.3/563.3	39.9/39.1	213.6/549.8
CDBTune	68.8/-	965.4/-	48.4/-	489.5/-	38.8/-	320.3/-	45.5/-	723.4/-	36.9/-	698.4/-
+ Cost Model	66.9/38.9	<u>63.2/70.9</u>	47.3/29.8	<u>49.6/81.4</u>	38.0/23.1	<u>48.8/54.9</u>	40.5/22.0	<u>42.1/52.9</u>	36.6/19.6	53.2/76.8
+ MP (w/ online tuning)	71.2/68.9	569.4/698.3	49.2/49.2	198.4/378.4	42.2/37.7	235.1/352.9	<u>47.3/46.1</u>	312.4/536.9	38.6/37.0	258.3/589.8
+ MP (w/o online tuning)	47.8/37.5	634.7/702.9	42.3/32.5	178.9/267.9	29.5/18.1	184.9/213.4	42.3/30.1	72.1/378.1	39.4/22.5	103.9/409.6
OpAdviser	73.4/67.3	579.3/638.9	52.2/49.1	328.8/391.0	43.9/36.3	219.6/380.2	47.1/45.2	412.5/456.3	<u>40.1/39.1</u>	283.1/312.5
DB-BERT	70.9/-	523.1/-	51.3/-	299.0/-	40.0/-	202.8/-	44.3/-	438.6/-	38.2/-	232.4/-
GPTuner	72.0/-	301.2/-	51.3/-	213.8/-	44.9/-	178.3/-	47.3/-	233.5/-	39.2/-	193.4/-
E2ETUNE	<u>72.5/71.0</u>	19.8/20.9	52.8/50.9	12.3/15.4	<u>44.0/42.5</u>	9.8/8.9	47.4/46.1	13.9/9.2	<u>40.1/39.9</u>	9.9/11.8

- 微调的参数调优大模型调优速度最快，且能实现与传统调优方法相媲美的性能改善效果
- 超越经典的贝叶斯调优方法、基于知识迁移的模型初始化、利用大模型的参数修剪方法

工程能力培养：辅助学习数据库调优经验

LLM Output

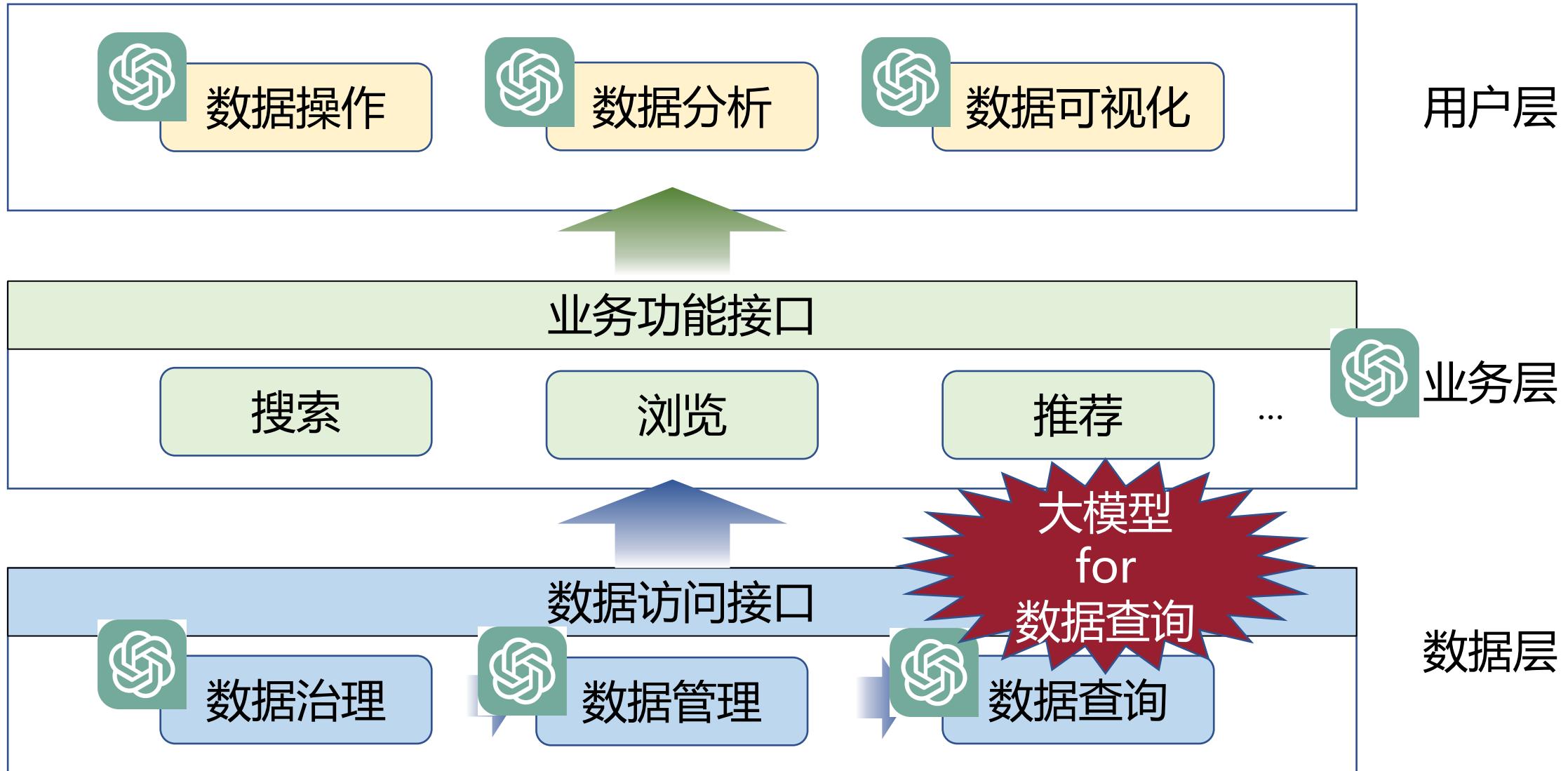
Recommendations for Further Tuning

- 1. Increase Buffer Pool Size:**
 - Current: 13GB
 - Recommendation: Increase to 14GB if the system has sufficient memory available to further reduce disk reads.
- 2. Optimize Read-Ahead:**
 - Metrics: `buffer_pool_read_ahead(0)`
 - Adjust `innodb_read_ahead_threshold` to 24 to improve sequential read performance.
- 3. Increase Write I/O Threads:**
 - Current: 16
 - Metrics: `os_data_writes(2104005120)`, `innodb_dblwr_writes(628)`
 - Recommendation: Increase to 32 to improve write throughput.

.....

- 语言模型能够以“思维链”的方式,为每个参数调整提供详细的理由和考虑因素。
- 与传统的“黑箱”方法相比,语言模型在参数调优中展现出了更好的可解释性。这不仅增强了实际应用性,也提高了结果的可靠性。
- 通过阐述推荐的原因,语言模型还可以帮助数据库管理员做出更明智的决策,促进参数推荐过程的协作和高效。

工程开发流程

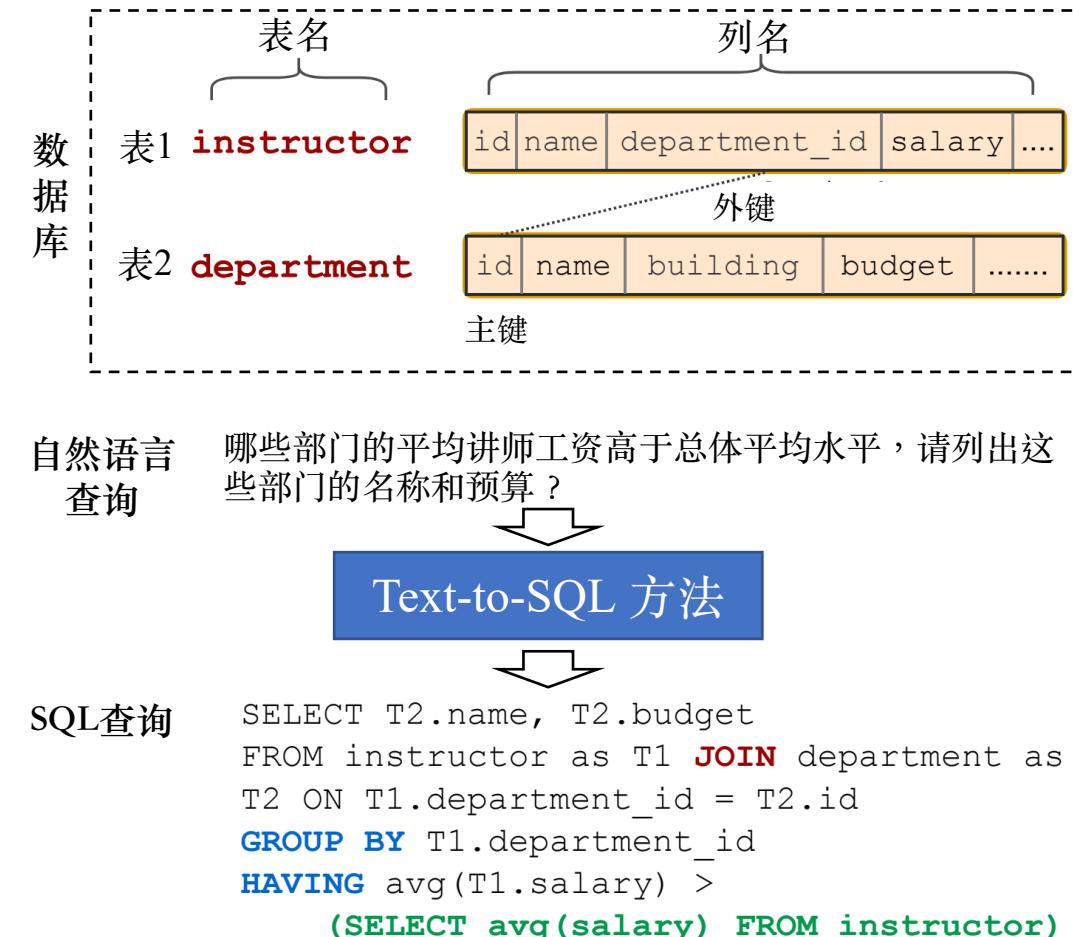


Text-to-SQL

任务：给定一个数据库，将自然语言问题转化为SQL语句

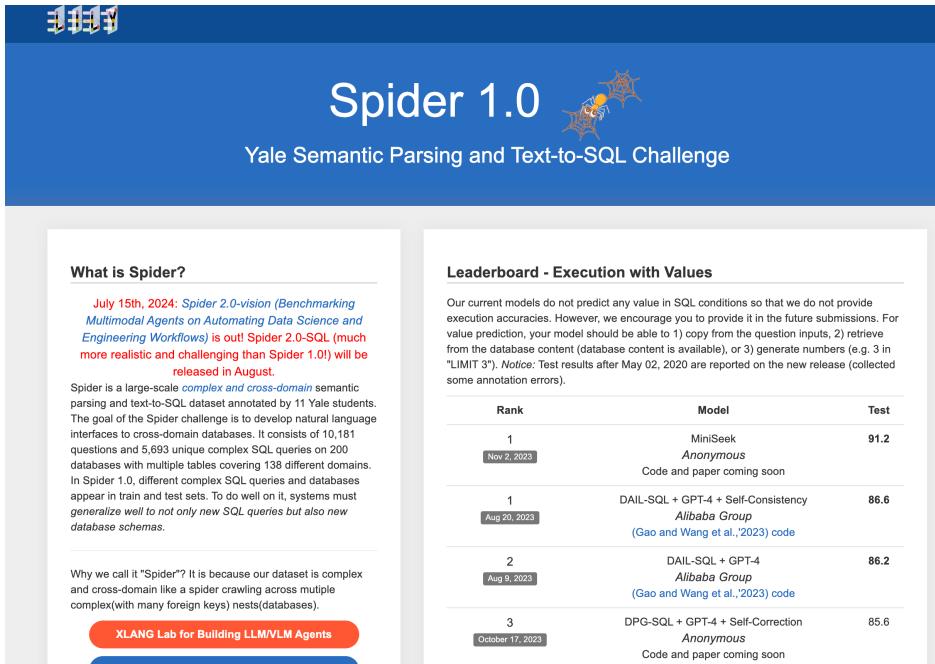
Text-to-SQL任务的核心挑战包括：

- 理解用户的自然语言查询意图，包括查询的关键词、条件、排序等要求。
- 将自然语言中提及的实体（如表名、列名）和关系（如列与列之间的关系）正确地映射到数据库的相应元素上。
- 生成复杂查询，包括多表连接、子查询、聚合操作等。



Text-to-SQL数据集

国际公认的Spider和BIRD等数据集，提供了丰富的<数据库，自然语言查询，SQL>样本，用于评估



The homepage for Spider 1.0 features a blue header with the title "Spider 1.0" and a small spider icon. Below the header, it says "Yale Semantic Parsing and Text-to-SQL Challenge". The main content area includes a "What is Spider?" section with a brief description and a "Leaderboard - Execution with Values" section showing the current ranking of models.

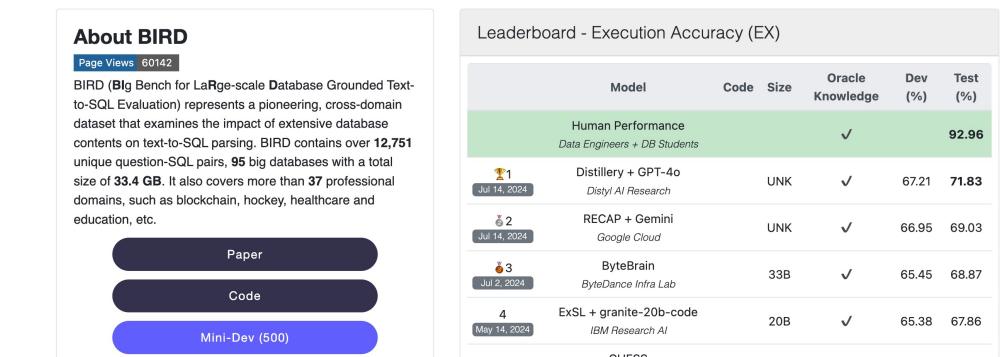
Rank	Model	Test
1	MiniSeek Anonymous Code and paper coming soon	91.2
1	DAIL-SQL + GPT-4 + Self-Consistency Alibaba Group (Gao and Wang et al., 2023) code	86.6
2	DAIL-SQL + GPT-4 Alibaba Group (Gao and Wang et al., 2023) code	86.2
3	DPG-SQL + GPT-4 + Self-Correction Anonymous Code and paper coming soon	85.6

<https://yale-lily.github.io/spider>

Spider是第一个跨领域且大规模的text2sql数据集，包含了10,181条文本到SQL数据对和200个独立的数据库，内容覆盖了138个不同领域。



The homepage for BIRD-SQL features a large image of a colorful cartoon bird holding a sign that says "SELECT CAST JOIN". The title "BIRD-SQL" is prominently displayed in green, with the subtitle "A Big Bench for Large-Scale Database Grounded Text-to-SQLs" below it.



The BIRD-SQL leaderboard shows execution accuracy across five models. The top model is Human Performance (Data Engineers + DB Students) at 92.96%. Other models include Distillery + GPT-4o, RECAP + Gemini, ByteBrain, and ExSL + granite-20b-code.

Model	Code	Size	Oracle Knowledge	Dev (%)	Test (%)
Human Performance Data Engineers + DB Students			✓	92.96	
Distillery + GPT-4o Distil AI Research	UNK		✓	67.21	71.83
RECAP + Gemini Google Cloud	UNK		✓	66.95	69.03
ByteBrain ByteDance Infra Lab	33B		✓	65.45	68.87
ExSL + granite-20b-code IBM Research AI	20B		✓	65.38	67.86

<https://bird-bench.github.io>

BIRD引入三个额外的挑战，即处理大规模和混乱的数据值、外部知识推理和优化生成SQL的执行效率。

Text-to-SQL评测指标

评价Text-to-SQL方法的性能主要依赖两个指标：精确匹配率（EM）和执行准确率（EX）

- 1. 精确匹配率（Exact Set Match, EM）**：衡量生成的SQL与标准SQL在语法树层面的一致性，其局限在于无法识别语义等价但结构不同的SQL查询，可能导致假负例。
- 2. 执行准确率（Execution Accuracy, EX）**：通过比较生成SQL与标准SQL在数据库上执行结果的一致性来评估。可能会将语义不同但偶然产生相同结果的SQL视为正确，导致假正例。

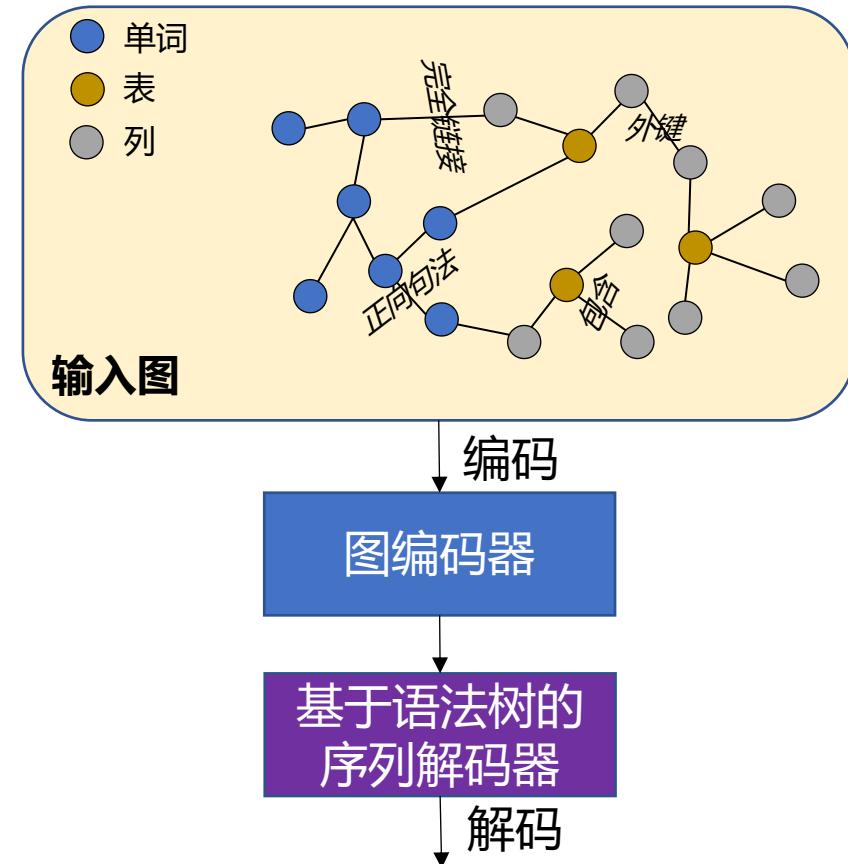
为全面评估Text-to-SQL模型性能，可综合使用EM和EX两个指标，并引入以下补充指标：

- 1. 测试组件准确率（Test Suit Execution Accuracy）**：针对EX指标的假正例问题，TS通过比较生成SQL与标准SQL在不同数据库实例上的执行结果，以判断其语义等价性。
- 2. 有效效率得分（Valid Efficiency Score）**：衡量生成SQL的执行效率，以评价模型输出的性能。

这些补充指标有助于更精确地评估模型（学生）的SQL撰写能力

基于编码器-解码器模型的训练方法

- **编码器：**将用户的自然语言查询和数据库信息整合成**图结构**，使用图编码器为图中的每个节点（包括数据库的表、列和自然语言查询中的单词）编码，捕获结构信息。
- **解码器：**生成相应的SQL查询。为了确保生成的SQL查询不仅语义上正确而且遵循SQL的语法规则，采用受语法限制的序列解码器，考虑SQL语法的约束，如正确的关键字顺序、括号匹配等。



```
SELECT T2.name, T2.budget  
FROM instructor as T1 JOIN department as  
T2 ON T1.department_id = T2.id  
GROUP BY T1.department_id  
HAVING avg(T1.salary) >  
(SELECT avg(salary) FROM instructor)
```

基于生成式语言模型的微调方法

- 随着T5、GPT、BART的出现，Text-to-SQL被视为序列到序列的转换问题，在包含自然语言查询和数据库信息的输入序列上进行微调，直接生成SQL查询。

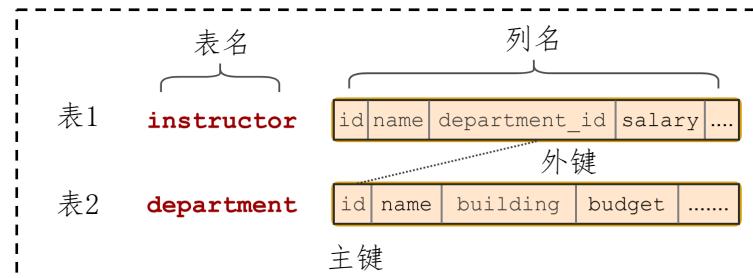
自然语言查询：

哪些部门的平均讲师工资高于总体平均水平，请列出这些部门的名称和预算？

自然语言查询 +
数据库信息
(输入序列)

SQL查询 (输出序列)

数据库信息：



将数据库信息序列化为文本

instructor: id, name, department_id, salary ... | department: id, name...

[哪些部门的平均讲师... | instructor: id, name, department_id ...]

输入

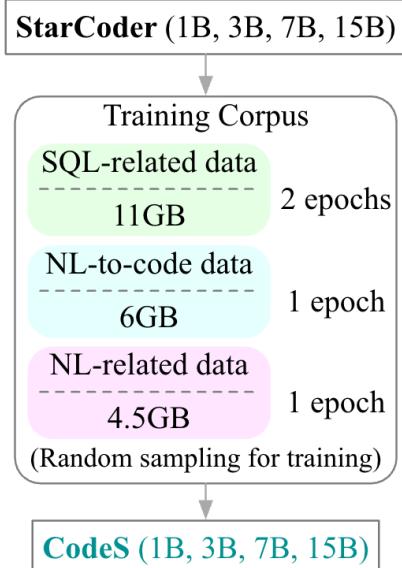
序列到序列的生成式语言模型 (如T5和BART)

输出

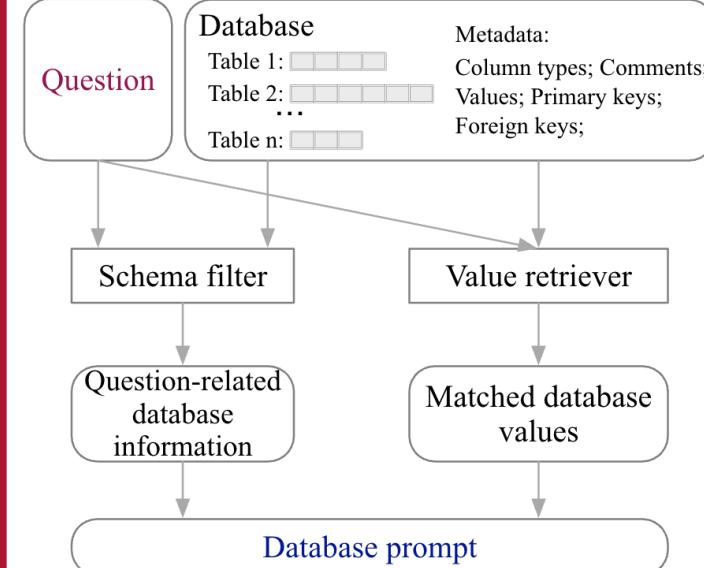
```
SELECT T2.name, T2.budget
FROM instructor as T1 JOIN department as
T2 ON T1.department_id = T2.id
GROUP BY T1.department_id
HAVING avg(T1.salary) >
(SELECT avg(salary) FROM instructor)
```

Codes: 大模型微调方法

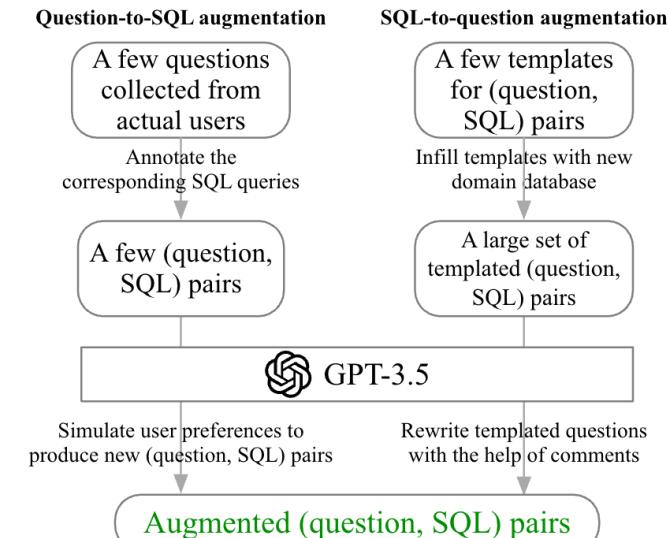
(a) Incremental pre-training



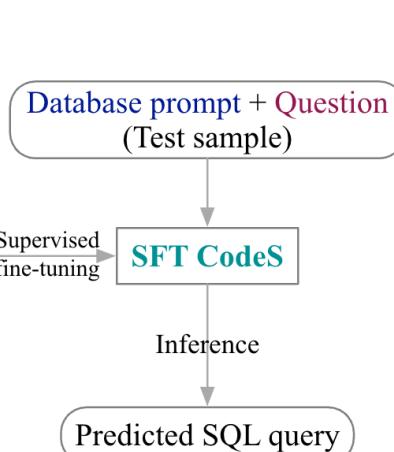
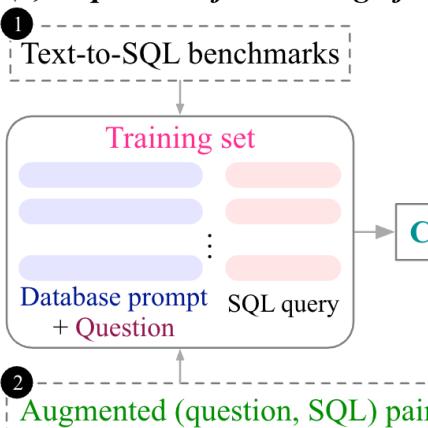
(b) Database prompt construction



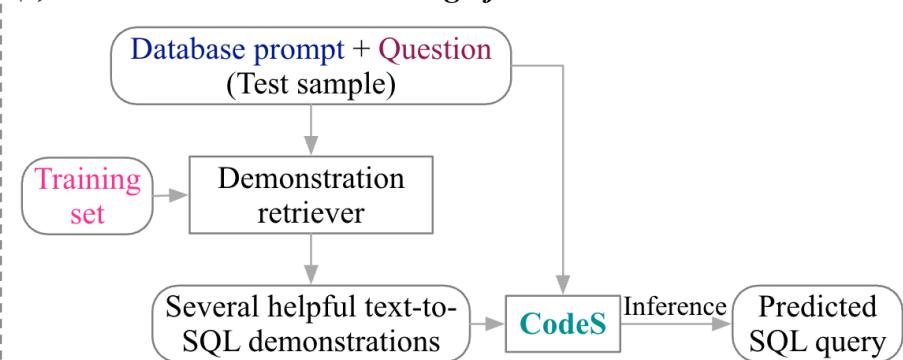
(c) Bi-directional augmentation for new domain adaptation



(d) Supervised fine-tuning of CodeS



(e) Few-shot in-context learning of CodeS



基于大语言模型的提示方法

- 利用如GPT-4等大语言模型的zero-shot和few-shot上下文学习能力，通过设计合适的提示词来引导模型完成任务，无需模型微调。
- 通常将任务分解为多个子任务，每个子任务使用特定提示词来解决，最终模型整合这些子任务的结果以生成SQL查询。

1-shot示例：提供了一个完整的text-to-SQL示例，包括<数据库信息，自然语言查询，SQL查询>的三元组，以指导模型了解任务格式和预期输出。

测试样例：给出了测试用的数据库信息和自然语言查询，但不包括对应的SQL查询。

指令：明确告诉模型需要执行的任务

指令

1-shot

测试样例

这是一个将文本转换为SQL语句的任务。我们将首先给出数据库模式，然后以自然语言形式提出问题。en ask a
要求您生成对应的SQL语句。

下面是一个示例，将文本转换为SQL：

[Schema (values)]: | farm | city : city_id , official_name , status , area_km_2 , population ,
census_ranking | farm : farm_id , year , total_horses , working_horses , total_cattle , oxen ,
bulls , cows , pigs , sheep_and_goats | farm_competition : competition_id , year , theme ,
host_city_id , hosts | competition_record : competition_id , farm_id , rank;

[Column names (type)]: city : city_id (number)| city : official_name (text)| city : status
(text)| city : area_km_2 (number)| city : population (number)| city : census_ranking
(text)| farm : farm_id (number)| farm : year (number)| farm : total_horses (number)| farm :
working_horses (number)| farm : total_cattle (number)| farm : oxen (number)| farm : bulls
(number)| farm : cows (number)| farm : pigs (number)| farm : sheep_and_goats (number)|
farm_competition : competition_id (number)| farm_competition : year (number)| farm_competition
: theme (text)| farm_competition : host_city_id (number)| farm_competition : hosts (text)
)| competition_record : competition_id (number)| competition_record : farm_id (number)|
competition_record : rank (number);

[Primary Keys]: city : city_id | farm : farm_id | farm_competition : competition_id |
competition_record : competition_id;

[Foreign Keys]: farm_competition : host_city_id equals city : city_id | competition_record :
farm 按年份升序排列的农场竞赛主题是什么？

这是要回答的测试问题：将文本转换为SQL：

[SQL]: select theme from farm_competition order by year asc;
Here is the test question to be answered: Convert text to SQL:

[Schema (values)]: | concert_singer | stadium : stadium_id , location , name , capacity
, highest , lowest , average | singer : singer_id , name , country , song_name ,
song_release_year , age , is_male | concert : concert_id , concert_name , theme , stadium_id ,
year | singer_in_concert : concert_id , singer_id;

[Column names (type)]: stadium : stadium_id (number)| stadium : location (text)| stadium : name
(text)| stadium : capacity (number)| stadium : highest (number)| stadium : lowest (number)|
stadium : average (number)| singer : singer_id (number)| singer : name (text)| singer : country
(text)| singer : song_name (text)| singer : song_release_year (text)| singer : age (number)
)| singer : is_male (others)| concert : concert_id (number)| concert : concert_name (text)|
concert : theme (text)| concert : stadium_id (text)| concert : year (text)| singer_in_concert :
concert_id (number)| singer_in_concert : singer_id (text);

[Primary Keys]: stadium : stadium_id | singer : singer_id | concert : concert_id |
singer_in_concert : singer_id (text);

[Foreign Keys]: concert : stadium_id equals stadium : stadium_id | singer_in_concert : singer_id
equals singer : singer_id | singer_in_concert : concert_id equals concert : concert_id

[Q]: How many singers do we have?;

[SQL]:

工程能力培养：辅助学习SQL

The screenshot shows a 'Text-to-SQL demo' interface. On the left, a message from 'You' asks: 'What is the most cited paper of Jing Zhang from renmin University of China?'. Below it, 'CodeS' responds with its predicted SQL query:

```
Database: Aminer_Simplified
Predicted SQL query:
SELECT paper.title FROM paper JOIN paper_authors ON
paper.id = paper_authors.paper_id JOIN author ON
paper_authors.author_id = author.id WHERE author.org =
'renmin Univ of China' AND author.name = 'Jing Zhang'
ORDER BY paper.n_citation DESC LIMIT 1;
```

On the right, a 'Select a database' dropdown menu is open, showing 'Aminer_Simplified' selected. The menu lists several CREATE TABLE statements:

```
CREATE TABLE venue(
    id TEXT,
    displayname TEXT,
    PRIMARY KEY (id)
);

CREATE TABLE affiliation(
    id TEXT,
    displayname TEXT,
    TYPE TEXT,
    url TEXT,
    PRIMARY KEY (id)
);

CREATE TABLE author(
    id TEXT,
    name TEXT,
    org TEXT,
    POSITION TEXT,
    n_pubs INTEGER,
    n_citation INTEGER,
    h_index INTEGER,
    PRIMARY KEY (id)
);

CREATE TABLE paper(
    id TEXT,
    title TEXT,
    YEAR INTEGER,
    n_citation INTEGER,
    page_start TEXT,
    page_end TEXT,
    lang TEXT,
    volume TEXT,
    doi TEXT,
    pdf TEXT,
    abstract TEXT,
    PRIMARY KEY (id)
);

CREATE TABLE venue_papers(
    venue_id TEXT,
    paper_id TEXT,
    FOREIGN KEY (venue_id) REFERENCES venue(id),
    FOREIGN KEY (paper_id) REFERENCES paper(id)
);
```

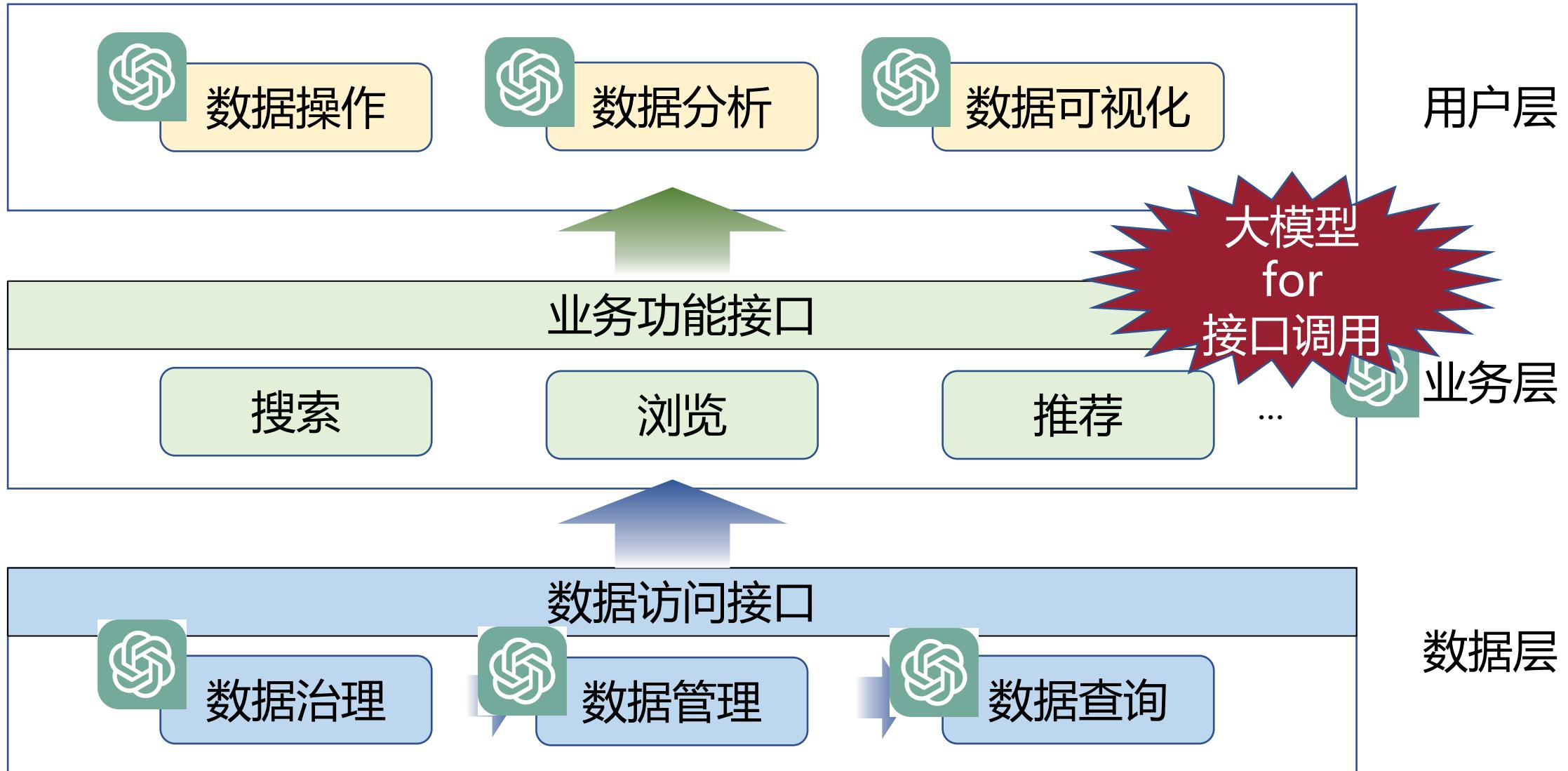
Text-to-SQL任务的应用场景广泛：

1. 商业智能 (BI) : 允许非技术背景的商业分析师通过自然语言查询企业数据库，快速获取决策支持信息。

2. 数据工程师降本增效：辅助数据工程师撰写SQL。

3. 教育：帮助学习SQL的学生验证他们的查询是否正确，或者提供查询示例；帮助教师出题。

工程开发流程



例子：学术挖掘与检索系统AMiner的接口调用

业务问题: 纽约大学Yann LeCun引用最高的论文的引用数是多少?

数据访问接口

ID	API name	Type	Parameter(s)	Return
1	searchPerson	fuzzy	name, organization, interest	[person_id, name, num_citation, interest, num_pubs, organization]
2	searchPublication	fuzzy	publication_info	[pub_id, title, year]
3	getCoauthors	exact	person_id	[id, name, relation]
4	getPersonInterest	exact	person_id	list of interests
5	getPublication	exact	pub_id	abstract, author_list, num_citation
6	getPersonBasicInfo	exact	pub_id	person_id, name, gender, organization, position, bio, education_experience, email
7	getPersonPubs	exact	person_id	[authors_name_list, pub_id, title, num_citation, year]

Yann LeCun, NYU

► searchPerson

► [{ person_id: ec0f***jsk,
person_name: Yann LeCun, ... }]

ec0f***jsk

► getPersonPubs

► [{ pub_id : al4k***8fa, ... },
{ pub_id : 79pa***rjk, ... },
{ pub_id : q2f4***n3c, ... }...]

al4k***8fa

79pa***rjk

q2f4***n3c

► getPublication

► [{ title: Efficient Backprop, citation: 7145},
{ title: DeepLearning, citation: 79904},
{ title: The MNIST Database, citation: 7592}...]

大模型接口调用

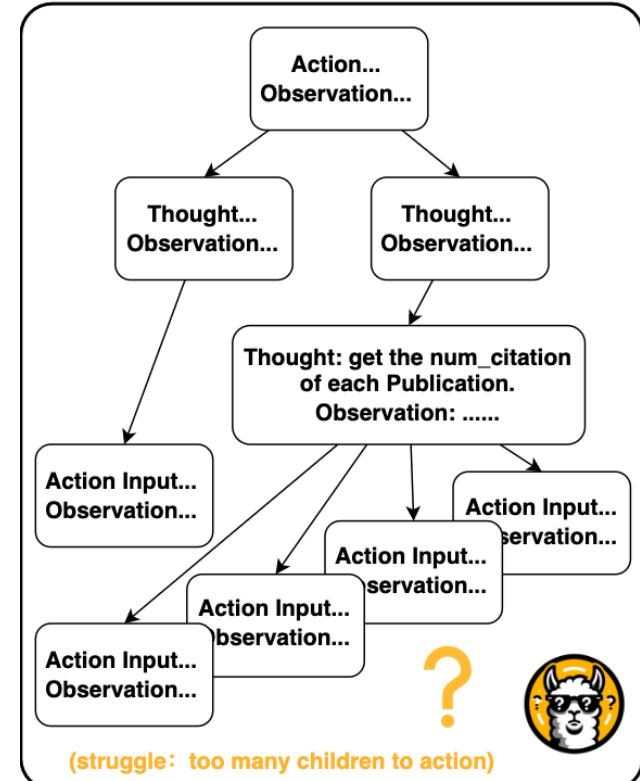
业务问题: 纽约大学Yann LeCun引用最高的论文的引用数是多少?

检索并执行:
API依赖问题

"Functions to call":
[searchPerson, getPersonPubs, getPublication]
"parameters":
[..., person_id: not available, ...]
Could not get person_id

 failed

DFSDT
推理效率太低



[solution]:
searchPerson --> getPersonPubs --> getPublication

[program]:

```
person_list = searchPerson(name = 'Yann LeCun')
if len(person_list) > 1:
    person_list = searchPerson(name = 'Yann LeCun',
                                organization = 'New York University')
    target_person_info = person_list[0]
target_person_id = target_person_info['person_id']
target_person_pubs = getPersonPubs(person_id =
                                    target_person_id)
target_publication_dict = sorted(target_person_pubs,
                                 key=lambda x: x['num_citation'], reverse=True)[0]
target_publication_id =
    target_publication_dict['pub_id']
target_publication_info = getPublication(pub_id =
                                         target_publication_id)
target_citation_count =
    target_publication_info['num_citation']
final_result = target_citation_count
```

[answer]:  correct

Yann LeCun's most cited publication has been cited 74731 times.

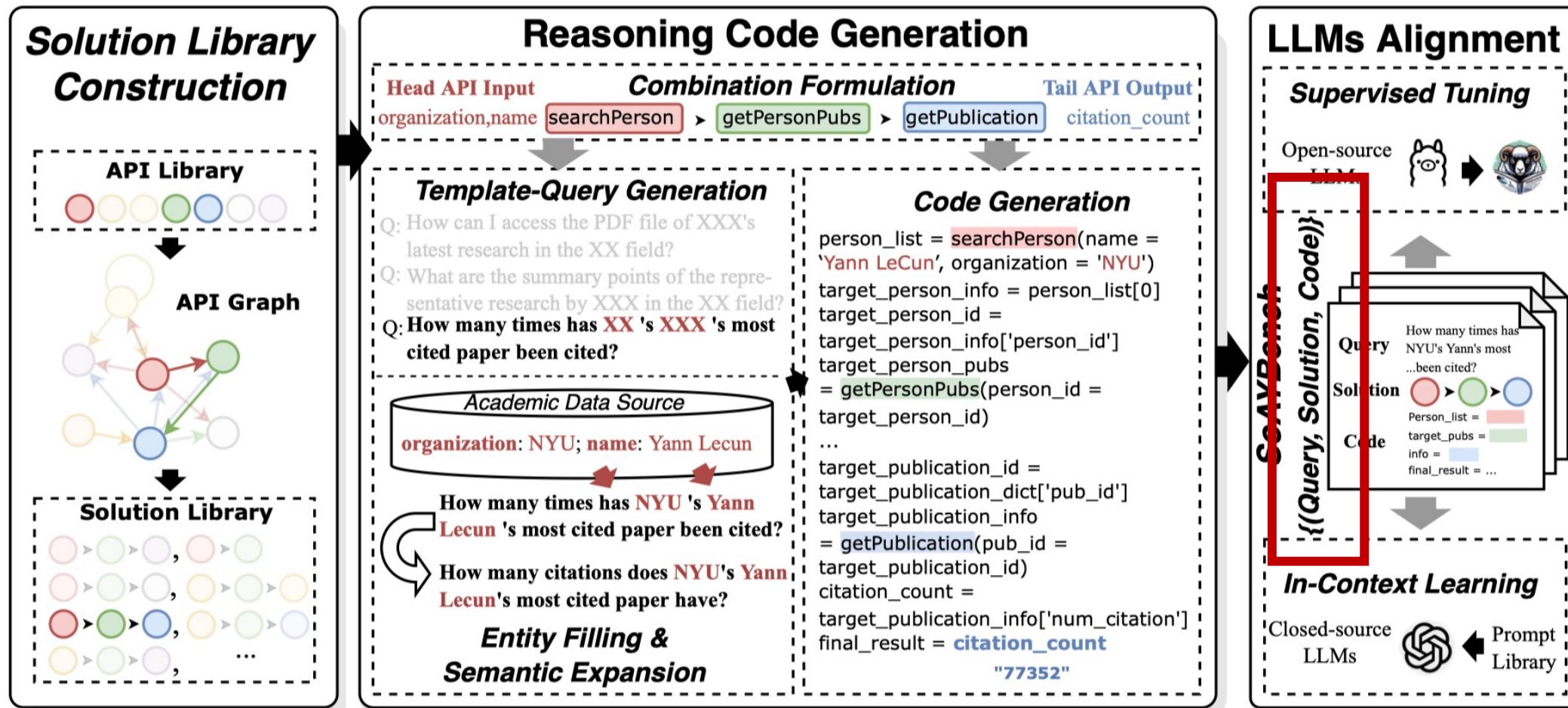


SoAy:

预定义Solution以及
Solution-based代码生成

SoAy: 大模型接口调用解决方法

业务问题: 纽约大学Yann LeCun引用最高的论文的引用数是多少?



大模型接口调用

实验结果

微调模型优于GPT-4, 效率也提升很多

SoAYGPT	3.5	one-hop	27.78±8.70	15.97±7.73	3.47±0.00	13.19±7.80	39.58±9.12	67.36	67.30
		two-hop	33.84±4.94	9.60±4.75	6.06±2.81	13.13±7.12	37.37±5.06	71.21	
		three-hop	22.22±6.43	12.70±5.91	9.52±4.42	13.10±6.72	42.46±6.00	64.68	
	3.5-16k	one-hop	28.47±11.67	15.28±6.12	1.39±0.00	17.36±7.78	37.50±9.07	65.97	66.76
		two-hop	35.86±6.01	7.32±3.41	5.30±2.18	15.91±7.16	35.61±4.65	71.46	
		three-hop	23.02±7.16	10.32±4.99	8.33±3.42	17.46±7.37	40.87±6.26	63.89	
SoAYLLaMA	4	one-hop	0.00±0.00	0.00±0.00	1.39±0.00	2.78±0.00	95.83±5.70	95.83	86.57
		two-hop	15.91±4.71	1.26±0.00	9.34±1.07	2.02±1.69	71.46±3.74	87.37	
		three-hop	6.75±0.00	0.40±0.00	14.68±1.68	1.98±0.00	76.19±3.25	82.94	
	Chat-7B	one-hop	0.00±0.00	0.00±0.00	0.00±0.00	0.69±0.00	99.31±2.94	99.31	85.76
		two-hop	0.00±0.00	0.00±0.00	20.20±3.84	2.53±1.97	77.27±2.70	77.27	
		three-hop	0.00±0.00	0.00±0.00	9.92±3.56	3.17±2.50	86.90±2.72	86.90	
SoAYLLaMA	Code-7B	one-hop	0.69±0.00	0.00±0.00	0.69±0.00	5.56±4.37	93.06±7.50	93.75	88.95
		two-hop	0.25±0.00	3.28±0.00	7.07±2.75	4.80±3.69	84.60±5.18	84.85	
		three-hop	0.40±0.00	0.00±0.00	4.76±2.14	5.16±4.57	89.68±6.54	90.08	
	Code-13B	one-hop	0.00±0.00	0.00±0.00	1.39±0.00	0.00±0.00	98.61±4.03	98.61	92.74
		two-hop	0.00±0.00	2.27±0.00	14.14±2.14	0.51±0.00	83.08±3.32	83.08	
		three-hop	0.00±0.00	0.00±0.00	2.38±2.86	0.40±0.00	97.22±4.28	97.22	

Method	7B	13B	3.5	3.5-16k	4
ToolLLaMA	45.10	/	/	/	/
GPT-DFSDT	/	/	39.12	53.73	70.92
SoAYGPT	/	/	6.15	6.40	26.05
SoAYLLaMA-Code	1.12	1.35	/	/	/

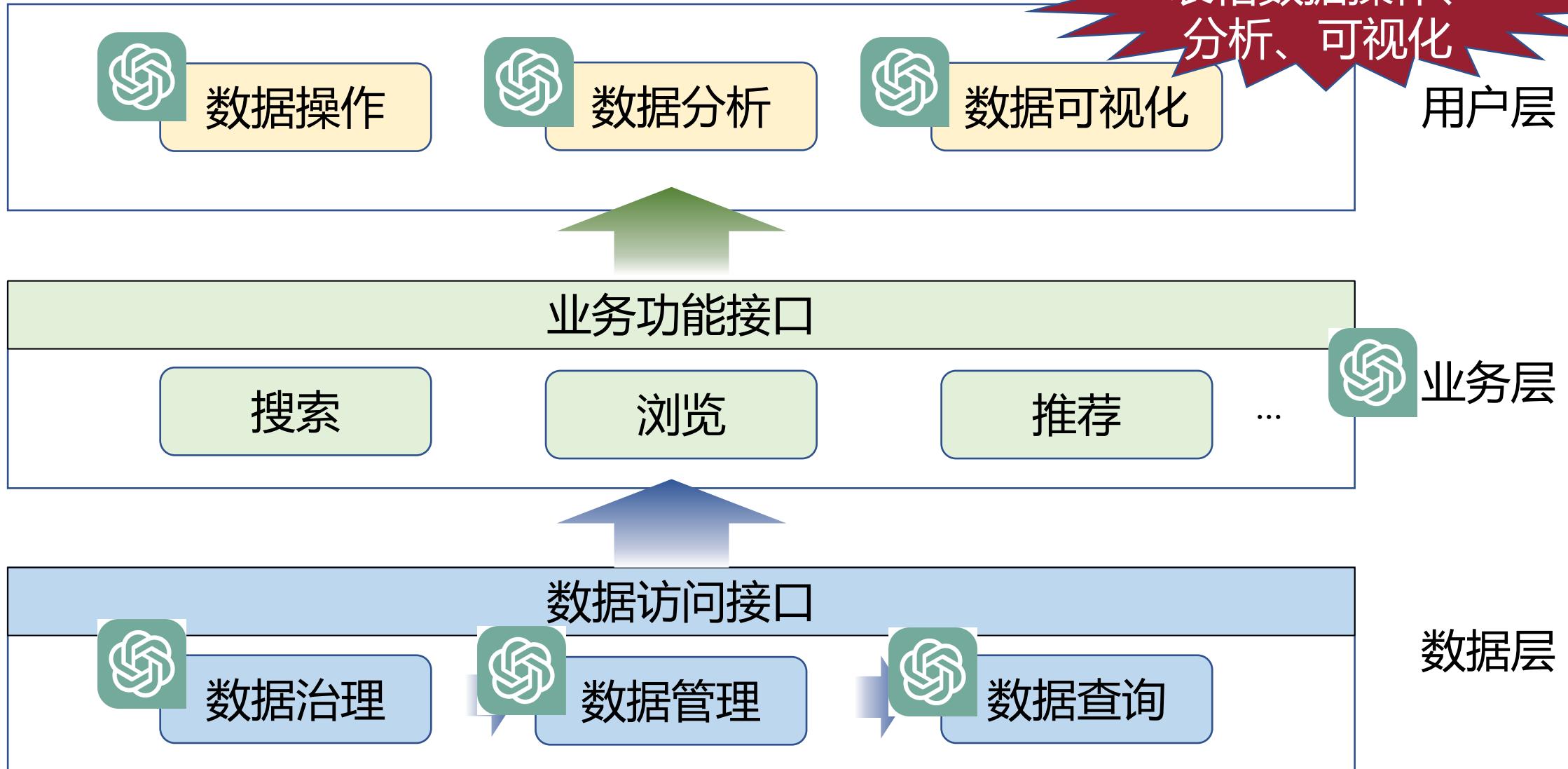
工程能力培养：业务建模

The screenshot shows a web browser window with the URL soay.aminer.cn. The page title is "SoAy: A Service-oriented APIs Applying Framework of Large Language Models". It features two circular logos at the top left. Below the title, there's a logo of a ram reading a book next to the text "x AMiner". A banner below the logo states "Based on 68,000,000+ scholar and 301,000,000+ publication data from AMiner". A large callout box contains a message from the AI: "Hi, this is SoAy x AMiner. If you have any questions about scholars or paper information, feel free to ask me anytime. I will provide you with accurate and reliable academic information with the help of powerful AMiner API." It also lists some sample questions: "You can have a try with these cases: Could you name a few researchers at OpenAI? How many papers has Yann Lecun published? How many citations does Neel Sundaresan from Microsoft have?". At the bottom, there's a text input field labeled "Type your question here" with a send icon.

大模型接口调用的应用场景广泛：

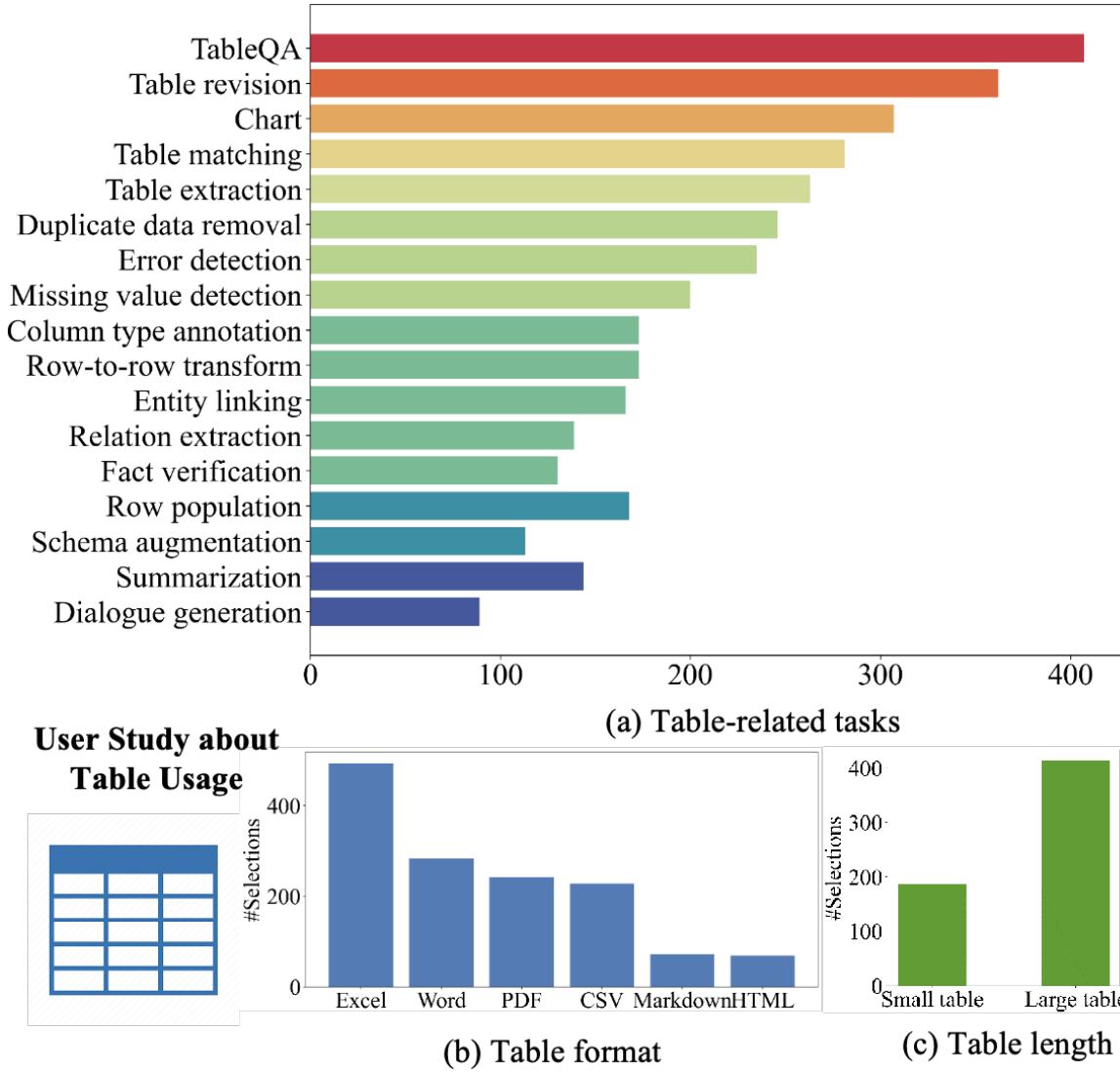
- 1. 商业智能 (BI) :** 允许非技术背景的商业分析师通过自然语言组合数据访问接口，灵活搭建业务功能。
- 2. 教育:** 帮助学习软件工程业务建模的学生验证他们建模并实现的业务功能是否正确。

工程开发流程



表格数据相关任务

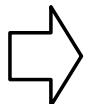
■ **多样的操作需求：**用户偏好的任务涉及广泛的操作，不仅包括表格问答，还有查询、更新、合并、分析、绘制图表等。



合成数据方法一：扩展COT

丰富现有基准数据的推理过程 (COT) , 促进语言模型的训练。

ID	Employer	#Employees
1	Medline	1,200
2	MPD	422
3	Amcor	350
4	FSD 79	287
5	Univ. of SML	220
6	ME School D.	213
7	M. High School	211
8	Village of M	183



Benchmark

Question: how many employers have at least 300 employees?

Answer: 3



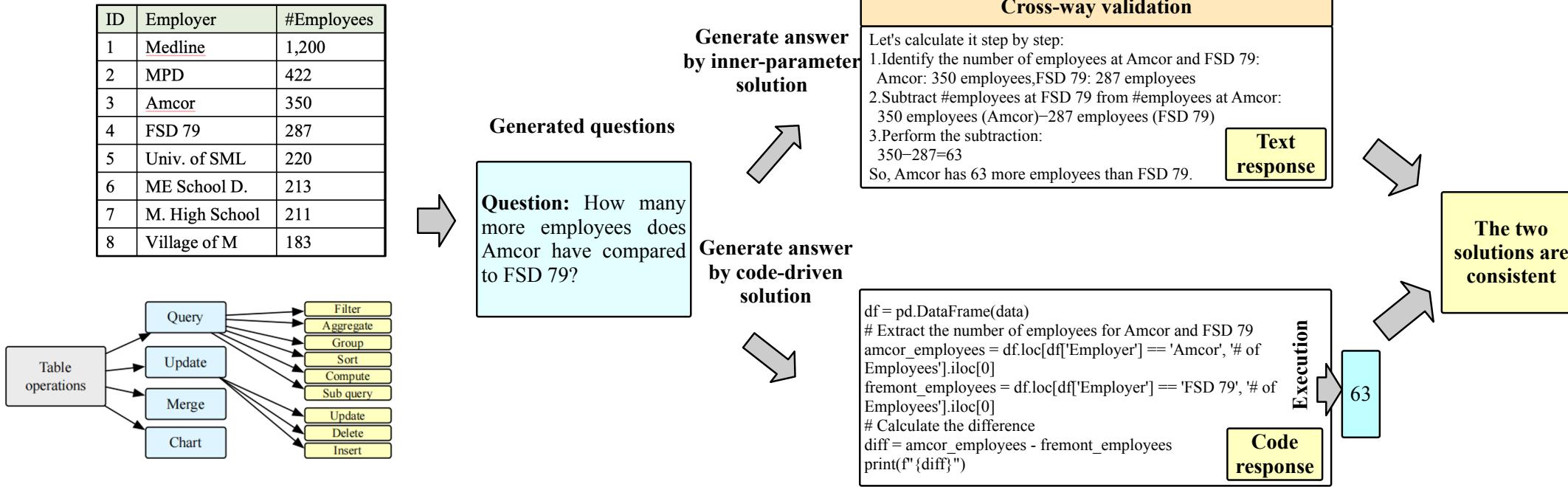
Reasoning process extending

Let's go through each row of the table:

- 1.Medline: #employees is 1,200, which is greater than 300.
 - 2.MPD: #employees is 422, which is greater than 300.
 - 3.Amcor: #employees is 350, which is greater than 300.
 - 4.FSD 79: #employees is 287, which is less than 300.
 - 5.Univ. of SML: #employees is 220, which is less than 300.
 - 6.ME School D.: #employees is 213, which is less than 300.
 - 7.M. High School: #employees is 211, which is less than 300.
 - 8.Village of M: #employees is 183, which is less than 300.
- Based on this analysis, we can conclude that there are 3 employers with at least 300 employees: Medline, MPD, and Amcor.

合成数据方法一：原子操作与一致性检查

为了增加训练数据的多样性和覆盖度，提供给大模型表格**原子操作列表**并采用**交叉验证策略**从提供的表格数据中自动生成新的问题和答案。



定理:

- (1) 如果 A 和 B 是从同一分布中抽取的，并且 $P(Y_a) = P(Y_b) = p > \frac{1}{2}$ ，那么一致性检查的表现优于单独推理，即 $P(Y|E) \geq P(Y_a)$ 。
- (2) 如果 A 和 B 进一步是从独立分布中抽取的，这种效果将更明显(从期望值的角度来看)。

表格数据大模型

实验结果

TableLLM 在表格操作任务上超越GPT-4

Table 2: Overall evaluation in both document-embedded and spreadsheet-embedded tabular data scenarios (%)

Model	Document-embedded tabular data				Spreadsheet-embedded tabular data			Average accuracy	Inference times
	WikiTQ	TAT-QA	FeTaQA	OTT-QA	WikiSQL	Spider	Our created		
TaPEX	38.55	-	-	-	83.90	15.04	-	45.83	1
TaPas	31.60	-	-	-	74.20	23.05	-	42.95	1
TableLlama	24.01	22.25	20.47	6.39	43.70	-	-	23.36	1
Llama2-Chat (13B)	48.82	49.63	67.73	61.50	-	-	-	56.92	1
GPT-3.5	58.45	<u>72.13</u>	71.18	60.80	81.70	67.38	77.08	69.82	1
GPT-4	74.09	77.13	78.35	69.50	84.00	69.53	77.83	75.78	1
CodeLlama (13B)	43.44	47.25	57.24	49.72	38.30	21.88	47.58	43.63	1
Deepseek-Coder (33B)	6.48	11.00	7.12	7.44	72.50	58.40	73.92	33.84	1
StructGPT (GPT-3.5)	52.45	27.53	11.80	13.96	67.80	84.80	-	43.06	3
Binder (GPT-3.5)	61.61	<u>12.77</u>	6.85	5.13	78.60	52.55	-	36.25	50
DATER (GPT-3.5)	53.40	28.45	18.26	13.03	58.20	26.52	-	32.98	100
TABLELLM (7B)	58.77	66.88	72.64	<u>63.11</u>	<u>86.60</u>	82.62	<u>78.83</u>	72.68	1
TABLELLM (13B)	<u>62.40</u>	68.25	<u>74.50</u>	62.51	90.70	<u>83.40</u>	80.83	<u>74.66</u>	1

* Underline represents the runner up.

工程能力培养：业务建模

TableLLM: Manipulating Tables As the Way You Like

Single Table Operation Double Table Operation

- We will provide you a table and a list of possible questions to ask.
- You can choose one of the provided questions or create your own question to have a conversation with the table.
- You can also upload your own file containing table to start a conversation.

Upload your own file if you like

Drag and drop file here
Limit 200MB per file • CSV, XLSX, XLS, DOCX, PDF

Browse files

Provided table:

Refresh Table

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight
0	M	0.455	0.365	0.095	0.514	0.2245	0.101	
1	M	0.35	0.265	0.09	0.2255	0.0995	0.0485	
2	F	0.53	0.42	0.135	0.677	0.2565	0.1415	
3	M	0.44	0.365	0.125	0.516	0.2155	0.114	
4	I	0.33	0.255	0.08	0.205	0.0895	0.0395	
5	I	0.425	0.3	0.095	0.3515	0.141	0.0775	
6	F	0.53	0.415	0.15	0.7775	0.237	0.1415	
7	F	0.545	0.425	0.125	0.768	0.294	0.1495	
8	M	0.475	0.37	0.125	0.5095	0.2165	0.1125	
9	F	0.55	0.44	0.15	0.8945	0.3145	0.151	
10	F	0.525	0.38	0.14	0.6065	0.194	0.1475	
11	M	0.43	0.35	0.11	0.406	0.1675	0.081	
12	M	0.49	0.38	0.135	0.5415	0.2175	0.095	



Draw a histogram of the 'Rings' column to illustrate the distribution of age.



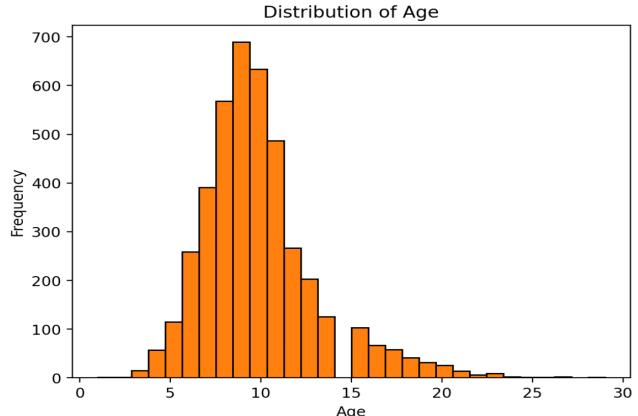
```
import pandas as pd
import matplotlib.pyplot as plt

# Load the data from the csv file
df = pd.read_csv('data.csv')

# Draw a histogram of the 'Rings' column
plt.hist(df['Rings'], bins=30, edgecolor='black')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')

# Show the plot
plt.show()

# Print the result
print('The histogram of the "Rings" column has been drawn.')
```



大模型表格数据操作、分析与可视化的应用场景广泛：

- 1. 商业智能 (BI)**：允许非技术背景的商业分析师通过自然语言直接操作表格、挖掘信息。
- 2. 教育**：帮助学习数据分析工具 (Python、Matplotlib、Sklearn、Pandas) 的学生验证他们撰写分析代码是否正确。

Demo: <https://tablellm.github.io/>

代码理解能力提升：更强大的代码大模型与项目级编程助手

The image displays three sequential screenshots of the CodeGeeX application interface, illustrating its capabilities in understanding code across different levels of scope.

从片段代码理解 (From snippet code understanding): This screenshot shows the initial stage where the AI is asked about a specific code snippet. The user's question is: "这段代码定义了一个类中的几个属性，这些属性用于描述一个管道 (pipe) 对象的状态和行为。下面是对每个属性的解释：" (This code defines several properties in a class, which are used to describe the state and behavior of a pipe object. Below is the explanation for each property). The AI's response details the properties `top_height_px`, `bottom_height_px`, `visible`, and `rect`, along with their descriptions and implementation details.

到整个代码文件理解 (To understanding the entire code file): In this stage, the AI has moved from understanding individual snippets to comprehending the entire file. The user's question is: "@currentFile 这个文件是做什么的" (What does this file do?). The AI provides a detailed explanation of the `eight_puzzle.py` file, which is a solver for the 8-puzzle problem. It describes the classes `Node` and `Solver`, their methods like `getAvailableActions` and `isGoalState`, and various search algorithms such as BFS, DFS, UCS, and A*. It also mentions the `getManhattanDistance` and `getHammingDistance` methods.

再到整个工程的理解 (To understanding the entire project): This final screenshot shows the AI's ability to understand the entire project context. The user's question is: "退方案。" (Retire plan). The AI's response discusses the `script.js` file, which contains the main logic of the game, and the `index.html` file, which serves as the entry point. It highlights how `index.html` links to CSS and JavaScript files and manages game elements like the score board and stage.

CodeGeeX (智谱AI)

Debug能力的提升：学生当助教，写Test Case，给修Bug建议

Problem: first_num_greater_than A

Write a Python function `first_num_greater_than(numbers_list, key)` that takes a list of integers (`numbers_list`) and an integer key (`key`), and returns the first number in the list that is greater than the key. If there is no number greater than the key, then you should return `None`.

Now you are chatting with a student. Please explain to them why their code is wrong by selecting the right explanation from the list. If you are right, the student will fix their code accordingly!

Student's Current Code D

```
1 def first_num_greater_than(numbers_list, key):  
2     for i in range(len(numbers_list)):  
3         if numbers_list[i] > key:  
4             return numbers_list[i]  
5     return None
```

View Code Differences F

```
for i in range(len(numbers_lis) for i in range(len(numbers_lis)  
if numbers_list[i] > key: if numbers_list[i] > key:  
    return numbers_list[i] return numbers_list[i]  
else: return None
```

Test Suite Development B

Add Test Case:

Input Expected output



Add Test Group:

Enter test group name +

Evaluate Test Suite

Your Test Cases

Passed? Actual Output

Default Group

assert(first_num_greater_than
([1, 2, 3], 2) == 3) ✓ 3 delete

assert(first_num_greater_than
([1, 2, 3], 1) == 2) ✓ 2 delete

1. No number in list greater than key

assert(first_num_greater_than
([3, 2, 1], 3) == None) ✓ None delete

2. Key in middle of list

assert(first_num_greater_than
([3, 2, 1], 2) == 3) ✓ 3 delete

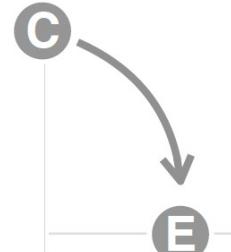
Office Hour Queue

There are several students waiting for your help, please click to start chatting with them.

Bob: Start helping

Chelsea: Waiting...

Dave: Waiting...



Office Hour!

Consider this test case: given input [1, 2, 3], 2, it is supposed to outputs 3, however, the actual behavior is unexpected as the program outputs None

Ok, I see my code got this test case wrong. Could you explain what's wrong with my code?

Your code returns None if the first number in the list is not greater than the key. It doesn't check the rest of the numbers in the list

I see! I've moved the 'return None' statement outside of the for loop. Is it good now?

Is the student correct?

Yes No I don't know

Your code returns None if the first number in the list is not...

All of your test cases passed.

Send

总结：利用大模型提升学生工程实践能力

■ 数据层

- 大模型for数据治理、数据管理与数据查询
- 辅助ER图、DBA、SQL的学习

■ 业务层

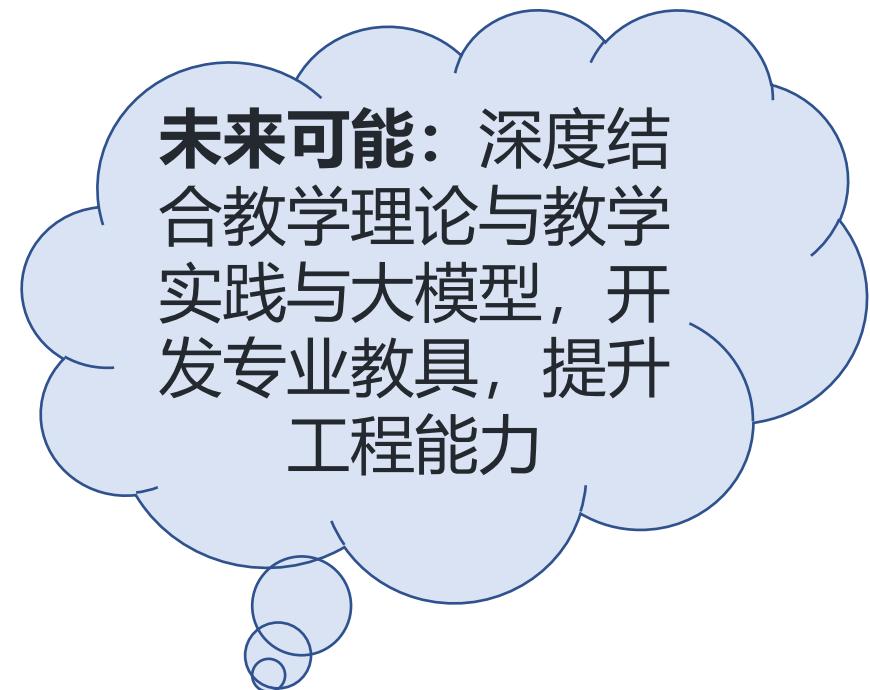
- 大模型for接口调用
- 辅助软件工程业务建模的学习

■ 用户层

- 大模型for表格数据操作、分析与可视化
- 辅助数据分析工具的学习

■ 更强大的通用代码模型：提升代码理解能力

■ 特定应用：提升Debug能力





感谢聆听!

