

大数据分析软件及其应用 (STA323) Project2

BIG DATA ANALYSIS SOFTWARE AND APPLICATION
(HADOOP OR SPARK) SPRING 2025

李思晴 12212964



01.

Text-to-SQL System

- 数据选择方法研究
- SynSQL-2.5M 数据集
- 微调 Qwen2.5-0.5B-Instruct 模型
- Text-to-SQL 演示





数据选择方法研究

Smaller Language Models are capable of selecting Instruction-Tuning Training Data for Larger Language Models

- 基于学习百分比 (Learning Percentage, LP) 的数据选择方法，用于为大型语言模型 (LLMs) 选择高质量的指令调优训练数据。
- 困惑度 (perplexity) 是一种衡量模型对数据不确定性的指标，困惑度越低，表示模型对该数据的预测越准确。
- 学习百分比 (LP) 是衡量数据样本难度的指标，基于模型在训练过程中对样本的学习进度来定义。对于一个数据样本，在第 (i) 个训练周期 (epoch) 结束时，其学习百分比定义如下：

$$LP(i) = \frac{P_{i-1} - P_i}{P_0 - P_n}$$

- P_0 是样本在训练开始时的困惑度 (perplexity)。
- P_i 是样本在第 (i) 个训练周期结束时的困惑度。
- P_n 是样本在训练结束时的困惑度。

- $LP(i)$ 反映了样本在第 (i) 个训练周期中的学习进度。如果一个样本在早期训练周期中困惑度下降得很快，说明它比较容易学习；反之，如果困惑度下降得很慢，说明它比较难学。



数据选择方法研究

**Smaller Language Models are capable of selecting Instruction-Tuning
Training Data for Larger Language Models**

具体步骤

通过衡量数据样本的学习难度来筛选出对模型训练最有价值的数据。

- 对数据样本进行聚类，以保证数据多样性

使用 all-MiniLM-L6-v2 模型对整个训练数据集中的每个句子生成嵌入向量，对这些嵌入向量应用 k-means 聚类算法，将数据划分为多个簇。

- 计算 LP(1) 分数

对于每个训练样本，计算其在第一个训练周期结束时的学习百分比 LP(1)。

- 数据分桶与选择

在每个聚类簇内，根据 LP(1) 分数对样本进行升序排列。从每个簇中选择最难学习的样本（即 LP(1) 分数最低的样本），选择比例为 top-k%。

数据选择方法研究

Instruction Mining: Instruction Data Selection for Tuning Large Language Models

- INSTRUCTMINING: 这是一种专为自动选择用于微调 LLMs 的高质量指令数据而设计的方法。
- 增加用于微调语言模型的子集大小时，观察到了双重下降现象，这表明一旦数据规模超过特定阈值，模型性能的主要决定因素就会从数据质量转变为数据数量。
- **关注初始的一组高质量数据点比浏览整个数据集更能有效地确定最佳点。**

BLENDSEARCH

- **作用:** BLENDSEARCH 结合了贝叶斯优化和局部搜索，能够在对数均匀分布中随机采样数据集大小，以找到在评估集上损失最小的数据子集。这种方法既考虑了数据质量，也考虑了数据量，有效平衡了两者的权衡。

$$\begin{aligned} \text{Model Evaluation Loss} & \downarrow \\ -Q_{D|M,S} \propto \log L(M_{ft}, D_{eval}) & \simeq L_0 + F\{I_1(D), I_2(D), \dots, I_i(D), \dots, I_n(D)\} \\ \text{Instruction Quality} & \uparrow \\ \text{Minimal Loss Constant} & \uparrow \\ \text{i-th Indicator on data } D & \downarrow \\ \text{Bag of Indicators} & \end{aligned} \tag{2}$$



数据选择方法研究

Active Instruction Tuning: Improving Cross-Task Generalization by Training on Prompt Sensitive Tasks

- 本文提出了一个名为Active Instruction Tuning的框架，用于在大规模指令调优（Instruction Tuning）中高效选择数据，以提升模型的跨任务泛化能力。
- 其数据选择方法的核心是基于提示不确定性（Prompt Uncertainty）的任务选择策略。**
- 这种策略的目标是识别出对当前模型最有信息量（即最能提升模型泛化能力）的任务。

Prompt Uncertainty

根据计算出的提示不确定性 U_t ，选择不确定性最高的任务进行训练。这些任务被认为是模型最不确定的任务，训练这些任务可以显著提升模型的泛化能力。

通过衡量模型在原始提示和扰动提示下的预测概率的差异来评估任务的不确定性。

- 选择任务实例：**从任务 t 的数据集中随机选择 n 个实例 $x_{i=1}^n$ 。
- 生成扰动提示：**对每个实例 x_i 的原始提示 I_0 进行 k 次扰动，生成 k 个扰动提示 I_j 。 $j=1, \dots, k$ 。扰动方法包括随机删除单词等，以保持提示的语义不变。
- 计算预测概率：**对于每个实例 x_i ，计算模型在原始提示($I_{\{0\}}$)下的预测概率 $p_{i,0} = P(y_i|x_i, I_0, W)$ ，以及在每个扰动提示 I_j 下的预测概率 $p_{i,j} = P(y_i|x_i, I_j, W)$ 。
- 计算平均差异：**计算每个实例的预测概率差异的平均值：

$$U_t = \frac{1}{n} \sum_{i=1}^n \frac{1}{k} \sum_{j=1}^k |p_{i,0} - p_{i,j}|$$



SynSQL-2.5M 数据集

数据集概述

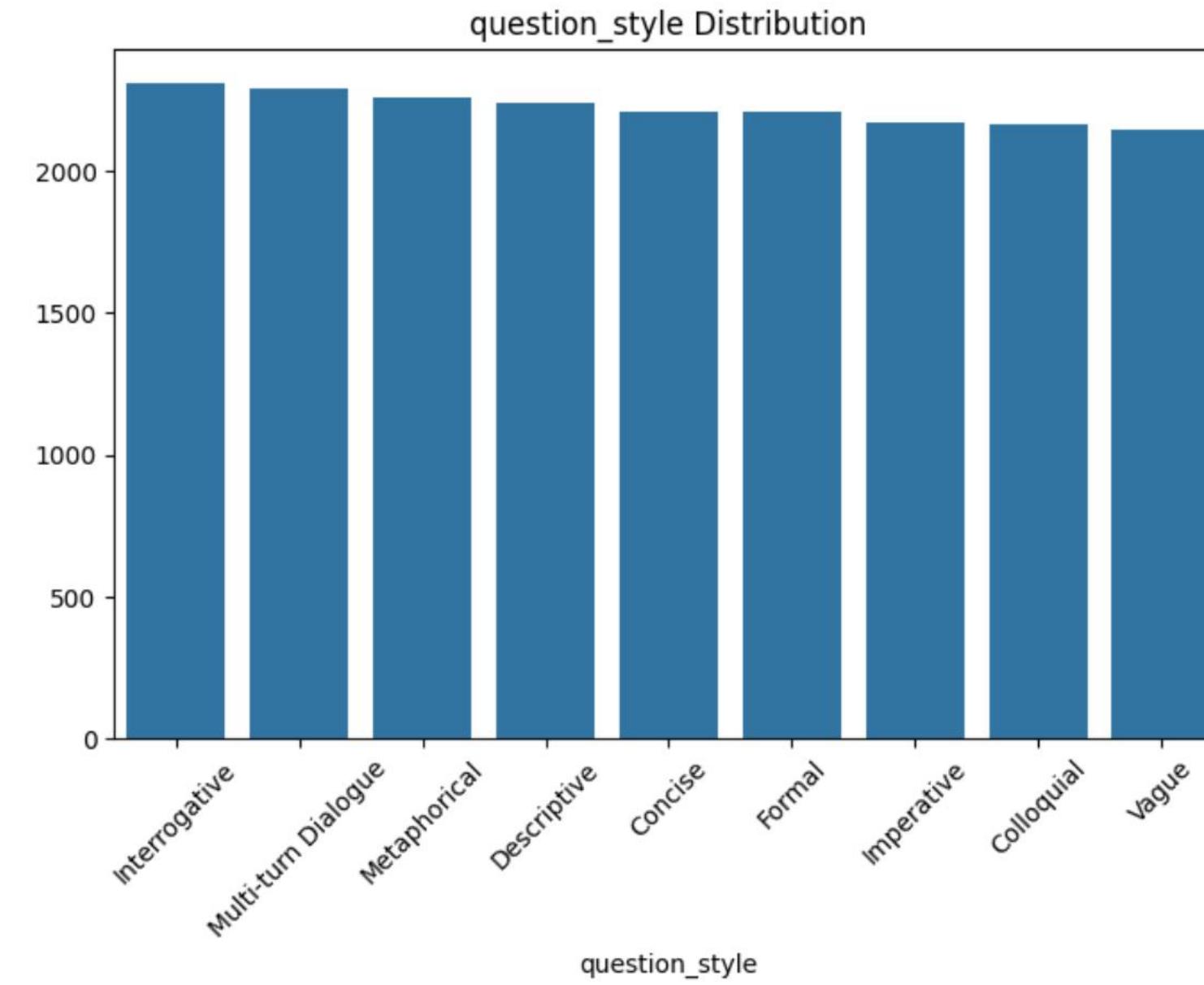
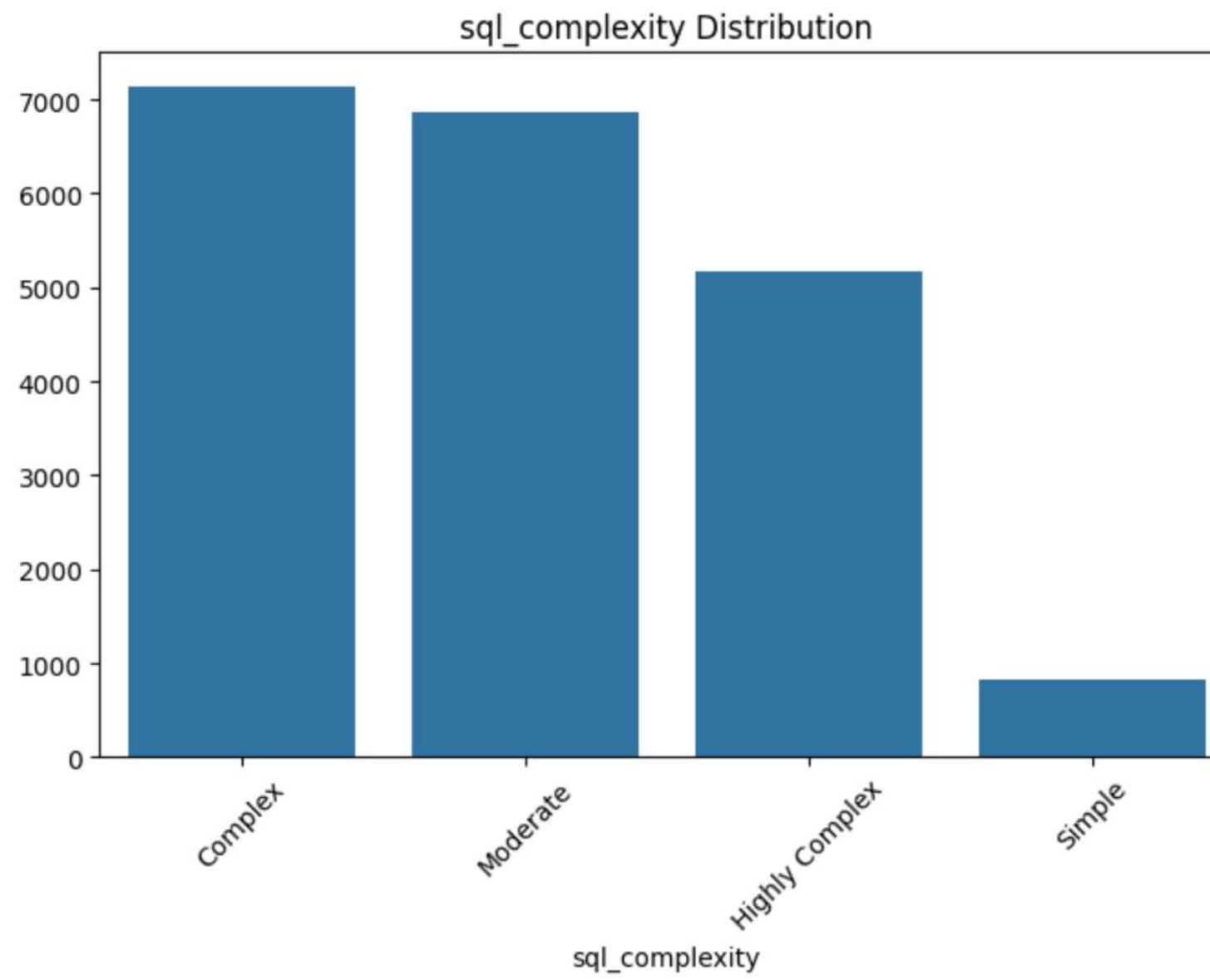
字段统计信息：

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   db_id            20000 non-null   object  
 1   sql_complexity  20000 non-null   object  
 2   question_style  20000 non-null   object  
 3   question         20000 non-null   object  
 4   external_knowledge  20000 non-null   object  
 5   cot              20000 non-null   object  
 6   sql              20000 non-null   object  
dtypes: object(7)
memory usage: 1.1+ MB
None
```

- **db_id**: 数据库标识符，用于标识与问题相关的数据库。、
- **sql_complexity**: SQL复杂度，描述了将自然语言问题转换为SQL查询的难易程度。
- **question_style**: 问题风格，描述了问题的表述方式。
- **question**: 问题，这是用户提出的自然语言问题，需要被转换为SQL查询。
- **external_knowledge**: 外部知识，提供了解决问题可能需要的额外信息或上下文。
- **cot**: 链式思维（Chain of Thought），这是将自然语言问题转换为SQL查询的详细步骤和推理过程。它通常包括识别关键元素、关注相关表、构建查询、排序等步骤。
- **sql**: SQL查询，这是根据自然语言问题和外部知识生成的可执行的SQL查询语句。

SynSQL-2.5M 数据集

数据集概述



SynSQL-2.5M 数据集

数据筛选

使用BLENDSEARCH方法进行选择数据选择

- 1.计算轻量特征：提取SQL语句长度、问题词数、思维链密度、文本多样性等特征。
- 2.生成TF-IDF特征：对问题文本进行向量化，并计算最近邻距离作为额外特征。衡量问题文本的多样性。
- 3.构建质量评分模型：根据特征权重计算每个样本的质量评分。质量评分综合考虑了SQL复杂度、问题长度、思维链密度、词汇多样性、SQL模式以及TF-IDF特征。
$$\text{quality_score} = \sum(\text{feature} \times \text{weight})$$
- 4.优化分层抽样：基于SQL复杂度和问题风格分层，动态调整抽样比例，选择高质量样本。
- 5.BlendSearch优化：通过优化算法进一步筛选最优子集。
- 6.划分训练集和验证集：从最优子集中随机抽取训练集和验证集，并保存结果。

```
def evaluate_subset(indices):
    """评估子集质量的函数"""
    subset = candidate_pool.iloc[indices]

    # 评估指标 (质量分均值 + 多样性惩罚项)
    quality = subset['quality_score'].mean()
    diversity_penalty = 0.01 * subset.duplicated(subset=['question_style', 'sql_complexity']).sum()
    return -(quality - diversity_penalty) # 最小化负质量
```

```
def trainable(config):
    """可训练的函数接口"""
    n_samples = config["n_samples"]
    # 从候选池中随机选择指定数量的样本
    indices = np.random.choice(len(candidate_pool), n_samples, replace=False)
    return evaluate_subset(indices)
```



SynSQL-2.5M 数据集

转换数据格式

按照Qwen模型训练和OmniSQL模型的要求准备所选数据

转换样本格式

`convert_to_qwen_format`函数将单个样本转换为Qwen模型所需的格式。具体步骤如下：

- 从样本中提取数据库ID、问题文本、思维链（cot）、SQL查询和外部知识。
- 构造系统消息（system_msg），包含数据库的表结构描述。
- 构造用户消息（user_msg），包含问题文本和可选的外部知识。
- 根据enable_thinking参数决定是否包含思维链，构造目标输出（target），格式为<think>思维链</think>\nSQL查询或仅SQL查询。
- 返回一个字典，包含系统消息、用户消息和目标输出。转换后的样本写入JSONL文件中，每行一个样本。

训练集 question_style 分布:	数量	比例 (%)
Metaphorical	238	11.90
Formal	236	11.80
Interrogative	234	11.70
Vague	231	11.55
Concise	231	11.55
Multi-turn Dialogue	224	11.20
Colloquial	210	10.50
Descriptive	209	10.45
Imperative	187	9.35

验证集 question_style 分布:	数量	比例 (%)
Vague	66	13.2
Multi-turn Dialogue	63	12.6
Interrogative	61	12.2
Formal	60	12.0
Descriptive	55	11.0
Concise	51	10.2
Colloquial	48	9.6
Metaphorical	48	9.6
Imperative	48	9.6



微调Qwen2.5-0.5B-Instruct模型

任务目标

- 将自然语言文本转换为 SQL 查询语句及其思考过程。
- 数据格式：JSONL，包含对话内容（messages）和输出（output）。

数据处理

- 加载数据：通过 datasets 库加载训练集和验证集。
- 构建提示：将对话内容和输出拼接为单一对话上下文。
- 编码处理：使用分词器将文本转换为模型输入格式，设置最大序列长度为512。

模型与训练配置

- 使用 Qwen2.5-0.5B-Instruct 预训练模型。

调试与优化

- 引入 Ray 进行调试，支持分布式训练和资源管理。通过 Ray 动态分配计算资源，优化训练效率。

训练与保存

- 训练过程：调用 trainer.train() 启动训练，每 100 步评估和保存一次。
- 模型保存：训练完成后，将模型和分词器保存到指定目录。

```
# ----- TrainingArguments ----- #
training_args = TrainingArguments(
    output_dir=args.output_dir,
    num_train_epochs=args.epochs,
    per_device_train_batch_size=args.train_bs,
    per_device_eval_batch_size=args.eval_bs,
    gradient_accumulation_steps=args.grad_accum,
    learning_rate=args.lr,
    eval_strategy="steps",
    eval_steps=100,
    save_strategy="steps",
    save_steps=100,
    logging_steps=50,
    fp16=False,
    bf16=use_cuda,
    report_to="none",
    no_cuda=args.no_cuda,
    ddp_find_unused_parameters=False,
    dataloader_num_workers=4,
    gradient_checkpointing=True,
)
```

5个epoch大致收敛，在验证集上的 eval_loss 约为 0.73



Text-to-SQL 演示

功能描述

- SQL 生成：用户输入自然语言问题，模型生成对应的 SQL 查询语句。
- SQL 解释：对生成的 SQL 语句进行详细解释，帮助用户理解。
- 流式生成：支持实时流式输出 SQL 生成和解释内容。

界面与交互

- Gradio 界面：提供用户友好的 Web UI，支持聊天式交互。
- 输入框：用户输入自然语言问题。
- 聊天框：实时显示生成的 SQL、解释和执行结果。
- SQL 显示区：高亮显示生成的 SQL 代码。
- 控制按钮：提交问题、清除历史、重试生成。

技术栈

- 模型：Qwen2.5-0.5B-Instruct
- 框架：Hugging Face Transformers、Gradio

```
File Explorer | Editor | Search | Problems | Output | Debug Console | Ports | Help
```

File: demo.py

```
def _launch_demo(args, model, tokenizer, ddl_text):
    task_history = get_state([1])
    submit_btn.click(
        predict,
        (query, chatbot, task_history, sql_display),
        [chatbot, task_history, sql_display],
        show_progress=True,
    )
    submit_btn.click(reset_user_input, [j], [query])
    empty_btn.click(
        reset_state,
        [chatbot, task_history, sql_display],
        [chatbot, task_history, sql_display],
        show_progress=True,
    )
    regen_btn.click(
        regenerate,
        [chatbot, task_history, sql_display],
        [chatbot, task_history, sql_display],
        show_progress=True,
    )
    gr.Markdown(
        f"<font size=2>Note: this demo is governed by the original license of Quen2.5. <br/> This demo only developed for learning propose. </font>"
    )
    demo.queue().launch(
        share=args.share,
        inbrowser=args.inbrowser,
        server_port=args.server_port,
        server_name=args.server_name,
    )

def main():
    args = _get_args()
    model, tokenizer = _load_model_tokenizer(args)

    ddl_text = ""
    if args.db:
        try:
            executor = SQLExecutor(args.db)
            ddl_text = executor.schema()
            print(f"Loaded database schema from: {args.db}")
        except Exception as e:
            ddl_text = f"Error loading database: {e}"
            print(ddl_text)

    _launch_demo(args, model, tokenizer, ddl_text)
```

Terminal

```
GROUP BY
    c.document_id, u.username
HAVING
    COUNT(c.document_id) > 1
ORDER BY
    c.document_id
Users: What are the usernames of initiators who have completed multiple comparisons, along with the names of the authors of the compared documents and the average similarity scores between these documents? Completed comparisons refer to those with a 'completed' status in the 'status' column of the comparisons table. The average similarity score is a measure of how similar the documents are, with higher scores indicating greater similarity.
Generated SQL:
WITH UserComparisonCounts AS (
    SELECT
        u.id,
        COUNT(*) AS comparison_count
    FROM
        comparisons
    GROUP BY
        u.id
),
UserAuthors AS (
    SELECT
        u.id,
        name,
        COUNT(*) AS author_count
    FROM
        users
    JOIN comparisons ON users.u_id = comparisons.u_id
    WHERE
        comparisons.completed = 'Y'
    GROUP BY
        u.id, name
),
SimilarityScores AS (
    SELECT
        c.id AS comparison_id,
        ca.name AS author_name,
        AVG(similarity) AS avg_similarity_score
    FROM
        comparisons c
    JOIN UserAuthors ua ON c.u_id = ua.u_id
    JOIN usercomparisoncounts uc ON c.id = uc.comparison_id
    JOIN users u ON uc.name = u.name
    JOIN comparisons sim ON sim.id = c.id AND sim.completed = 'Y'
    GROUP BY
        c.id, ca.name
)
SELECT
    sc.id AS username,
    sa.name AS author_name,
    sc.avg_similarity_score
FROM
    similarities sc
JOIN
    userAuthorities ua ON sc.id = ua.id
JOIN
    usercomparisoncounts uc ON sc.id = uc.comparison_id
JOIN
    comparisons sim ON sc.id = sim.id AND sim.completed = 'Y'
JOIN
    users u ON uc.name = u.name
WHERE
    sc.id IN (
        SELECT id FROM usercomparisoncounts
        WHERE comparison_id IN (
            SELECT id FROM SimilarityScores
            GROUP BY id
            HAVING count(*) > 1
        )
    )
ORDER BY
    sc.id, sc.avg_similarity_score;
Keyboard interruption in main thread... closing server.
(qmz_mrw) [base] yang:767@ceg-cmc167889d:~/qmz_mrw
```

File Edit Selection View Go Run Terminal Help

cli.py - qwen [SSH: pcv0] - Cursor

eb.py cli.py datafram Deploy text-to Terminal bash - qwen bash qwen bash qwen

QWEN [SSH: pcv0]

artifacts/qwen_t2sql... validate_storage_m... trainer.pkl tuner.pkl qwen_t2sql_ckpt checkpoint-96 added_tokens.json config.json generation_config.j... merges.txt model.safetensors optimizer.pt rng_state.pth scheduler.pt special_tokens_ma... tokenizer_config.js... trainer_state.json training_args.bin vocab.json added_tokens.json chat_template.jinja config.json generation_config.js... merges.txt model.safetensors special_tokens_map... tokenizer_config.json train.log training_args.bin vocab.json chinook.db cli.py qwen_selected_train.j... qwen_selected_valida... school.db train.py

152 d_prompt(question: str, ddl: Optional[sal 156 "Respond ONLY with the SQL query, without 157 "Do not wrap the SQL in markdown code blo 158 f"(schema_part)" 159 f"### Instruction:\n{question}\n" 160 "### Response:\n" 161 162 _stream(model, tokenizer, prompt): 163 ts = tokenizer([prompt], return_tensors=" 164 amer = TextIteratorStreamer(tokenizer, sk 165 skip_special_ 166 ad(target=model.generate, kwargs={**input 167 token in streamer: 168 yield token 169 170 act_sql(text: str) -> str: 171 # 模型输出中提取纯 SQL 语句"" 172 试匹配 SQL 代码块 173 block_match = re.search(r'`sql\s*(.*?)\ 174 ql_block_match: 175 sql = sql_block_match.group(1).strip() 176 : 177 # 尝试匹配普通代码块 178 code_block_match = re.search(r'`(.*)? 179 if code_block_match: 180 else: 181 # 如果没有代码块, 尝试识别 SQL 关键字后 182 sql_keyword_match = re.search(r'\b(SE 183 if sql_keyword_match: 184 sql = sql_keyword_match.group(0). 185 186

Ctrl+K to generate a command

(qwen_env) (base) yang.7670@ceg-cnc197009d:~/qwen\$ ls artifacts cli.py qwen_selected_validation.jsonl train.py web.py chinook.db qwen_selected_train.jsonl qwen_t2sql_ckpt web_demo_py_from_qwen.py

(qwen_env) (base) yang.7670@ceg-cnc197009d:~/qwen\$ python train.py --train_file "./qwen_selected_train.jsonl" --val_file "./qwen_selec ted_validation.jsonl" --output_dir ./qwen_t2sql_ckpt --epochs 3 --train_bs 2 --eval_bs 2 --grad_accum 8

2025-05-30 23:46:42,229 INFO 找到 3 个 GPU:

2025-05-30 23:46:42,231 INFO GPU 0: NVIDIA GeForce RTX 3090

2025-05-30 23:46:42,233 INFO GPU 1: NVIDIA GeForce RTX 3090

2025-05-30 23:46:42,234 INFO GPU 2: NVIDIA GeForce RTX 3090

2025-05-30 23:46:42,235 INFO 使用设备: cuda

2025-05-30 23:46:42,236 INFO 使用 GPU 训练, 将 batch size 增加到: train_bs=4, eval_bs=4

| 0/2000 [00:00<?, ? examples/s] Map: 0% | 0/2000 [00:00<?, ? examples/s]

Ctrl+K to generate a command

(qwen_env) (base) yang.7670@ceg-cnc197009d:~/qwen\$ python cli.py -c ./qwen_t2sql_ckpt --db school.dp

Filter log text, **/ts/*/*node_modules/*

No problems have been detected in the workspace.

OUTLINE

TIMELINE

SSH: pcv0 0 0 1

18°C 气温

Cursor Tab 搜索 Spaces: 4 UTF-8 LF Python 3.9.21 ('qwen_env': conda) 127 2025/5/31 2025/05/31 01:27:11

02. Startup Business Plan

- 商业计划路演
- 大预言模型示例





项目概况



定位：K12 智能辅导平台

核心技术：知识图谱 + 学习路径规划 + 多模态生成

目标用户：6-18 岁学生

愿景：7×24 小时的“数字学习伙伴”

项目名称：智学通

Slogan：让学习更智能，让成长更高效

“智学通”是一款深度融合大语言模型辅导平台，定位于“以学习科学为内核、以生成式 AI 为引擎、以数据闭环为驱动”的在线学习基础设施。



用户痛点

● 学习效率低

传统课堂基于“平均数”展开教学，导致“高分重复练、低分听不懂”的普遍现象。

● 个性化辅导缺位

无法根据学生的错误思路提供多路径讲解。

● 家庭监督低效

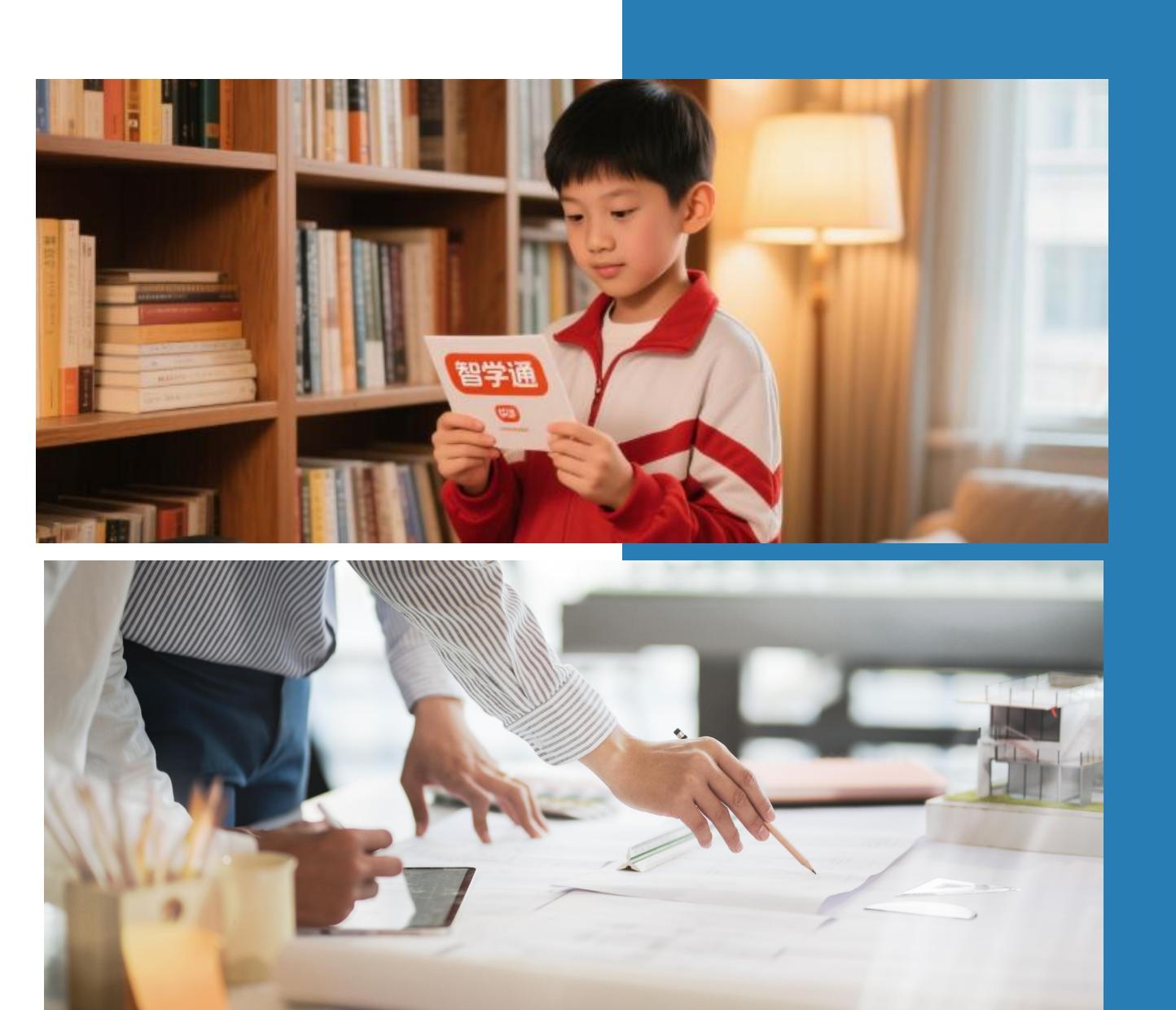
家长投入了大量陪伴时间，却收效甚微，家庭冲突加剧。

● 教师教学负担加重

“双减”政策后，本应回归校园的个性化辅导由校内教师承担，出现普遍的“疲劳式教学”

● 家庭在教育上花销巨大

孩子的养育成本占家庭收入的比例接近50%，而其中教育支出占养育成本比例达34%。





行业分析

政策环境

2024年9月，联合国教科文组织发布了两个人工智能能力框架：《学生人工智能能力框架》和《教师人工智能能力框架》。其中提出将面向未来和本地化的人工智能作为课程载体。

市场规模

K12 在线教育的服务主体为学前教育至高中阶段的受教育群体。艾瑞咨询数据显示，2023 年中国 K12 在线学习工具类产品付费用户约 4200 万，整体市场规模 560 亿元，同比增长 17.8%。

技术支撑

LLM 在中文语义理解、上下文推理、多模态融合等方面已接近高中乃至大学生水平；知识追踪 (KT) 模型如 Deep Knowledge Tracing 迁移到 Transformer 架构后，预测正确率提升。

竞争格局

双减后以“直播大班课”为主的流量打法受到遏制，资本与流量正向“高留存、高粘性”的轻服务工具类迁移。市场处于新一轮窗口期，充满机遇与挑战。

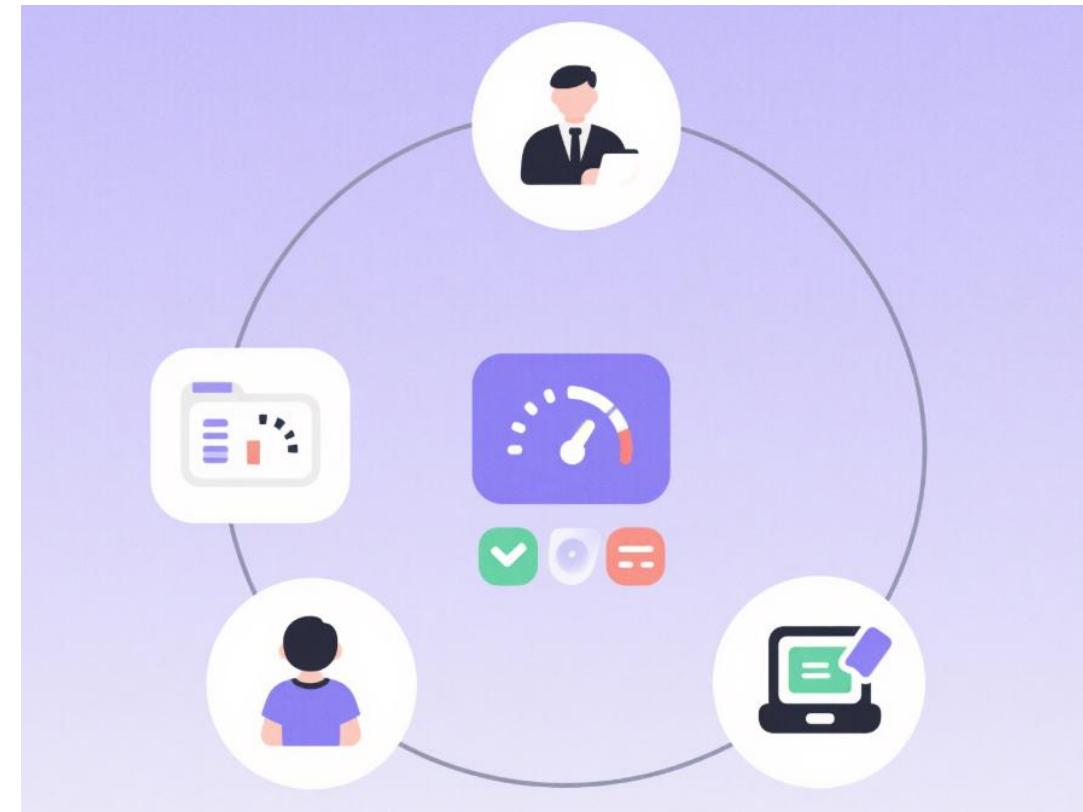
核心功能与产品优势

个性化学习计划

- 智能诊断：首登即进行 15 分钟自适应测评，构建知识图谱掌握度矩阵；
- 动态路径：结合 Ebbinghaus 遗忘曲线与 spaced repetition，在用户答题、观看微课、进行错因标注后即时调整学习节奏；
- 目标管理：学生可视化查看每日/周目标完成率，家长端实时同步。

智能答疑与“思路生产”

- 语义级拍照：OCR + 结构化解析将复杂手写题拆解为公式、文字、图形三类 token；
- 多步推理：LLM 在解析后生成“逐步推理树”，展示每一步思考假设，让学生“学会过程而非背答案”；
- 生成视频：调用虚拟教师 Avatar 即时生成 1-2 分钟讲解，满足听觉学习者需求。



核心功能与产品优势

学习数据闭环

- 行为采集：鼠标/触屏停留、滑动、输入节奏多维记录；
- 因果分析：运用 DoWhy 框架识别“高频跳题→认知负荷过载→正确率下降”链路，向家长推送干预建议；
- 成就系统：以“知识里程碑”代替传统积分，驱动内在动机。

边学边练一体化体验

- 微课-练习-测试串联成“3分钟学习单元”，充分匹配碎片化场景；
- 离线缓存+端侧微推理加速，地铁、乡镇网络不佳亦可流畅使用。

安全与合规

- 遵循《个人信息保护法》《未成年人网络保护条例》：人脸识别仅用于设备登录，所有生物信息加密存储；
- 云端采用国密算法 SM4 加密，日志 30 天脱敏；
- 家长可在“隐私面板”中一键删除或导出子女数据。



竞品分析



猿辅导 (猿题库)

- 商业模式：以拍照搜题切入，延伸到直播大班课；
- 优势：品牌声量大、题库积累深；
- 痛点：答疑仍以“标准步骤”模板输出，缺乏即时个性化反馈；大班课停摆后转型方向尚不清晰。

作业帮

- 商业模式：工具流量→大班课→中小班；
- 优势：日活近 5000 万，低年级渗透率高；
- 痛点：题库同质化、社区内容噪声大、用户学习链条割裂（搜题后需跳转多应用）。

腾讯课堂 / 学而思网校

- 优势：背靠大厂或教育巨头，资金充裕、版权资源多；
- 痛点：课程 SKU 繁杂，用户面临选择困难；AI 技术主要用于客服和推荐，未深入教学场景。

松鼠 AI

- 亮点：早期提出“自适应学习”，拥有线下学习中心；
- 痛点：线下店模式重资产、疫情期间扩张受阻；AI 引擎以知识切片为主，对文本理解与生成薄弱。



商业模式

01

订阅收入

- 基础会员：59 元/月，包含学科全覆盖的个性化学习计划、每日 30 次智能答疑、成长报告；
- 进阶会员：99 元/月，增加“思维导图生成”“AI 口语陪练”“错题一键组卷”等功能；
- 家庭套餐：199 元/月，可绑定 2 名学生 + 2 名家长，平均 ARPU 提升 40%。

02

增值服务

- AI 伴学官” 1 对 1 “视频辅导：¥120/小时，LLM 预处理后由名师进行短时深度点拨，实现“人机协同”降本；
- 题库 API 授权：向出版社、学校开放接口，收取千次调用 30 元；
- 学情数据报告：向校方提供班级、学科维度洞察 SaaS，每年 3 万/校。



发展规划

短期 (1-12 个月)

- 产品迭代：上线小学阶段语文作文 AI 评改、“三维度提分雷达”功能；
- 用户增长：与 50 家网红教育 MCN 深度合作，打通“短视频种草—App 下载—社群伴学”闭环，目标新增 8 万注册；
- 数据验证：完成 3,000 人规模 A/B Test，发布白皮书《生成式 AI 在 K12 个性化学习的实证研究》。

中期 (1-2 年)

- 学科扩容：覆盖全学段英语口语、科学实验仿真，接入自研 3D 物理引擎；
- 生态合作：与人民教育出版社共建题库 API，联合硬件厂商推出“AI 学习平板”；
- 海外市场：针对东南亚中文国际学校推出多语种版本，复制中国经验，打造“教育出海”第二增长曲线。

长期 (3-5 年)

- 领域延伸：切入职业教育和成人终身学习，推出“智学通 Pro”自我提升平台；
- 教育大模型平台化：开放 LLM 微调与知识图谱构建工具，构建开发者生态；
- 社会价值：与教育主管部门合作，提供乡村学校公益账号 10 万个，推动教育公平。



与大模型交互常见错误

- 信息检索不完全
- 逻辑错误
- 幻觉过重

事实性幻觉：是指模型生成的内容与可验证的现实世界事实不一致。

忠实行幻觉：则是指模型生成的内容与用户的指令或上下文不一致。

数据：数据缺陷、数据中捕获的事实知识的利用率较低。

训练：预训练阶段（大模型学习通用表示并获取世界知识）、对齐阶段（微调大模型使其更好地与人类偏好一致）两个阶段产生问题。

推理：固有的抽样随机性，不完美的解码表示。

Table 1. Examples of each category of LLM hallucinations. Content marked in Red represents the hallucinatory output, while content marked in Blue indicates user instruction or provided context that contradicts the LLM hallucination.

Category	Type	Example Dialog	Explanation
Factuality Hallucination	Factual	Instruction: What are Thomas Edison's main contributions to science and technology? Response: Thomas Edison developed the first practical telephone and invented the light bulb.	The response is factually incorrect. In reality, Edison improved the light bulb, building on earlier designs by others , and Alexander Graham Bell invented the telephone .
	Factual Fabrication	Instruction: What are the major environmental impacts of the construction of the Eiffel Tower? Response: The construction of the Eiffel Tower in 1889 led to the extinction of the Parisian tiger , a species that played a crucial role in the region's ecosystem. Additionally, it is widely recognized as the event that sparked the global green architecture movement .	The response erroneously states that the construction of the Eiffel Tower led to the extinction of the 'Parisian tiger' and sparked the global green architecture movement. In fact, there is no evidence of a 'Parisian tiger' ever existing , making this a fabricated claim. Moreover, attributing the origins of green architecture to the Eiffel Tower is an exaggeration , as this movement has diverse roots and cannot be traced back to a single event.
Instruction Inconsistency	Instruction	Instruction: Translate the English question into Spanish: "What is the capital of France?" Response: The capital of France is Paris.	The user explicitly provided an instruction to translate the English question into Spanish. Instead of performing the translation as instructed, the LLM chose to answer the question .
	Inconsistency	Retrieved context: The Nile is a major north-flowing river in northeastern Africa. It's traditionally considered the longest river in the world. Originating in the Great Lakes region of central Africa , it flows through several countries before emptying into the Mediterranean Sea. Instruction: Summarize the retrieval document above.	The summary incorrectly states that the Nile originates from the "mountain ranges of central Africa", while the paragraph mentions that the Nile originates in the Great Lakes region of central Africa .
Faithfulness Hallucination Context Inconsistency	Context	 Response: The Nile originates from the mountain ranges of central Africa and flows into the Mediterranean.	
	Inconsistency	 Instruction: Solve the equation $2x + 3 = 11$ step by step. Response: Step 1: Subtract 3 from both sides to get: $2x = 8$. Step 2: Divide both sides by 2 to get: $x = 3$. So the result of the equation is $x=3$.	The first step correctly isolates the term with x to get $2x = 8$. However, the next step inaccurately divides 8 by 2 to yield a result of $x = 3$, which is inconsistent with the earlier reasoning.



与大模型交互常见错误

➤ 图像生成



THANK YOU FOR WATCHING

BIG DATA ANALYSIS SOFTWARE AND APPLICATION
(HADOOP OR SPARK) SPRING 2025

李思晴 12212964

