

STA323 Project2 Report

李思晴 12212964

2025.5.30

任务一：Text-to-SQL system

1. 数据选择方法研究

1.1 基于学习百分比 (LP) 的数据选择方法 (2402.10430)

本文介绍了一种基于学习百分比 (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。该方法的核心在于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。

学习百分比 (LP) 的定义

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：

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

其中：

- 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。

困惑度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。因此， $LP(i)$ 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

基于 LP 指标和聚类，数据选择方法的具体步骤如下：

1. 数据多样性的增强

- 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 (sentence embeddings)。
- 对这些嵌入向量应用 k-means 聚类算法，将数据划分为多个簇 (cluster)。
- 确保每个簇中包含的样本数量不少于 50 个，以保证每个簇都有足够的数据多样性。

2. 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)。
- 将所有样本按照 LP(1) 分数进行升序排列。分数越低，样本越难学习。

3. 数据分桶与选择

- 在每个聚类簇内，根据 LP(1) 分数对样本进行排序。
- 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%。例如，如果选择 10% 的数据，则从每个簇中选择 LP(1) 分数最低的 10% 的样本。

1.2 INSTRUCTMINING：基于自然语言指标的数据选择方法 ([2307.06290](#))

本文首次提出了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。该方法利用自然语言指标作为数据质量的衡量标准，将其应用于评估未见过的数据集。

在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，即模型性能先变好，然后变差，再变好。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。在这种情况下，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。

鉴于确定这一最佳点的成本敏感性，文章采用 **BLENDSEARCH 算法** 来自动搜索满足需求的最佳子集。

BLENDSEARCH 算法

- **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。

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

1.3 Active Instruction Tuning：基于提示不确定性的任务选择策略 (2311.00288)

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, IT）中高效选择数据（任务），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。

在训练大型语言模型时，由于任务数量庞大，无法对所有任务进行训练，因此需要一种有效的任务选择策略。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。

提示不确定性（Prompt Uncertainty）

提示不确定性是一种任务级别的不确定性度量方法，用于评估模型对任务提示的敏感性。具体来说，它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。

1. **选择任务实例**：从任务 t 的数据集中随机选择 n 个实例 $x_i |_{i=1}^n$ 。
2. **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 $I_j |_{j=1}^k$ 。扰动方法包括随机删除单词等，以保持提示的语义不变。
3. **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 $p_{i,j} = P(y_i | x_i, I_j, W)$ 。
4. **计算平均差异**：计算每个实例的预测概率差异的平均值：

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

其中， $|p_{i,0} - p_{i,j}|$ 表示原始提示和扰动提示下的预测概率的绝对差异。

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。

2. SynSQL-2.5M 数据集

数据集包含2,544,390个不同且复杂的文本到SQL样本，每个样本都由一个象限组成。

`<database, question, SQL query, chain-of-thought solution>`。涵盖来自真实场景的16,583个合成数据库。

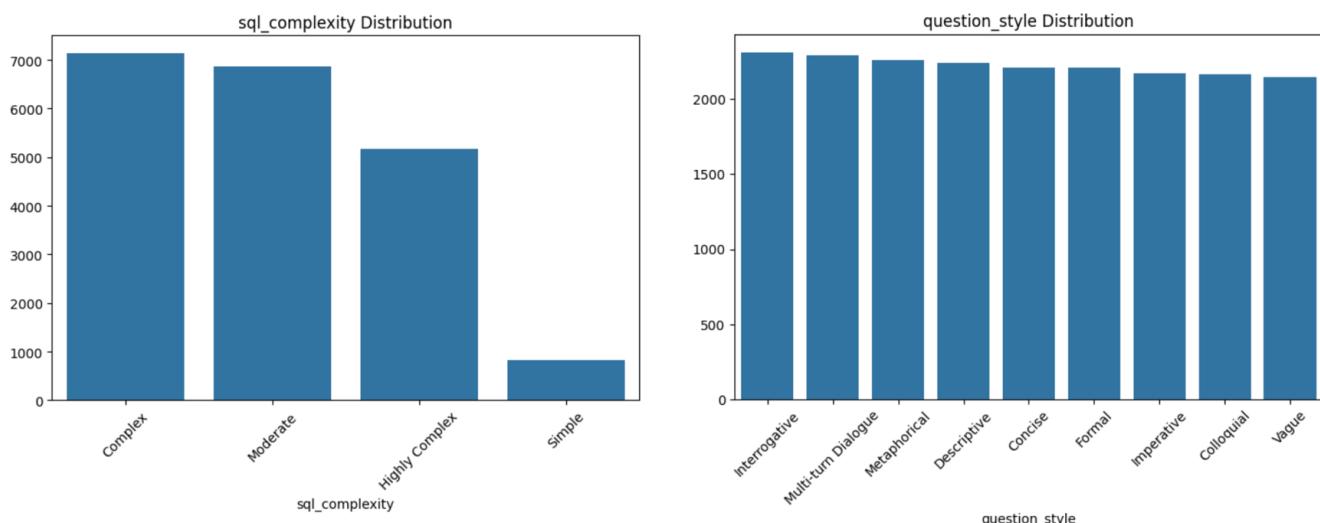
在我们的任务重，我们将提取其前20000条数据，并进行筛选，最后得到2000条作为训练集，500条作为验证集。

2.1 数据集概述

- 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。

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
```



2.2 数据筛选

- 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。
- 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。

- 构建质量评分模型：定义了一个特征权重字典 `feature_weights`，为每个特征分配了权重。然后通过加权求和的方式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。

$$\text{quality_score} = \sum (\text{feature} \times \text{weight})$$

- 优化分层抽样：定义了一个函数 `enhanced_stratified_sampling`，用于从数据中进行分层抽样。具体步骤如下：

- 分层：基于 `sql_complexity` (SQL复杂度) 和 `question_style` (问题风格) 的组合对数据进行分层。
- 应用复杂度权重：根据SQL复杂度的权重调整质量评分，复杂度越高，权重越低。
- 动态计算抽样比例：根据每个分层的占比动态计算抽样数量，并增加20%的缓冲区，确保抽样数量足够。
- 选择高分样本：在每个分层中选择质量评分最高的样本，并将所有分层的样本合并后随机打乱，最终选择前 `n_samples` 条样本。

通过这种方式，最终从数据中筛选出5000条候选样本，存储在 `candidate_pool` 中。

- BlendSearch优化：使用 `BlendSearch` 对候选样本进行进一步优化，目标是选择一个最优的子集。具体步骤如下：
 - 定义评估函数：`evaluate_subset` 函数用于评估子集的质量，综合考虑质量评分的均值和多样性惩罚项（通过 `duplicated` 检测重复的 `question_style` 和 `sql_complexity` 组合）。
 - 定义可训练函数：`trainable` 函数用于从候选池中随机选择指定数量的样本，并调用评估函数。
 - 配置搜索空间：使用 `f1aml.tune.run` 运行 `BlendSearch`，搜索空间为子集大小 (`n_samples`)，范围在2000到5000之间，目标是最小化负质量。
 - 获取最优结果：通过 `analysis.best_config` 获取最优子集大小。
- 划分训练集和验证集：从最优子集中随机选择2000条作为训练集，从剩余数据中选择500条作为验证集，并将它们保存为CSV文件。

```

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)

```

2.3 数据筛选

按照Qwen模型训练和OmniSQL模型的要求准备所选数据。`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) , 格式为思维链\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。

```

def convert_to_qwen_format(
    sample: Dict,
    table_info_dict: Dict[str, str],
    enable_thinking: bool = True,
    include_external_knowledge: bool = True,
) -> Dict:
    """
    把一条样本转成 Qwen 所需格式
    """

    db_id = sample["db_id"]
    question = sample["question"]
    cot = sample["cot"]
    sql = sample["sql"]
    external = sample.get("external_knowledge", "")

    system_msg = {
        "role": "system",
        "content": f"Database schema: {table_info_dict.get(db_id, 'No schema available.')}",
    }

    user_prompt = f"Generate an SQLite query for the following question: {question}"
    if include_external_knowledge and external:
        user_prompt += f"\n\nExtra context: {external}"
    user_msg = {"role": "user", "content": user_prompt}

    target = f"<think>{cot}</think>\n{sql}" if enable_thinking else sql
    return {"messages": [system_msg, user_msg], "output": target}

```

3. 微调Qwen2.5-0.5B-Instruct模型

这一任务将自然语言文本转换为 SQL 查询语句及其思考过程。数据格式为JSONL，包含对话内容（messages）和输出（output）。构建了Python 脚本，该脚本旨在通过 Ray 的分布式训练框架对Qwen2.5-0.5B-Instruct模型进行微调，任务目标是将输入的自然语言转换为 SQL 查询（即Text-to-SQL任务）。训练数据使用 JSONL 格式存储，通过自回归的方式对模型进行监督微调（SFT，Supervised Fine-Tuning）。

3.1 工具函数部分

- `print_gpu_info()`

检测当前机器的 GPU 信息，并打印到日志中。如果没有可用的 GPU，提示将使用 CPU 进行训练。

功能意义：帮助用户快速确认训练资源的可用性。

- `build_prompt(example: dict)`

将输入数据中的对话上下文 `messages` 和目标输出 `output` 拼接为模型的输入提示，格式如下：

```
<system>: ...
<user>: ...
<assistant>: ...
```

功能意义：整理输入数据，使其符合 Qwen 模型的提示格式。

- `get_dataset(path: str, max_samples: Optional[int])`

从指定路径加载 JSONL 格式数据集，并通过 `build_prompt` 转化为适合训练的格式（即将数据拼接为单一文本对话）。支持选择最多加载 `max_samples` 条数据，便于调试。

功能意义：预处理输入数据，为训练提供标准化的数据集。

3.2 参数解析与准备

通过 `argparse` 模块解析训练参数，包括训练集、验证集路径、超参数（学习率、批量大小、序列长度等），以及输出目录。

- **随机种子设置：**通过 `set_seed()` 保证实验的可复现性。
- **日志记录：**将训练过程中的日志记录到控制台和文件中，便于调试与后续分析。

3.3 数据加载与处理

- **数据加载：**使用 Hugging Face 的 `datasets` 加载 JSONL 格式数据。
- **数据处理：**将对话拼接为文本后，使用 `AutoTokenizer` 对数据进行分词，并生成 `input_ids` 和 `labels`（二者相同，符合自回归任务需求）。
 - 特别注意：Qwen 官方建议**不使用 fast tokenizer**，且需要将 `pad_token` 设置为 `eos_token`。

3.4 训练函数定义

- `train_function(config)`

该函数是 Ray 分布式框架中每个工作节点的核心训练逻辑。具体步骤如下：

1. **加载模型和分词器：**从 Hugging Face Hub 加载 `Qwen/Qwen2.5-0.5B-Instruct` 模型及分词器。
2. **定义数据收集器：**使用 `DataCollatorForLanguageModeling` 对输入数据进行动态批量处理，确保训练过程高效。
3. **设置训练参数：**通过 `TrainingArguments` 配置优化器、batch size、梯度累积、fp16 精度训练、梯度检查点等。

4. **定义 Trainer**: 利用 Hugging Face 的 `Trainer` API, 将训练集、验证集和模型结合起来。
5. **启动训练**: 调用 `trainer.train()` 开始训练, 并定期保存模型到指定的输出目录。

3.5 Ray 分布式训练配置

脚本采用 Ray 的 `HuggingFaceTrainer` 作为分布式训练的框架, 通过以下方式实现分布式加速:

- `ScalingConfig` : 配置分布式训练的 GPU/CPU 数量 (如使用 4 个 GPU worker) 。
- `RunConfig` : 设置日志输出和检查点保存策略。
- `trainer_init_per_worker` : 为每个工作节点传递训练参数 (如 batch size、学习率等) 。
Ray 会在每个 worker 上独立运行 `train_function`, 从而实现高效的分布式训练。

部分代码示例:

```

def train_function(config):
    # 加载模型
    model = AutoModelForCausalLM.from_pretrained(
        "Qwen/Qwen2.5-0.5B-Instruct",
        trust_remote_code=True,
        torch_dtype=torch.bfloat16 if torch.cuda.is_available() else
    torch.float32,
    )

    tokenizer = AutoTokenizer.from_pretrained(
        "Qwen/Qwen2.5-0.5B-Instruct",
        trust_remote_code=True,
        use_fast=False,
    )
    tokenizer.pad_token = tokenizer.eos_token

    data_collator = DataCollatorForLanguageModeling(
        tokenizer=tokenizer,
        mlm=False,
    )

    from transformers import TrainingArguments, Trainer
    training_args = TrainingArguments(
        output_dir=config["output_dir"],
        num_train_epochs=config["epochs"],
        per_device_train_batch_size=config["train_bs"],
        per_device_eval_batch_size=config["eval_bs"],
        gradient_accumulation_steps=config["grad_accum"],
        learning_rate=config["lr"],
        eval_strategy="steps",
        eval_steps=100,
        save_strategy="steps",
        save_steps=100,
        logging_steps=50,
        fp16=torch.cuda.is_available(),
        report_to="none",
        dataloader_num_workers=4,
        gradient_checkpointing=True,
    )

```

4. Text-to-SQL演示

脚本基于 Gradio 构建了一个交互式 WebUI，结合 Qwen2.5-Instruct 模型，实现了支持自然语言到 SQL 查询生成（Text-to-SQL）的功能。它还能够对生成的 SQL 查询进行解释，并通过 SQLite 数据库执行这些查询，进一步展示查询结果。

4.1 SQL 执行器（SQLExecutor）

- **功能**: 封装 SQLite 数据库操作，提供以下能力：
 - 获取数据库的完整 Schema 信息 (DDL)。
 - 列出数据库中的所有表名。
 - 执行 SQL 查询并返回结果 (支持预览前 N 行)。
- **意义**: 通过与 SQLite 数据库集成，支持将生成的 SQL 直接执行，并返回查询结果，便于验证生成 SQL 的正确性。

4.2 模型加载与初始化

- **模型与分词器加载**: 通过 `transformers` 库加载 Qwen2.5-Instruct 模型和分词器。支持从 Hugging Face Hub 下载模型，也可以从本地路径加载。
- **生成配置**: 模型的最大生成长度设置为 2048 个 token，适配长文本生成需求。

4.3 SQL 生成与解释功能

(1) SQL 生成功能

- **核心逻辑**: 通过流式生成 (`_stream_generate`)，模型根据用户输入问题和数据库 Schema，逐步生成 SQL 查询。
- **提示设计**: 通过系统消息 (`system_msg`) 引导模型仅生成有效 SQL，不包含 Markdown 标记或其他多余内容。
- **清理与提取 SQL**: 生成的 SQL 会通过正则表达式精准提取，确保清除非 SQL 相关的内容。

(2) SQL 解释功能

- **核心逻辑**: 基于生成的 SQL，利用模型生成对应的语言解释。
- **语言支持**: 默认支持中文 (`zh`) 和英文 (`en`)，可根据用户需求调整解释语言。

4.4 WebUI 界面设计

- **Web 框架**: 基于 Gradio 构建，提供交互式的 Web 界面。
- **主要界面组件**：
 - **聊天窗口**: 用于显示用户问题、生成的 SQL 和 SQL 解释。
 - **SQL 生成窗口**: 用于实时显示生成的 SQL 查询。
 - **功能按键**：
 - `Submit` : 提交用户问题并启动 SQL 生成与解释。
 - `Regenerate` : 重新生成 SQL 和解释。
 - `Clear History` : 清空聊天历史和 SQL 显示窗口。

- **数据库信息展示：**如果提供了 SQLite 数据库文件，界面会显示数据库的表信息和完整 Schema。

4.5 交互功能逻辑

- **用户输入问题：**
 - 提交后，模型会根据问题和数据库 Schema 生成对应的 SQL 查询。
 - 实时流式更新生成的 SQL。
- **SQL 执行（可选）：**
 - 若提供了 SQLite 数据库，生成的 SQL 会被直接运行，返回查询结果。
- **历史记录保存：**
 - 每次生成的 SQL 和用户问题会存入历史记录，便于上下文追踪。
- **重新生成：**
 - 用户可以通过“重试”按钮重新生成 SQL 和解释，同时保留历史上下文。

界面展示：

This screenshot shows the Qwen2.5 Text-to-SQL interface. At the top, there's a logo and the text "Qwen2.5". Below it, a message says "This WebUI is based on Qwen2.5-Instruct, developed by Alibaba Cloud, modified by Siqing Li from SUSTech. (本WebUI基于Qwen2.5-Instruct打造，仅用于SUSTech大数据分析软件与应用课程的project演示。)"

The main area has a dark background with white text. On the left, there's a text input field containing a question: "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." Below this, a box contains explanatory text and numbered steps (1. 主表连接, 2. 过滤条件, 3. 分组) with corresponding bullet points.

On the right, there's a "Generated SQL" section with the following code:

```

1 SELECT
2   c.user_name,
3   COALESCE(a.author_name, 'Unknown') AS author_name,
4   AVG(similarity_score) AS avg_similarity_score
5 FROM
6   comparisons c
7 JOIN
8   authors a ON c.comparer_id = a.id
9 WHERE
10  c.completed = 1
11 GROUP BY
12  c.user_name, COALESCE(a.author_name, 'Unknown')
13 HAVING COUNT(c.document_id) > 4;

```

At the bottom, there's an "Input Question" field with placeholder text "Enter your question about the database...", and three buttons: "Clear History (清除历史)", "Submit (发送)", and "Regenerate (重试)". A note at the bottom states: "Note: This demo is governed by the original license of Qwen2.5. This demo only developed for learning propose."

部分代码展示：

```

def _stream_generate(model, tokenizer, conversation):
    """conversation: list[dict(role, content)]"""
    input_text = tokenizer.apply_chat_template(
        conversation, add_generation_prompt=True, tokenize=False
    )
    inputs = tokenizer([input_text], return_tensors="pt").to(model.device)
    streamer = TextIteratorStreamer(
        tokenizer, skip_prompt=True, timeout=60.0, skip_special_tokens=True
    )
    Thread(target=model.generate, kwargs={**inputs, "streamer": streamer}).start()
    for piece in streamer:
        yield piece

# ----- Text-to-SQL -----
def _sql_generation_stream(model, tokenizer, query, history, ddl_text):
    system_msg = {
        "role": "system",
        "content": "You are an expert in SQL. Answer ONLY with valid SQL queries. "
                   "Do not include any markdown formatting (such as ```sql or ``) and"
        do not include any additional explanations."
        f"\n\nDatabase Schema:\n{ddl_text if ddl_text else '(No database provided)'}"
    }
    conversation = [system_msg]
    for q, a in history:
        conversation.extend([
            {"role": "user", "content": q},
            {"role": "assistant", "content": a},
        ])
    conversation.append({"role": "user", "content": query})
    return _stream_generate(model, tokenizer, conversation)

```

任务二：Startup Business Plan

1. 商业计划书

一、项目概况

“智学通”是一款深度融合大语言模型（LLM）的K12智能教育辅导平台，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”的下一代在线学习基础设施。平台通过自研知识图谱、学习路径动态规划算法和多模态生成模型，为6-18岁学生提供覆盖课前预习、课中练习、课后复盘到阶段性评估的全流程数字化学习解决方案；同时引入家长、教师双端协同机制，构建

“学生—家长—教师—平台”四维联动生态。与市面上以视频课程或题库检索为主的在线教育产品不同，“智学通”强调“任务式学习”与“实时反馈”相结合——平台在学生完成任务的瞬间即可根据错误类型、耗时、思考路径等 30 余个维度生成行为画像，并调用 LLM 即时推送针对性讲解、同类拓展题、思维可视化流程图，从源头减少“重复刷题—机械记忆—短期遗忘”的低效循环。产品形态包含响应式 Web 端、iOS/Android App 及轻量级小程序，底层采用云原生 Serverless 架构与端侧推理混合部署，既保证了跨端一致体验，又显著降低了高峰期推理成本，为后续大规模用户增长预留充足弹性空间。项目愿景是：让每一名学生都能拥有一位 7×24 小时不知疲倦、永远耐心、懂他所学更懂他如何学的“数字学习伙伴”，最终实现“让学习更智能、让成长更高效”。

二、用户痛点

1. 学习效率瓶颈

传统课堂基于“平均数”展开教学，教学节奏由教师统一控制，进度快的学生缺乏进一步挑战，进度慢的学生则被迫跟跑，导致“高分重复练、低分听不懂”的普遍现象。数据表明，国内初高中生每日用于作业与补习的时间已接近 3.3 小时，但实际知识迁移率不足 35%，大量无效练习被浪费在“模板化刷题”上。

2. 个性化辅导缺位

此前流行的题库类 App 往往只能做到“拍照搜题”，解答维度局限在标准答案，无法根据学生的错误思路提供多路径讲解；线下 1 对 1 虽能补足这一缺陷，但价格高昂且师资参差。结果是多数学生在“技术层面会做题 --> 认知层面不会总结 --> 认知层面不会反思”的阶梯上止步。

3. 家庭监督低效

家长固有的“宏观把关”角色被碎片化的在线学习场景进一步削弱：

- 无法实时获知孩子在 App 里学了什么、难点在哪；
- 传统“家长群”沟通依赖老师手动反馈，滞后且主观；
- 各类打卡软件仅解决“做没做”问题，忽略“学得好不好”。

结果是家长投入了大量陪伴时间，却收效甚微，家庭冲突加剧。

4. 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，但班级人数并未减少。教师既要备课、批改，又要响应家长诉求，出现普遍的“疲劳式教学”，影响课堂创新与教学品质。

5. 家庭在教育上花销巨大

北京大学中国社会科学调查中心（ISSS）发布的中国家庭追踪调查 CFPS2010-2018 的数据显示，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%。2014-2017 年我国人均教育支出占家庭支出比重逐年上升，并在 2017 年达到 11.4%。《国家中长期教育改革和发展规划纲要》提出，2020 年要实现教育财政性经费支出占国内生产总值的

4%，而在2012年就已提前达到该水平，而近年来，国家在教育经费投入、教育固定资产投资、教育信息化投资方面均不断加大力度。

三、行业分析

1. 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》。《学生人工智能能力框架》作为人工智能战略的能力基石，主要包括建立跨学科的人工智能核心课程和学科课程群、将面向未来和本地化的人工智能作为课程载体、定制螺旋式课程顺序、营造良好的学习环境、促进教师在人工智能方面的专业发展、基于社群指导教学活动以及人工智能能力评估。《教师人工智能能力框架》定义了AI赋能教育教学的15个能力模块。这些模块旨在支持教师从了解人工智能到成长为具备丰富人工智能能力和经验的专家。

2. 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，通过应用信息科技和互联网技术对受教育者进行教学内容的传播。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%。在“内容付费”趋于饱和的背景下，“工具 + 服务”型产品复合增速超过 30%。随着 AIGC 降本增效，预计 2026 年智能辅导市场可达 1200 亿元。

3. 技术成熟度

LLM (>70B 参数) 在中文语义理解、上下文推理、多模态融合等方面的应用已较为成熟；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 8-12 个百分点；GPU 单卡推理成本在过去两年下降近 40%。技术门槛与硬件成本同步降低，为创业公司提供“弯道超车”机会。

4. 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。尚未出现一家“以 AI 为核心、用户规模破千万”的绝对龙头，市场处于新一轮窗口期。

四、核心功能与产品优势

1. 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。

2. 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 TTS + 虚拟教师 Avatar 即时生成 1-2 分钟讲解，满足听觉学习者需求。

3. 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。

4. 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。

5. 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；
- 家长可在“隐私面板”中一键删除或导出子女数据。

相较竞品，“智学通”在三大维度具备显著优势：

- A. AI 深度：自研跨学段知识图谱 + LLM 微调，答题思路更符合中小学解题逻辑；
- B. 数据驱动：精细化行为采集与因果推断模型闭环，非简单正确率打分；
- C. 多端协同：教师 SaaS 工具 + 家长仪表盘，构建完整学习生态链。

五、竞品分析

1. 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。
- 对比：智学通以 LLM“思维树”展现多策略解法，能够随学生提问实时调整；同时避免直播销售高 CAC 模式。

2. 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）。

- 对比：智学通在单一应用内完成测评、练习、复盘，减少跳转成本；家长端无广告信息流，提升信任度。

3. 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。
- 对比：智学通聚焦“工具 + 个性化辅导”，避免内容平台同质竞争；依托 Serverless 降本，保持产品高频迭代。

4. 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。
- 对比：智学通完全在线化，利用生成式模型解释逻辑推演，实现“会诊断 + 会讲题 + 会追问”闭环。

5. 未来可能威胁

苹果、华为等硬件厂商正大力布局端侧 AI，如果其自带学习助手功能，则基础答疑可能免费化。智学通以深层教学设计和数据洞见构筑壁垒，短期内难以被简单复制。

六、商业模式

1. 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%。

2. 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。

3. 收费逻辑

LLM 带来的单位服务边际成本随用户增多而趋近于 0，订阅制可最大化长期 LTV，而增值服务提供高毛利缓冲带。根据现有内测数据，付费转化率约 8%，进阶会员占比 45%，月均 ARPU 达 42 元。随着 K12 流量红利消退，我们将以“老客复购 + 渗透家庭场景”驱动营收增长，预计用户量达到 10 万时即可实现单月盈利。

七、运营现状

1. 内测数据

- 注册用户：1,000 人；
- 日活 DAU：300；
- 7 日留存：70%；
- 平均日学习时长：42 分钟，高于同类工具 30 分钟水平。

用户调研显示，最受欢迎功能为“错因标注 + AI 追问”，NPS 达 63 分。

2. 用户画像

- 年级分布：初中 55%，高中 30%，小学高段 15%；
- 地区分布：二线及以下城市 68%，验证了产品对下沉市场的吸引力；
- 支付能力：家庭月可支配教育支出 1,000–3,000 元。

3. 运营动作

- 与 3 所重点中学信息化处合作，接入校园网，进行跨端数据采集；
- 在 B 站和小红书开设“学伴 AI 课堂”账号，两个月涨粉 4.8 万；
- 组织 5 场“家长夜话”线上直播，累计观看 2.2 万次，用户拉新贡献 18%。

4. 技术迭代

- LLM 从 7B 参数升级到 13B 私有模型，中文复杂问题解答正确率提升 11%；
- 新增端侧知识点缓存，日均推理成本降低 23%；
- 完成等效 ISO/IEC 27001 信息安全管理体系建设内部审计。

八、发展规划

1. 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》。

2. 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。

3. 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。

关键里程碑：

- 2025 Q4——注册用户破 100 万，订阅收入月流水 > 3000 万；
- 2026 Q4——自研大模型通过教育部教学成果鉴定；
- 2027 Q2——赴科创板或港交所递交上市申请。

十、融资计划（暂无）

十一、联系方式

为保护信息安全，本文件不展示个人电话与微信号，可通过电子邮件 bp@zhixuetong.ai 与我们联系，或访问官方网站 www.zhixuetong.ai 获取更多资料。

项目 Slogan：让学习更智能，让成长更高效

2. 与大模型问题交互

Case 1: 信息检索不完全

Prompt:

“请详细列出2024年联合国教科文组织发布的《学生人工智能能力框架》的全部能力模块内容。”

LLM Response:

“2024年联合国教科文组织的《学生人工智能能力框架》包括以下能力模块：1. 基本AI知识；2. 数据处理；3. 简单编程；4. AI伦理。”

Expected Response:

“《学生人工智能能力框架》的能力模块包括：1. 跨学科AI核心课程；2. 基于未来和本地化的AI课程载体；3. 定制螺旋式课程顺序；4. 良好的学习环境营造；5. 促进教师的专业发展；6. 社群指导的教学活动；7. AI能力评估。”

Issue:

模型未能完整检索到所需信息，仅提供了部分内容，遗漏了框架中关键的能力模块。

Case 2: 夸大事实

Prompt:

请根据我商业计划概述，帮我写出我的项目优势。以下是我的项目概述：

“智学通”是一款深度融合大语言模型（LLM）的 K12 智能教育辅导平台，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”的下一代在线学习基础设施。平台通过自研知识图谱、学习路径动态规划算法和多模态生成模型，为 6–18 岁学生提供覆盖课前预习、课中练习、课后复盘到阶段性评估的全流程数字化学习解决方案；同时引入家长、教师双端协同机制，构建“学生—家长—教师—平台”四维联动生态。与市面上以视频课程或题库检索为主的在线教育产品不同，“智学通”强调“任务式学习”与“实时反馈”相结合——平台在学生完成任务的瞬间即可根据错误类型、耗时、思考路径等 30 余个维度生成行为画像，并调用 LLM 即时推送针对性讲解、同类拓展题、思维可视化流程图，从源头减少“重复刷题—机械记忆—短期遗忘”的低效循环。产品形态包含响应式 Web 端、iOS/Android App 及轻量级小程序，底层采用云原生 Serverless 架构与端侧推理混合部署，既保证了跨端一致体验，又显著降低了高峰期推理成本，为后续大规模用户增长预留充足弹性空间。项目愿景是：让每一名学生都能拥有一位 7×24 小时不疲倦、永远耐心、懂他所学更懂他如何学的“数字学习伙伴”，最终实现“让学习更智能、让成长更高效”。

LLM Response:

LLM (>70B 参数) 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。

Expected Response:

LLM (>70B 参数) 在中文语义理解、上下文推理、多模态融合等方面的应用已较为成熟；

Issue:

模型在生成的过程中容易夸大其词，也就是我们说的“产生幻觉”，不一定符合实际。

Case 3: 逻辑错误

Prompt:

“为什么智学通比猿辅导更适合中小学生个性化学习？”

LLM Response:

“智学通比猿辅导更适合中小学生个性化学习，因为智学通覆盖面更广，适合所有年级学生，而猿辅导只适合某些年级。”

Expected Response:

“智学通更适合中小学生个性化学习，原因有：1. 智学通通过LLM实现即时多路径解题分析，而猿辅导多为标准模板解题；2. 智学通提供知识图谱动态调整学习路径，而猿辅导缺乏这种能力；3. 智学通支持家长、教师协同机制，生态更全面。”

Issue:

模型的逻辑分析过于简单，且错误地认为猿辅导的适用范围有限。

Case 4: 幻觉过重

Prompt:

"智学通的Serverless架构如何将推理成本降低40%？详细说明。"

LLM Response:

"智学通的Serverless架构通过使用量子计算芯片，结合边缘计算节点，实现了推理成本降低40%。此外，它还利用了自适应神经网络压缩技术来优化性能。"

Expected Response:

"智学通的Serverless架构通过动态资源分配减少闲置计算资源，结合端侧推理分担云端负载，从而降低了推理成本。此外，知识点缓存进一步减少了重复推理的需求。"

Issue:

模型虚构了“量子计算芯片”和“自适应神经网络压缩技术”，这些技术在智学通中使用有点过于“小题大做。