arXiv:2603.08443v1 [cs.DB] 9 Mar 2026

大语言模型驱动的非结构化文本分析在线聚合

Chao Hui[1⋆]、Weizheng Lu[2*(✉)]、Yanjie Gao[2*]、Lingfeng Xiong[2]，Yunhai Wang[2]，和 Yueguo Chen[2]

[1] 山东大学，青岛，中国 chaohui@mail.sdu.edu.cn[2] 中国人民大学，北京，中国
{luweizheng,gaoyanjie,lenfeng2022,wang.yh,chenyueguo}@ruc.edu.cn

摘要。 大语言模型（大语言模型）在文本处理方面展现出强大能力，近期研究已通过由大语言模型驱动的语义算子增强了 SQL 和 DataFrame，以支持数据分析。然而，与关系查询相比，基于大语言模型的数据处理受限于较慢的令牌生成速度。为提升实时响应能力，我们提出 OLLA，一种大语言模型驱动的在线聚合框架，可加速关系查询中的语义处理。与仅在整个数据集处理完成后才给出结果的批处理系统不同，我们的方法将文本增量转换为结构化数据流，并应用在线聚合以提供渐进式输出。为增强在线聚合过程，我们引入语义分层抽样方法，以改进数据选择并加速向真实值收敛。评估表明，与标注真值相比，OLLA 使用不到全量数据处理时间的 4% 即可达到 1% 的准确率误差界限。通过将达到 5% 误差界所需时间与全量数据处理时间进行比较，它在不同领域实现了 1.6× 到38× 的加速比。我们在 https://github.com/olla-project/llm-online-agg.git 发布代码。

关键词： 大语言模型 · 在线聚合 · 文本处理。

1 引言

从不断增长的非结构化文本体量中提取洞见，长期以来一直是一个研究问题 [6,26]。凭借强大的语义理解能力，大语言模型（大语言模型）最近已成为非结构化文本分析的强大范式，研究人员正将基于大语言模型的文本分析集成到关系查询中，近期的例子包括 LOTUS [22] 和 UQE [8]。这些系统利用大语言模型将非结构化文本解析为结构化字段，然后对所得数据执行关系分析。这种方法使用户能够对

⋆ 贡献相同。

文本使用 SQL 或 DataFrame 等熟悉的接口。然而，将文本转换为结构化字段需要逐行处理，且大语言模型的词元生成速度显著慢于关系查询的执行速度 [17]。例如，使用大语言模型处理一组文本可能需要数分钟，而关系查询可能在数秒内完成。这种性能差距阻碍了基于大语言模型的文本分析在生产环境中的大规模采用。因此，仅仅将大语言模型集成到关系引擎中，并不足以满足低延迟文本分析的需求。

受关系分析中在线聚合的启发，该方法能够逐步细化近似结果。我们提出了一种将在线聚合与基于大语言模型的文本分析相结合的新方法。我们介绍 **OLLA**（**O**nline**L**arge **L**anguage model **A**ggregator，大语言模型在线聚合器），这是一种新颖的大语言模型驱动的在线聚合框架，旨在支持对大规模非结构化文本进行快速、交互式分析。OLLA 运用大语言模型驱动的信息转换，将非结构化数据转换为结构化数据流，然后对该数据流进行在线聚合。为确保在线聚合过程迅速收敛至真实值，OLLA 引入了一种语义分层抽样策略。它首先利用嵌入模型将非结构化文本转换到向量空间中，然后对嵌入向量进行聚类，最后对生成的簇进行抽样和调整。

我们的评估表明，与标注真值相比，OLLA 仅需不到完整数据处理时间的 4%，即可将准确度的绝对误差界控制在 1%。此外，它实现了 1.6×到 38× 的加速比；加速比通过比较达到置信区间误差界 5% 所需的时间与处理完整数据所需的时间来衡量。

综上所述，我们的贡献如下：

– OLLA 基于在线聚合的原理，能够尽早提供并逐步优化查询结果。这使得用户无需等待完整数据处理完成，即可实时获得近似洞察。– 我们提出了一种语义索引与分层采样机制：将非结构化文本转换为嵌入向量，聚类为多个层，然后从每一层中均匀抽样。这一动态过程会随时间同时提升精度与效率。–我们实现了一个原型系统，并在多种真实世界的非结构化数据集上，针对一系列代表性查询进行了评估。实验结果表明，我们的方法在响应延迟和收敛速度方面均显著优于基线方法。

2 背景与相关工作

2.1 用于数据处理的大语言模型

大语言模型（LLM）近期展现出从非结构化文本中提取并理解语义的强大能力。近期一些研究通过嵌入由大语言模型驱动的语义算子，扩展 SQL [8] 或 DataFrame [22]，以支持对非结构化数据的分析。同样，诸如

Google BigQuery[3] 和 Databricks[4] 等生产系统已将大语言模型能力直接集成到其 SQL API 中。这类查询会针对每一行调用大语言模型来处理文本列。处理大型数据集时，用户必须等待很长时间才能获得结果，而延迟也限制了这类方案在大规模场景中的实用性。

2.2 在线聚合

在线聚合能够尽早向用户提供近似结果，并随着处理更多数据而逐步细化结果 [11, 16]。与必须等到全部完成后才输出结果的批处理查询不同，在线聚合支持增量计算和交互式探索，使用户能够观察查询结果随时间的收敛情况，并在结果达到预期准确度后提前终止查询，从而显著缩短洞察时间。Ripple Joins [9]、BlinkDB [1], 和 VerdictDB [21] 等系统已证明了其在流式处理和近似查询处理场景中的有效性。

3 大语言模型驱动的在线聚合

3.1 问题定义

考虑一个包含混合属性的表 $T = \{x_i\}_{i=1}^{N}$：某些列包含非结构化文本 $\mathcal{D}_i$（例如日志、评论、对话），其余列则是数值型、类别型或其他结构化数据。用户查询 Q 将常规 SQL 与大语言模型驱动的函数 $LLM(\mathcal{D}_i)$ 相结合，然后应用聚合函数 g（例如 COUNT、AVG）。由于对每一行调用大语言模型的成本很高，我们采用增量聚合，并在每一步提供当前估算值及其 α 级置信区间。

3.2 OLLA 的工作流程

为实现快速、交互式文本分析这一愿景，OLLA 集成了若干关键组件，如图 1 所示。

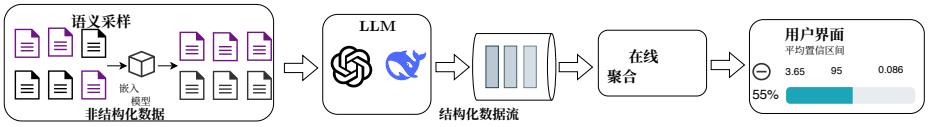


图 1：OLLA 系统架构：大语言模型模块处理非结构化数据，生成流式结构化数据，随后由在线聚合引擎对其进行增量聚合。

[3] https://cloud.google.com/blog/products/ai-machine-learning/llm-with-vertex-ai-only-using-sql-queries-in-bigquery[4] https://docs.databricks.com/aws/en/large-language-models/ai-functions

语义分层抽样。在 OLLA 中，每个文本条目都会使用嵌入模型嵌入到高维向量空间中。随后，使用 K-means 算法 [2] 对嵌入向量进行聚类。然后，我们从生成的簇中均匀抽样。由于最初基于嵌入的簇可能无法与最终的大语言模型转换结果完全一致，因此我们引入了一种自适应调整算法。该算法通过迭代调整并抽样各层，逐步对其进行优化，使之与大语言模型的输出保持一致。

将非结构化数据转换为结构化数据流。原始非结构化数据通过大语言模型转换为结构化数据流。
在线聚合。OLLA 采用在线聚合来处理逐步到达的数据流，并提供带有置信界的近似答案，从而无需等待完整计算即可及时获取洞察。

3.3 实现

我们构建了一个演示系统，以验证我们方法的可行性。在语义采样模块中，我们采用 SentenceTransformers v4.1.0 [23] 和 Faiss v1.8.0 [12] 来实现采样算法。我们使用 *all-MiniLM-L6-v2* 模型 [5] 作为嵌入模型，将文本编码为稠密向量。在非结构化到结构化转换模块中，我们部署了基于 vLLM v0.8.4 [15] 的大语言模型推理服务。为确保广泛适用性，该模块支持本地部署（自托管）和云托管部署配置。大语言模型服务的输出会持续流式传输到 Kafka v4.0.0 [14] 消息代理，后者充当文本转换模块与下游在线聚合引擎之间的缓冲层和传输层。对于在线聚合和查询处理，我们使用 Apache Spark Streaming v3.5.1 [3]。Spark Streaming 使我们能够对实时结构化数据执行复杂的转换和聚合操作，例如分组或过滤操作。它还会计算置信区间。

3.4 查询示例

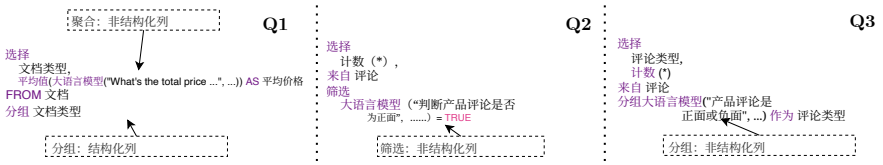


图2：三类用户查询。

如图 2 所示，我们将可通过本方法加速的用户查询分为三类：

[5] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Q1 - SELECT 子句中的大语言模型：对结构化列进行分组，并使用大语言模型对非结构化列进行聚合。对结构化列采用传统抽样。我们将此查询类型称为 SELECT。

第二季度——WHERE 子句中的大语言模型：对非结构化列进行筛选，并对结构化列进行聚合。我们将此查询类型称为 WHERE。

Q3——分组子句中的大语言模型：按非结构化列分组，并对结构化列进行聚合。我们将此查询类型称为分组。

我们将其分为三类，因为在线聚合依赖于数据抽样，而抽样方法会直接影响结果的准确度以及置信区间的收敛速度 [16]。对结构化列进行抽样时，可以采用传统技术，例如均匀抽样 [19] 或分层抽样 [13]。分组或 WHERE 子句中的非结构化列无法利用传统抽样技术，我们将在下一节第4节中讨论这些技术

4 语义抽样与统计估计

4.1 问题定义

给定文本集合 $\mathcal{D}$，我们首先通过映射函数 $E: \mathcal{D} \to \mathbb{R}^d$构建语义嵌入空间，其中每段文本都被转换为一个$d$维稠密向量。分层过程使用 K 均值聚类将嵌入空间划分为多个层。每个层 S_h 由$S_h = (\mathcal{D}_h, \mathcal{M}_h)$描述，其中 $\mathcal{D}_h \subset \mathcal{D}$ 表示该层包含的文本， $\mathcal{M}_h$ 维护层特定统计量。对于在线估计，我们使用以下公式计算置信区间：

$$\epsilon_n = \left(\frac{z_p^2 V_n}{n} \right)^{1/2} \tag{1}$$

其中， z_p 是由置信水平决定的分位数值， n 是样本量， V_n 表示方差项。 p 和 V_n的具体计算方式因过滤和分组聚合场景而异。

分层主要实现两个目标。对于过滤 场景，它有助于识别并优先处理可能包含有效样本（WHERE TRUE）的层，从而加快置信区间收敛。对于分组聚合场景，它会从语义连贯的组中进行均匀抽样。在过滤 场景中，估计量是无偏的，因为它基于具有固定层的标准分层随机抽样。在分组聚合场景中，估计量是渐近无偏的，因为动态分层调整可能引入的偏差会随着样本量增加而趋近于零。

4.2 在线过滤与语义分层抽样

在过滤场景中，我们的目标是估计满足 WHERE 子句的记录上的聚合结果。两项关键观察为我们的抽样策略提供了依据。

首先，只有有效样本，即 LLM(x) $=$ True 对应的记录，才会对最终结果有所贡献，因为不满足条件的数据会被丢弃。其次，随着收集到更多有效样本，在线聚合的置信区间会逐渐缩小，如公式 1 所述。因此，优先从更可能产生有效记录的层中进行早期抽样，可以加快置信区间的收敛。

每个层 S_h 都维护一组统计量 $\mathcal{M}_h = \{\hat{p}_h,\ \pi_h\}$ 其中 $\hat{p}_h$ 表示估计的有效记录率，π_h 表示基于该比率的采样优先级。为计算这些统计量，我们首先从整个数据集 $\mathcal{D}$中抽取一个大小为m 的初始总样本，并假设该样本能够代表整体数据分布。随后，按照各层的大小将该样本按比例分配到每个层 S_h。从层 S_h 中抽取的记录集记为 $\mathcal{R}_h$，其大小为 $m_h = |\mathcal{R}_h| = m \cdot |\mathcal{D}_h|/|\mathcal{D}|$。然后使用这个初始样本集估计有效记录率：

$$\hat{p}_h = \frac{1}{m_h} \sum_{x \in \mathcal{R}_h} \mathbb{I}[\mathrm{LLM}(x) = \mathrm{True}], \tag{2}$$

其中，$\mathbb{I}[\cdot]$ 表示指示函数。按照 $\hat{p}_h$ 从高到低对各层进行排序，以分配采样优先级 π_h，从而引导后续抽样过程优先选择更有可能产生有效记录的层。令 $v(x)$ 表示文本为 x 的记录的聚合列取值。例如，如果我们要计算文本描述符合特定条件的人群的平均年龄，那么$v(x)$将返回包含文本 x 的记录的年龄值。置信区间（公式 1）中的方差项V_n 计算如下

$$V_n = \frac{1}{n-1} \sum_{i=1}^{n} (v(x_i) - \bar{v})^2, \tag{3}$$

其中，$v(x_i)$ 是第 i 个有效样本的聚合值，$\bar{v} = \frac{1}{n}\sum_{i=1}^{n} v(x_i)$ 是它们的均值。置信水平 $p = \alpha$ 通常是预先指定的（例如 95%）。

4.3 采用语义分层抽样的在线聚合

对于文本上的分组，主要挑战在于，在进行大语言模型推理之前，组成员关系是未知的。我们的方法利用了语义相似性原理：嵌入相似的文本很可能会被分类到同一组。基于这一洞见，我们制定了一种自适应分层策略，通过抽样、记录与调整阶段迭代细化各层。

设 $\mathcal{D}$ 为一个包含 N 条记录的数据集，其中每条记录在大语言模型推理后属于 K个可能类别之一。在初始阶段（即 $t = 0$），数据集的分层通常基于先验知识或经验进行。一种常见方法是利用总样本量 N 和类别数 K来确定层数 H。例如，可以应用经验规则 $H_0 = K \log N$，$H_{\max} = 2K \log N$，其中 H_0 为初始估计的层数，$H_{\max}$ 表示预定义的最大

层数。在时刻 t，每个层 $S_h^{(t)}$ 维护一组统计量：

$$\mathcal{M}_h^{(t)} = \{m_h^{(t)}, X_{hk}^{(t)}, \hat{p}_{hk}^{(t)}, \tau_h^{(t)}, \hat{V}_h^{(t)}\}, \tag{4}$$

其中，$m_h^{(t)}$ 表示截至时间 t收集的样本数量，$X_{hk}^{(t)}$ 记录类别 k的频次，$\hat{p}_{hk}^{(t)} = X_{hk}^{(t)}/m_h^{(t)}$ 估计类别 k所占的比例，$\tau_h^{(t)} = \arg\max_k X_{hk}^{(t)}$ 用于识别主导类别，而$\hat{V}^{(t)}_h$ 用于衡量层间异质性，定义如下：

$$\hat{V}_h^{(t)} = \sum_{k=1}^{K} \hat{p}_{hk}^{(t)}(1 - \hat{p}_{hk}^{(t)}). \tag{5}$$

我们的迭代过程由三个阶段组成：抽样、记录和调整。图 3 展示了这三个阶段，我们将在下文详细讨论。

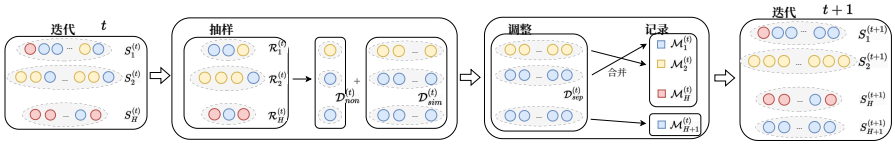


图 3：调整过程在每个层内实现同质性，并在各层之间实现异质性。

抽样。在每次迭代 t中，我们从所有层中总共抽取 $n^{(t)}$ 个样本，记作 $\{\mathcal{R}_h^{(t)}\}_{h=1}^{H}$。对于每个层 $S_h^{(t)}$，样本大小 $n_h^{(t)} = |\mathcal{R}_h^{(t)}|$由内曼分配决定：

$$n_h^{(t)} = \begin{cases} n^{(t)} \cdot \dfrac{N_h^{(t)}}{N}, & \text{if } t = 0, \\ n^{(t)} \cdot \dfrac{N_h^{(t)}\sqrt{\hat{V}_h^{(t)}}}{\sum_{i=1}^{H} N_i^{(t)}\sqrt{\hat{V}_i^{(t)}}}, & \text{if } t > 0. \end{cases} \tag{6}$$

内曼分配方法通过同时考虑层大小及其方差，从理论上最小化分层抽样的方差，确保将更多样本分配给变异性较高的层，从而提高估计值的精度和效率。 [18] 在初始迭代（$t = 0$）中，样本量按层大小的比例进行分配。在后续迭代（$t > 0$）中，分配会同时考虑层大小及其估计方差 $\hat{V}_h^{(t)}$。

记录。获得样本 $\{\mathcal{R}_h^{(t)}\}_{h=1}^{H}$后，我们将其流式传输给大语言模型进行推理。随着响应陆续返回，我们通过记录频数 $X_{hk}^{(t)}$、计算比例 $\hat{p}_{hk}^{(t)}$、识别主导类别 $\tau_h^{(t)}$ 以及测量异质性 $\hat{V}^{(t)}_h$，来更新层统计量 $\mathcal{M}_h^{(t)}$。

调整。每个层 $S_h^{(t)}$ 的归一化方差为

$$\tilde{V}_h^{(t)} = \frac{\hat{V}_h^{(t)}}{1 - 1/K}. \tag{7}$$

此处，分母 $1 - 1/K$ 对应于均匀类别分布下的理论最大方差，用作归一化因子。若归一化方差超过阈值 θ，我们便触发细化过程，首先识别异质群组，以提高层同质性：

$$\mathcal{D}_{non}^{(t)} = \{x \in \mathcal{R}_h^{(t)} \mid \text{label}(x) \neq \tau_h^{(t)}\}, \tag{8}$$

然后定位语义相似的未采样记录：

$$\mathcal{D}_{sim}^{(t)} = \{x \in \mathcal{D}_h^{(t)} \backslash \mathcal{R}_h^{(t)} \mid \exists y \in \mathcal{D}_{non}^{(t)} : \text{sim}(E(x), E(y)) > \gamma\}, \tag{9}$$

其中 $\text{sim}(u, v)$ 表示嵌入向量 u 与 v之间的余弦相似度。这两个集合合并为 $\mathcal{D}_{sep}^{(t)} = \mathcal{D}_{non}^{(t)} \cup \mathcal{D}_{sim}^{(t)}$，代表可能被错误聚类或需要重新分配的离群样本。根据现有层的数量，我们要么拆分，要么合并 $\mathcal{D}_{sep}^{(t)}$。当 $H < H_{\max}$时，我们从 $\mathcal{D}_{sep}^{(t)}$ 创建一个新层。否则，我们将 $\mathcal{D}_{sep}^{(t)}$ 与共享相同主导类别的现有层合并。目标层 h^* 根据最大嵌入相似度进行选择：

$$h^* = \arg\max_{h'} \{\text{sim}(E(\mathcal{D}_{sep}^{(t)}), E(\mathcal{D}_{h'})) \mid \tau_{h'}^{(t)} = \tau_{\mathcal{D}_{sep}^{(t)}}\}, \tag{10}$$

其中 $E(\mathcal{D})$表示集合 $\mathcal{D}$中所有文本的平均嵌入向量。每次迭代后，我们计算公式 1 的方差项 V_n 和置信水平 p：

$$V_n = \sum_{h=1}^{H} \left(\frac{N_h^{(t)}}{N} \right)^2 f_h^{(t)} \hat{V}_h^{(t)}, \tag{11}$$

其中，$f_h^{(t)} = \frac{N_h^{(t)} - m_h^{(t)}}{m_h^{(t)} \, N_h^{(t)} - 1}()$ 是由于每个层内不放回抽样而产生的超几何分布有限总体校正项，而 $p = \alpha/K$ 则用于校正跨 K 个类别的多重比较。

这一迭代过程会持续进行，直到收敛（所有层均达到足够的纯度（$\tilde{V}_{(t)}$ $_h \leq \theta$））， 每轮循环都会提高层纯度（$V_{(t+1)} < V_{(t)}$），同时保持有界方差（$0 \leq V_h \leq 1 - \frac{1}{K}$）。完整流程总结见算法 1。 —

5 评估

5.1 实验设置

数据集。 我们使用一组多样化的数据集评估系统，涵盖产品评论、文档理解和文本分类。由于

算法 1 基于语义分层抽样的在线聚合

要求： 数据集 $\mathcal{D}$，大语言模型函数 LLM，嵌入函数 E，初始层数量 H_0，最大层数 $H_{\max}$，置信水平 α，方差阈值 θ，相似度阈值 γ

对于 每个数据点 $x \in \mathcal{D}$ **执行**
  计算嵌入向量 $e_x \leftarrow E(x)$
结束循环
将嵌入 $\{e_x\}$ 划分为 H 层：$\{S_1^{(0)}, \ldots, S_{H_0}^{(0)}\}$
$t \leftarrow 0$
当 $\exists h$： $\tilde{V}_h^{(t)} > \theta$ **执行**
  如果 $t = 0$ **那么**
    $n_h^{(t)} \leftarrow n^{(t)} \cdot N_h^{(t)}/N$ 对于每个层 $S_h^{(t)}$
  else $\leftarrow n^{(t)} \cdot N_h^{(t)} \sqrt{\hat{V}_h^{(t}} / \sum_{i=1}^{H} N_i^{(t)} \sqrt{)\hat{V}^{(t)}}$
    $n_h^{(t)}{}_i$
  结束如果
  从每个层抽取大小为 $n_h^{(t)}$ 的样本 $\mathcal{R}_h^{(t)}$
  对于每个样本 $x \in \mathcal{R}_h^{(t)}$ **执行**
    获取类别标签 $k \leftarrow$大语言模型(x)
    更新 $\mathcal{M}_h^{(t)} \leftarrow \{m_h^{(t)}, X_{hk}^{(t)}, \hat{p}_{hk}^{(t)}, \tau_h^{(t)}, \hat{V}^{(t)h}\}$
  结束循环
  对于每个层 h 具有 $\tilde{V}^{(t)}{}_h > \theta$ **执行**
    获取 $\mathcal{D}^{(t)}{}_{sep} \leftarrow \mathcal{D}_{non}^{(t)} \cup \mathcal{D}_{sim}^{(t)}$
    如果 $H < H_{\max}$ **then**
      从 $\mathcal{D}^{(t)}$创建新层 $_{sep}$
    else
      将 $\mathcal{D}_{sep}^{(t)}$ 与最相似的现有层合并 S_{h^*}
    结束 if
  结束 for
  使用 V_n 和 $p = \alpha/K$更新置信区间
  $t \leftarrow t + 1$
结束循环
返回 比例估计 $\{\hat{p}_{hk}^{(t)}\}$ 和置信区间

缺乏针对大语言模型查询的标准基准，我们从多种来源整理了真实世界数据集，并针对这些数据集设计了具有代表性的查询。表1展示了数据集的描述、大小以及针对它们的查询。

指标。 我们采用以下统计指标：绝对误差、置信区间和累计有效结果。实验中报告的所有时间均指每次查询的平均耗时。渐进式流输出与标注真值之间的绝对误差 可视为准确度。置信区间 按照公式 1 计算，其中置信水平 p 和方差项 V_n 的具体计算方式因过滤场景和分组聚合场景而异，详见第4节。累计有效结果 指满足预定义有效性约束（例如语法正确性）的输出总数。

表1：实验中使用的数据集和查询。大小 列（n_{rows}）表示所使用的行数。查询类型 列对应于图2和第3.4节所示的三类，即分别作为选择、FILTER 或分组子句中的大语言模型算子。

数据集	描述	Size (n_{rows})	查询类型
公司文档 [7]	一个包含商业文档的文档数据集如发票、采购订单。	10,000	Q1
BBC 新闻 [5]	A 多类别新闻分类 数据集 with 合成添加的数值字段（例如，浏览量），用于聚合。	15,000	第二季度, 第三季度
arXiv [25]	包含 arXiv 研究论文的元数据，包括标题和摘要。	10,000	Q2
亚马逊产品 [10]	产品评论和元数据 ，带有情感分布，并针对分析保持平衡。对于我们的实验-中，我们创建了一个平衡数据集，其中正面评论与负面评论数量相等。	15,000	Q2
电影 [20]	来自评论家的电影评论集，配有评分等元数据。	15,000	第二季度, 第三季度
中文简历 [24]	一组中文简历，其中自我-介绍部分包含个人信息例如姓名和年龄。	20,000	Q1

环境。 我们的实验在一台配备 2 颗 Intel Xeon 6438M 中央处理器（2.2 GHz，60 MB 缓存）、512 GB 主存和 8 个 NVIDIA A800 GPU（80 GB 高带宽内存）的机器上进行，操作系统为 Rocky Linux 8.9。

5.2 OLLA 的准确度

我们从两个维度评估 OLLA 的准确度：(1) 大语言模型将非结构化数据转换为结构化数据的正确性，以及 (2) 与全标注数据集中的真实值相比，流式聚合结果的精度。

表 2：*BBC* 新闻 分类和电影 情感分析的准确度。

模型	BBC 新闻	电影
Qwen2.5 7B	0.90	0.84
DeepSeek-V3.1 671B	0.95	0.88
真实值	1.00	1.00

表 2 比较了模型在 *BBC* 新闻 和 电影 数据集上的准确度与其真实标签的一致性。结果表明，大语言模型的表现取决于模型和上下文，其输出并非与预定义标签完全一致。DeepSeek-V3.1 671B（无思考）模型表现最佳，但推理成本较高。体积小得多的紧凑型 Qwen2.5-7B 模型则是颇具吸引力且经济高效的替代方案，其准确度超过 80%，甚至达到 90%。

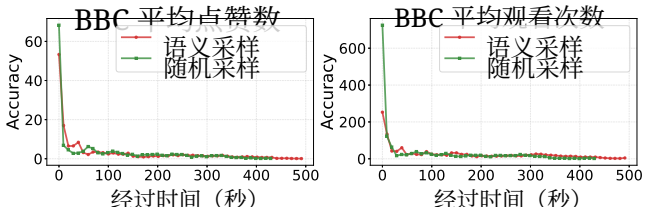


图4：在线聚合中准确度随时间的收敛。准确度：流式处理聚合函数结果与真实值之间的绝对误差。左：*BBC* 新闻的平均点赞_数 。右：*BBC* 新闻的平均浏览_量 。

我们将在线聚合结果与完整数据集的真实值进行比较，以直接衡量 OLLA 的实测表现，而非仅依赖其置信区间估计值。图 4 绘制了流式聚合结果与基于整个数据集计算出的真实值之间的绝对误差，显示在线聚合过程能够非常迅速地收敛。在我们的两种设置中，OLLA 仅需处理完整数据集时间的 2.43% 和 3.49%，即可达到与最终标注真实值相比的 1% 误差界。该图展示了在线聚合中的基本权衡：随着执行时间（x 轴）增加，误差会降低。此外，我们的语义采样方法比随机采样更快地降低误差，尤其是在聚合过程的初始阶段。

5.3 OLLA的效率

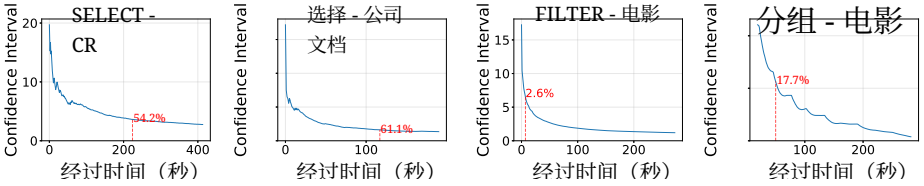


图5：代表性查询类型的置信区间收敛。红色百分比是我们的方法达到5%误差界所需时间除以批处理总执行时间。从左到右：选择：中文简历（CR）上简历的平均年龄 ；选择：公司文档 （CD）上类型发票 的平均总_价 ；筛选：正面电影 评论的平均长度；分组：中性电影评论的比例。

为验证OLLA的效率，我们针对图2所示的三种代表性查询类型开展实验。在每种情况下，我们随着抽样的推进跟踪置信区间的变化。该设置反映了一种常见用例：用户希望尽早获得并逐步细化结果，而无需等待完整数据集处理完成。

图 5 展示了每个查询的置信区间轨迹。我们进一步评估达到 5% 误差界所需的总执行时间占比。该比例约为总执行时间的 61.1%（1.6×，与扫描全部数据相比）

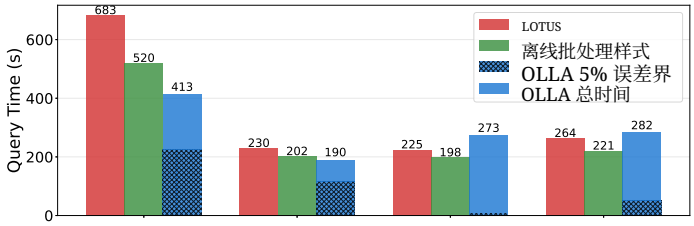


图 6：图中展示了 LOTUS、批处理基线和 OLLA 在不同查询类型下的查询执行时间对比。OLLA 柱状条上的斜线部分表示达到 5% 误差界所需的时间，完整柱状条则表示其总执行时间。

其中，SELECT 场景为 2.6%（38×），WHERE 场景为 17.7%（5.6×），GROUP BY 场景为 17.7%（5.6×）。

我们进一步将 OLLA 与最先进的系统 LOTUS [22],以及通过并行随机抽样处理全部数据的朴素离线基线进行了比较分析。图 6 所示结果凸显了我们方法的权衡与核心优势。对于 SELECT 查询，OLLA 显著降低了延迟。对于 WHERE 和分组任务，尽管我们的语义分层抽样会带来少量延迟开销，但 OLLA 的主要优势在于其在线特性。达到 5% 误差界所需的时间，远短于等待任何离线批处理方法完成所需的时间。

5.4 抽样优化的效率

基于语义抽样的在线过滤。 我们在两个数据集上评估基于语义抽样的在线 FILTER 的有效性：亚马逊产品和*BBC* 新闻。在每项实验中，我们根据在线聚合过程中信息丰富的样本累积速度以及置信区间的收敛速度，将我们的方法与随机抽样基线进行比较。

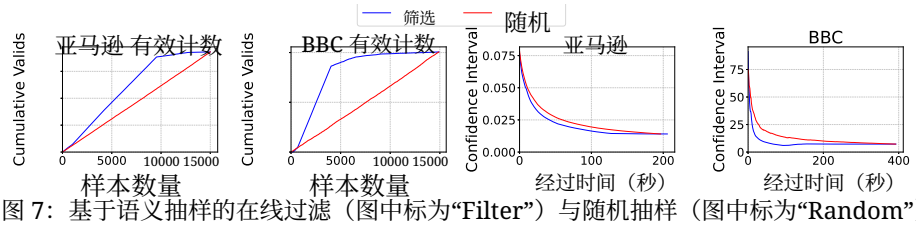


图 7：基于语义抽样的在线过滤（图中标为“Filter”）与随机抽样（图中标为“Random”）的比较。左图：在亚马逊产品中，被大语言模型判定为正面的评论的已验证_购买 率；右图：在*BBC* 新闻中，被大语言模型识别为体育相关的新闻文章的平均点赞 数。

结果如图 7 所示。上方两幅图展示了在抽样步骤推进过程中，经大语言模型评估且满足谓词（即标记为 TRUE）的条目累计计数。我们的过滤感知策略在抽样早期阶段始终能够更快地累积有效样本。下方两幅图展示了

相应目标聚合量置信区间的宽度，表明我们的方法达到目标误差界的速度显著快于基线方法。具体而言，对于 亚马逊 数据集，过滤采样在获取完整抽样过程的 5.28% 后即可达到 5% 的误差界，而随机采样则需要 6.44%。同样，对于 *BBC* 新闻，过滤采样在完成抽样过程的 2.55% 时即可达到该误差界，而随机采样则需要 6.04%；这表明我们的方法能够以低得多的抽样成本提供更紧致的估计值，从而在实践中减少大语言模型调用次数。

基于语义分层抽样的分组聚合。我们进一步评估语义分层抽样在基于分组的聚合场景中的性能，其目标是估计大语言模型识别出的不同组的比例或总量。我们针对两项分类任务开展实验：按情感对电影 条目进行分类，以及按主题对 *BBC* 新闻 文章进行分类。语义采样以迭代方式进行，并且每完成总采样步骤的 10%，就动态调整一次层。我们的基线方法包括标准随机采样，以及 UQE [8] 提出的静态分层（即迭代过程中不调整层）。

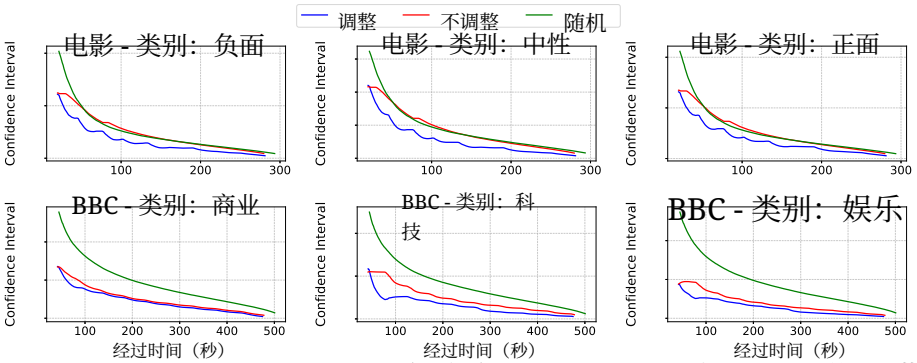


图 8：不同采样策略下组比例估计的置信区间变化。图例说明：“调整”——采用第 4.3 节提出的采样-记录-调整工作流；“不调整”——仅基于嵌入向量进行采样，不调整 UQE [8] 提出的聚类层；“随机”——通过简单随机选择抽取样本。上图：电影 情感（正面、中性、负面）的比例；下图：*BBC* 新闻 主题（商业、科技、娱乐）的比例。

图8展示了两个数据集的结果。在所有情况下，调整策略——我们的动态分层抽样方法——相比基线方法，始终能更快地实现置信区间收敛。在电影 数据集中，静态分层（不调整）的表现与随机采样相近，这表明初始聚类未能与语义标签很好地对齐。相比之下，*BBC* 新闻 数据集表明，即使不进行调整，分层抽样仍优于随机采样，说明初始聚类对齐效果更好。为了定量评估不同采样

策略的效率，我们报告了达到5%误差界所需时间占全数据处理时间的平均比例，该比例是在每个数据集的所有目标类别上取平均得到的。对于电影 数据集，调整 方法平均只需全程时间的13.75%即可达到这一阈值，在所有类别上均显著优于不调整 （23.71%）和随机 （21.80%）。同样，在*BBC* 新闻 实验中，调整、不调整和随机所需的平均时间比例分别为10.82%、14.58%和39.40%。

5.5 超参数的影响

本节研究关键超参数对 OLLA 性能的影响，旨在为选择合适的超参数设置提供实证指导，从而最大限度地提高估计准确度。

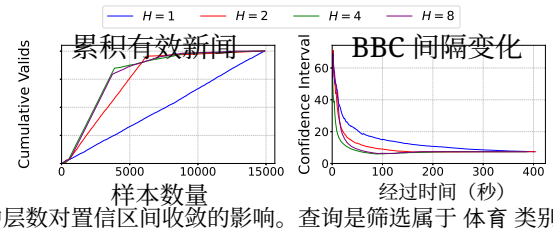


图 9：过滤场景中层数对置信区间收敛的影响。查询是筛选属于 体育 类别的 *BBC* 新闻 条目。

对于过滤场景，我们研究了层数对置信区间收敛的影响。图 9 所示的结果表明，即使只有两层，语义抽样也已能显著加快收敛。这一改进可归因于 *BBC* 新闻 数据集具有很强的语义可分性，使聚类算法能够有效地区分不同的语义组。此外，随着层数增加到四层或五层（这与数据集中存在的五个主要新闻类别相对应），收敛进一步加快，从而以更少的抽样量获得更准确的估计值。

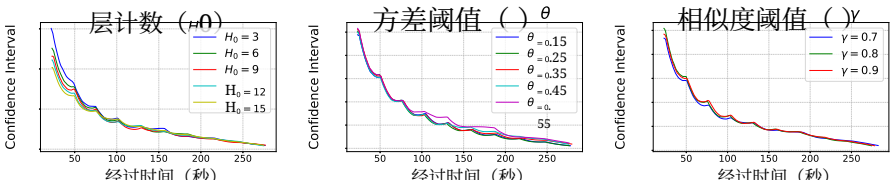


图 10：超参数设置对分组场景中收敛的影响。该查询旨在计算电影 数据集中不同类别（正面、负面和中性）的比例。

对于分组聚合场景，我们进一步研究了关键超参数对收敛行为的影响。该实验基于前文所述的 电影数据集，目标是计算每个语义组所占的比例。图10展示的结果表明，采用更多的初始层通常有助于在抽样早期阶段实现更快的收敛。这一现象可以通过

更多层所提供的增强语义分辨率来解释，因为这能够实现更准确的分组估计。图10中的第二幅图展示了方差阈值对收敛的影响：当该阈值设置得过高时（例如 0.55），可能会阻碍形成更细粒度的层，从而减缓收敛速度。相比之下，第三幅图表明，相似度阈值对整体收敛性能的影响相对较小。

5.6 可扩展性

我们通过增加大语言模型服务实例的数量来评估系统的可扩展性，以考察性能的提升情况。我们使用 LiteLLM [4] 作为服务层部署系统，在硬件规格相同的条件下，将实例数量从 1 扩展到 4，并测量在两个具有代表性的工作负载——亚马逊产品 和 *BBC* 新闻上的过滤和聚合——中达到相同误差界所需的执行时间。图 11 展示了大语言模型服务实例数量与执行时间之间的关系。如图所示，两个工作负载均受益于扩展，但随着实例数量的增加，额外收益逐渐减少。在使用 2、3 和 4 个服务实例时，对于亚马逊产品 工作负载，我们观察到的加速比分别为 1.32×、1.46× 和 1.52× ；对于*BBC* 新闻 工作负载，加速比分别为更具可扩展性的 1.52×、1.83× 和 2.03× 。这些结果证实了我们的架构能够有效扩展。其低于线性的趋势主要源于 LiteLLM 的默认路由和批处理配置尚未针对我们的工作负载进行调优。经过优化设置后，可扩展性有望接近线性理想状态。

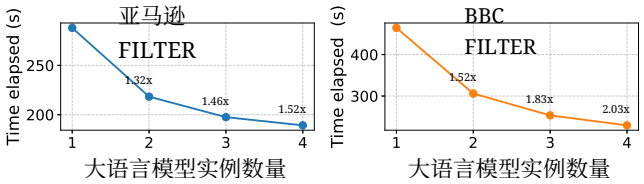


图 11：系统可扩展性评估。

6 结论

本文提出了 OLLA，这是一种将大语言模型与在线聚合相结合的新型框架，旨在实现对非结构化数据的可扩展交互式分析。OLLA 通过基于大语言模型的提取将非结构化输入转换为结构化表示，将其嵌入以进行语义索引，并通过在线聚合应用近似查询处理，从而应对延迟、可扩展性和语义歧义等长期存在的挑战。来自多个不同领域的实验结果表明，与现有方法相比，OLLA 能够缩短洞察时间，并支持更高质量的探索。

致谢。本研究得到中央高校基本科研业务费专项资金及中国人民大学科研经费的支持

(24XNKJ22)、北京市自然科学基金（L247027）和国家自然科学基金（No.62272466、No.U2436209 和 No.U24A20233）的资助。计算资源由中国人民大学PCC提供支持。

参考文献

1. Agarwal, S., Mozafari, B., Panda, A., Milner, H., Madden, S., Stoica, I.: BlinkDB：在超大规模数据上实现有界误差和有界响应时间的查询。载于：EuroSys 13。第 29–42 页（2013） 2. Ahmed, M., Seraj, R., Islam, S.M.S.: K-means 算法：全面综述与性能评估。《Electronics》**9**(8), 1295（2020） 3. Armbrust, M., Das, T., Torres, J., Yavuz, B., Zhu, S., Xin, R., Ghodsi, A., Stoica, I., Zaharia, M.: Apache Spark 中面向实时应用的结构化流处理：一种声明式API。载于：SIGMOD 18。第 601–613 页（2018） 4. BerriAI：LiteLLM。https://github.com/BerriAI/litellm （ 2024 ）， 访问日期：2025- 06-16 5. Bose, B.: BBC 新闻分类。https://kaggle.com/competitions/learn-ai-bbc（2019）， Kaggle 6. Chen, X., Jin, L., Zhu, Y., Luo, C., Wang, T.: 自然场景文本识别：综述。《ACM Computing Surveys》**54**(2)（2021） 7. Cherguelain, A.: 公司文档数据集。Kaggle（2024）， https://www.kaggle.com/datasets/ayoubcherguelaine/company-documents-dataset 8. Dai, H., Wang, B., Wan, X., Dai, B., Yang, S., Nova, A., Yin, P., Phothilimthana, M., Sutton, C., Schuurmans, D.: UQE：非结构化数据库的查询引擎。载于：NeurIPS 24。第 37 卷, 第 29807–29838 页（2024） 9. Haas, P.J., Hellerstein, J.M.: 用于在线聚合的 Ripple Joins。《SIGMOD Record》**28**(2), 287–298（1999） 10. He, R., McAuley, J.: 起起落落：利用单类协同过滤对时尚趋势的视觉演化建模。载于：WWW 16。第 507–517 页（2016） 11. Hellerstein, J.M., Haas, P.J., Wang, H.J.: 在线聚合。载于：SIGMOD 97。第 171–182 页（1997） 12. Johnson, J., Douze, M., Jégou, H.: 利用 GPU 进行十亿规模相似性搜索。《IEEE Transactions on Big Data》**7**(3), 535–547（2019） 13. Joshi, S., Jermaine, C.: 面向低选择率查询的稳健分层抽样方案。载于：ICDE 08。第 199–208 页。IEEE（2008） 14. Kreps, J., Narkhede, N., Rao, J., 等：Kafka：用于日志处理的分布式消息系统。载于：NetDB 11。第 1–7 页（2011） 15. Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C.H., Gonzalez, J., Zhang, H., Stoica, I.: 使用 PagedAttention 优化大语言模型服务的内存管理。载于：SOSP 23。第 611–626 页（2023） 16. Li, Y., Wen, Y., Yuan, X.: 在线聚合：综述。载于：Web Information Systems and Applications。第 103–114 页（2018） 17. Liu, S., Biswal, A., Cheng, A., Mo, X., Cao, S., Gonzalez, J.E., Stoica, I., Zaharia, M.: 优化关系型工作负载中的大语言模型查询。载于：MLSys 25（2025） 18. Neyman, J.: 论代表性方法的两个不同方面：分层抽样法与目的抽样法。载于：《统计学突破：方法论与分布》，第 123–150 页。Springer（1992）

19. Olken, F., Rotem, D.: 数据库随机采样：综述。Statistics and Computing 5(1), 25–42（1995） 20. Pang, B., Lee, L.: 看星星：利用类别关系，针对评分量表进行情感分类。arXiv 预印本 cs/0506075（2005） 21. Park, Y., Mozafari, B., Sorenson, J., Wang, J.: VerdictDB：实现近似查询处理的通用化。载于：SIGMOD 18。第 1461–1476 页（2018） 22. Patel, L., Jha, S., Pan, M., Gupta, H., Asawa, P., Guestrin, C., Zaharia, M.: 语义算子及其优化：迈向具有准确性保证的基于人工智能的数据分析。PVLDB 18(11), 4171–4184（2025） 23. Reimers, N., Gurevych, I.: Sentence-BERT：使用孪生 BERT 网络生成句子嵌入。载于：EMNLP 19（2019 年 11 月） 24. Su, Y., Zhang, J., Lu, J.: 简历语料库：用于信息抽取系统研究的大型数据集。载于：CIS 19。第 375–378 页（2019） 25. arXiv.org 投稿者：arXiv 数据集（2024），https://www.kaggle.com/dsv/ 7548853 26. Wankhade, M., Rao, A.C.S., Kulkarni, C.: 情感分析方法、应用与挑战综述。Artificial Intelligence Review 55(7), 5731–5780（2022）