Rejoinder to the Discussions on "LAMBDA: A Large Model Based Data Agent"

Maojun Sun[b], Ruijian Han[b], Binyan Jiang[b],

Houduo Qi[a,b], Defeng Sun[a], Yancheng Yuan[a*] and Jian Huang[a,b*]

[a]Department of Applied Mathematics, The Hong Kong Polytechnic University

[b]Department of Data Science and Artificial Intelligence, The Hong Kong Polytechnic University

February 10, 2026

We thank the editor, Professor Hongtu Zhu, for organizing the discussion and the distinguished discussants for their insightful comments and stimulating questionson our work, "LAMBDA: Large Model Based Data Agent" (Sun et al., 2025a). In this rejoinder, we synthesize the main themes raised, clarify key design choices, and outline concrete plans to address the comments and suggestions. Specifically, we consider domain-knowledge integration, benchmark for data agents, reasoning and planning, collaborating agents, human–AI collaboration and supporting infrastructure. Where applicable, we report preliminary progress and highlight open challenges that will shape our ongoing research agenda.

1 Strengthening Domain Knowledge Integration

Knowledge integration is a core component of LAMBDA and a key driver of its capabilities. We have demonstrated its empirical effectiveness through examples. Following suggestions from Dr. Sammi Tang, Dr. Xuewei Wang, Professor Fan Zhou, and Professor Bang Liu, we plan several enhancements. First, we will refine the integration pipeline, including

*Corresponding authors.

the choice of embedding model, the similarity metric, and the thresholding strategy, as these directly affect retrieval accuracy and downstream analysis. Second, we will address a current limitation: retrieval relies solely on user-provided natural language instructions and is not conditioned on the dataset itself. Third, we will develop a dynamic, continuously updated knowledge base to further improve LAMBDA's performance over time.

1.1 Retrieval Improvement via Contrastive Fine-Tuning

Inspired by contrastive learning and dual-encoder (two-tower) architectures, we explore two methods to improve the embedding model's retrieval accuracy: *symmetric fine-tuning*, which fine-tunes the encoder to process both queries and knowledge entries, and *asymmetric alignment*, which aligns the embedding model's representation space with that of a downstream task-oriented large language model (LLM).

Problem Formulation. Let $\mathcal{Q} = \{q_1, \ldots, q_N\}$ be a set of queries and $\mathcal{D} = \{d_1, \ldots, d_N\}$ be the corresponding ground-truth relevant document. $\{(q_i, d_i)\}_{i=1}^{N}$ denotes a collection of positive knowledge pairs, and all other irrelevant combinations are considered negative pairs. We aim to learn a shared latent semantic space where positive knowledge pairs are closer than negative ones. The semantic similarity between a query q_i and a knowledge entry d_i is quantified by the cosine similarity score of their embeddings $\mathbf{e}_{q_i}, \mathbf{e}_{d_i} \in \mathbb{R}^m$: $s(q_i, d_i) = \frac{\mathbf{e}_{q_i}^\top \mathbf{e}_{d_i}}{\|\mathbf{e}_{q_i}\| \|\mathbf{e}_{d_i}\|}$. To optimize the parameters, we minimize the InfoNCE (Information Noise Contrastive Estimation) loss with in-batch negatives. For a batch of size B, the objective is defined as: $\mathcal{L} = -\frac{1}{B} \sum_{i=1}^{B} \log \frac{\exp(s(q_i, d_i)/\tau)}{\sum_{j=1}^{B} \exp(s(q_i, d_j)/\tau)}$, where τ is a temperature hyperparameter. We utilize *symmetric fine-tuning* and *asymmetric alignment* to generate these embeddings.

Method 1: Symmetric Encoder Fine-Tuning. In this approach, we utilize a single encoder model $f_\theta(\cdot)$ (e.g., `all-MiniLM-L6-v2`), parameterized by θ, to generate dense vector representations for both input types. The embeddings are computed as: $\mathbf{e}_{q_i} = f_\theta(q_i)$ and $\mathbf{e}_{d_i} = f_\theta(d_i)$. This method fine-tunes the shared parameters θ to directly minimize $\mathcal{L}$, pulling positive pairs closer while pushing away in-batch negatives.

Method 2: Asymmetric Alignment. In the second approach, we employ an asymmetric architecture to transfer the reasoning capabilities of an LLM into the retrieval task. We utilize two distinct encoders: a fixed query encoder $g_\phi(\cdot)$ (using `Qwen3-4B-Instruct`) and a trainable knowledge encoder $h_\psi(\cdot)$ (using `all-MiniLM-L6-v2`). To address the dimension mismatch between the LLM output (k) and the embedding model (m), we introduce a learnable linear projection matrix $\mathbf{W} \in \mathbb{R}^{m \times k}$. The embeddings are computed as: $\mathbf{e}_{q_i} = \mathbf{W} g_\phi(q_i)$ and $\mathbf{e}_{d_i} = h_\psi(d_i)$. Here, $g_\phi(q_i)$ represents the aggregated hidden state from the LLM. During training, we freeze ϕ and update only the projection matrix $\mathbf{W}$ and the encoder parameters ψ to minimize $\mathcal{L}$.

Experimental Results. After training for 100 epochs, during which the model loss had already stabilized, we evaluated the models on the test set using the Hit Rate metric. The Hit Rate (HR) at rank k is defined as: $\text{HitRate@}k = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\{\text{rank}(y_i) \leq k\}$, where N is the number of queries, y_i denotes the ground-truth knowledge for query i, and $\mathbf{1}(\cdot)$ is the indicator function. The performance of the original `all-MiniLM-L6-v2` model is used as the baseline. The experimental results are summarized in Table 1.

We observe that the fine-tuned `all-MiniLM-L6-v2` model achieves a substantial improvement over the baseline, especially on HR@1, with a gain of 3.72%. Moreover, the model aligned with `Qwen3-4B-Instruct` demonstrates greater improvements after fine-tuning: a 26.2% gain over the baseline and a 23% improvement compared with directly

Table 1: Hit rate of different approaches. Here, all-MiniLM-L6-v2 denotes the embedding model from the original paper, FT all-MiniLM-L6-v2 is the symmetrically fine-tuned version, and Qwen3-4B Alignment represents the asymmetric fine-tuning approach.

Hit Rate @ K	**all-MiniLM-L6-v2**	**FT all-MiniLM-L6-v2**	**Qwen3-4B Alignment**
HR@1	45.28%	49.00%	72.00%
HR@2	57.13%	58.50%	81.50%
HR@3	61.45%	63.50%	83.00%
HR@4	65.66%	65.00%	84.00%
HR@5	67.77%	67.50%	84.50%
HR@6	69.58%	68.50%	86.00%
HR@7	70.98%	70.00%	87.00%
HR@8	71.49%	72.00%	87.50%
HR@9	72.49%	73.00%	88.50%
HR@10	73.09%	73.50%	89.50%

fine-tuning the embedding model alone.

It is worth noting that in our contrastive learning setup, only the top-1 knowledge entry is treated as a positive sample, while all others are negatives. This encourages the model to focus primarily on rank-1 relevance, resulting in a larger gain for HR@1 and comparatively smaller improvements for higher ranks.

Overall, these experiments show that continued fine-tuning on a knowledge base can effectively enhance retrieval accuracy in future applications. In the future, we may further enhance retrieval accuracy and strengthen human involvement by proactively recommending the top-K candidate knowledge items after retrieval, allowing users to select the most appropriate ones.

1.2 Alternative Similarity Metrics

Recent work suggests that cosine similarity is not universally optimal for assessing alignment quality, particularly when embedding magnitudes carry semantic information or when training-time normalization introduces artifacts (Steck et al., 2024). To examine this empirically, we evaluate our retrieval system under multiple similarity measures on the datasets used above. We consider vector-space metrics (cosine similarity, Euclidean distance) and set-based lexical metrics (Jaccard similarity, overlap coefficient). For the set-based metrics, we compute scores at both the word level and over character-level 3-grams. Consistent with our prior experiments, we use `all-MiniLM-L6-v2` as the embedding model for vector-space metrics.

A brief definition of the core set-based metrics for two sets S_A and S_B (derived from query and knowledge text) is provided below:

Jaccard Similarity measures the intersection over union of two sets, that is, $J(S_A, S_B) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|}$. **Overlap Coefficient** measures the size of the intersection relative to the size of the smaller set, effectively handling length imbalance: $\text{Overlap}(S_A, S_B) = \frac{|S_A \cap S_B|}{\min(|S_A|,|S_B|)}$.

We employ Hit Rate (HR@K) and Mean Reciprocal Rank (MRR@10) to evaluate retrieval accuracy. The comparative results are illustrated in Figure 2. As shown in Figure 2, the two vector-based metrics yield comparable performance and substantially outperform lexical matching. All two vector metrics (ℓ_2, and Cosine) achieve a Hit Rate@10 exceeding 70%, whereas the best lexical metric (Jaccard on character 3-grams) reaches only roughly 51%, with word-level metrics failing to surpass 20%. This confirms that the semantic gap between user instructions and code descriptions requires dense representation rather than keyword overlap.

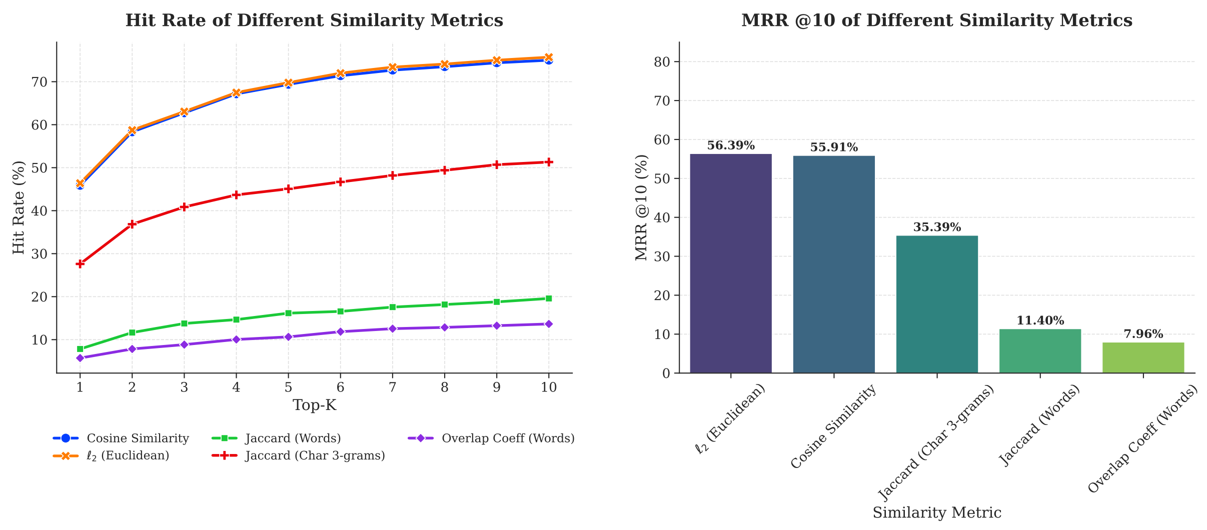

Figure 1: Hit Rate and MRR@10 across different similarity metrics. Vector-based methods (left group) significantly outperform lexical methods (right group).

As pointed out by the MRR experiment, the results mirror the Hit Rate findings; the vector-based metrics perform similarly and consistently outperform lexical matching. ℓ_2 Euclidean distance achieves the highest retrieval performance with an MRR@10 of 56.39%, followed closely by Cosine Similarity (55.91%). The marginal performance gap ($< 0.5\%$) indicates that for the `all-MiniLM-L6-v2` model, the retrieval quality is stable across these standard vector metrics. Consequently, while switching to ℓ_2 distance may offer a slight improvement, the primary driver of performance remains the quality of the embedding space itself rather than the specific distance function employed.

1.3 Threshold, Mode and Quality Control

Dr. Sammi Tang and Dr. Xuewei Wang also mentioned that relying on a fixed threshold (default $\theta = 0.2$) is a heuristic that may not generalize across all domain-specific tasks. We agree that a fixed threshold risks either retrieving irrelevant code snippets (introducing noise) or missing critical context when the semantic overlap is subtle. To address this, the future versions of LAMBDA will move toward `adaptive retrieval` mechanisms. Recent advancements in Retrieval-Augmented Generation (RAG) suggest that models can be trained or prompted to dynamically assess the necessity and quality of retrieval (Jiang et al., 2023). Instead of a static hard cutoff, we plan to implement a confidence-based retrieval system, where the agent evaluates the uncertainty of its internal knowledge before querying the knowledge base, effectively creating a dynamic threshold tailored to the complexity of the user's instruction. Furthermore, we aim to explore calibration techniques to normalize similarity scores across different embedding models, ensuring that θ represents a consistent semantic distance regardless of the underlying encoder.

In addition, user-supplied code descriptions vary significantly in clarity and quality. To involve quality control mechanisms in the knowledge base, we will introduce an automated

`Quality Controls Agent`. This agent would utilize an LLM to standardize and rewrite user-contributed knowledge entries before they are indexed.

Finally, regarding the difference between the Full and Core integration modes, we agree that the relationship between task complexity and integration level deserves closer examination. The Full mode provides more complete support but also leads to higher latency and greater token usage. We are considering a new architecture that includes a Routing Agent for the future version of LAMBDA. This agent would analyze the user's intent and decide how much knowledge integration is needed. It could choose the Core mode for routine dataframe operations and switch to the Full mode for more complex, domain-specific modeling tasks. Such dynamic selection would help balance computational efficiency and analytical capability.

1.4 Data-Awareness Retrieval

In agent systems, generation-focused RAG typically relies solely on semantic similarity between user queries and code descriptions. In the data science contexts, this creates a blind spot: the system overlooks the dataset's intrinsic statistical properties. As a result, purely text-based retrieval may surface solutions that are semantically relevant (e.g., "perform regression") but statistically inappropriate for the specific data (e.g., ignoring high dimensionality, sparsity, or heteroscedasticity).

Motivated by the insights from Professor Fan Zhou and Professor Bang Liu, we propose a *Data-Aware Retrieval* framework that synthesizes user intent with statistical characteristics of the dataset. Rather than treating the dataset as a passive input, we make it an active component in the retrieval process. Specifically, retrieval is conditioned on a *Data Profile*—a vector that encodes key meta-features of the dataset, such as dimensionality, skewness, sparsity, and correlation structure.

In this framework, retrieval operates jointly on both the data context and the semantic context. The final retrieval score is computed as a weighted sum: $\text{Score} = \alpha \cdot S_{\text{data}} + \beta \cdot S_{\text{semantic}}$, where S_{data} measures the similarity between the target dataset's profile and the metadata associated with candidate code snippets, and S_{semantic} measures the alignment with user instructions. By fine-tuning the weights (α, β) and the underlying embeddings, the system retrieves knowledge that is not only semantically appropriate but also data-consistent, potentially addressing key limitations of traditional RAG in empirical data analysis.

2 Benchmark for Data Agents

Data science problems are inherently open-ended, making it difficult to define a single ground-truth solution and, in turn, to build reliable evaluation benchmarks. Professor David Donoho emphasized that academia must move beyond tool-building to establish standards for scientific rigor, reproducibility, and evaluation. Dr. Sammi Tang and Dr. Xuewei Wang similarly underscored the need for benchmarks to guide more rigorous and principled progress. Recent efforts have introduced benchmarks for data agents, but limitations remain: some include only a small set of tasks (Huang et al., 2024), while others span a narrow set of domains (Hu et al., 2024), constraining applicability and generality.

To this end, we have recently tried to build a reliable, comprehensive benchmark for data agents. Our preliminary efforts involve constructing the benchmark data derived from a corpus of statistical learning textbooks and supplemented by highly-voted, complex datasets from platforms such as Kaggle. Based on these resources, we are constructing the most comprehensive benchmark for data agents, designed to systematically test agents across diverse data modalities, domains, analytical pipelines, and programming

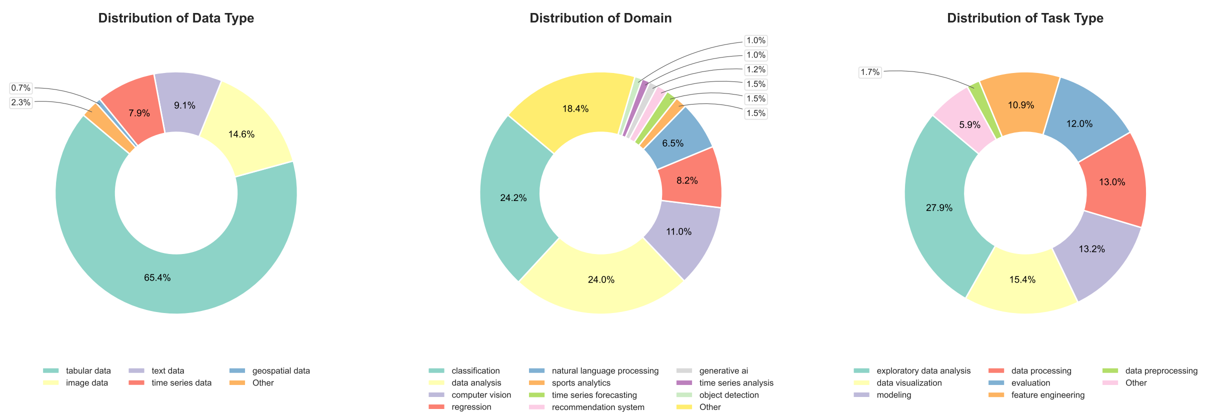

Figure 2: Overview of the benchmark instances.

languages. Each benchmark item includes structured metadata, such as data type, domain, task formulation, reasoning steps, and expected answer or metrics to support rigorous and reproducible assessment. Currently, we have collected more than 2,000 datasets encompassing over 10,000 tasks across diverse domains. Figure 2 shows an overview of the benchmark instances we have curated to date, including their distribution across data modalities, and problem domains. The finalized benchmark can be found in DSAEval Sun et al. (2026).

Evaluating data agents requires broad coverage across domains, data types, and analytical methodologies. Because different disciplines rely on specialized analytic strategies, expanding benchmarks with domain-specific datasets is essential for assessing the generality and robustness of systems like LAMBDA. In addition, as Professor David Donoho noted, competitive platforms such as Chatbot Arena can capture real users' preferences and feedback. We believe such benchmarks are important for advancing rigor and reproducibility in the field.

3 Reasoning and Planning

Reasoning can enhance LLM's ability in downstream tasks by test-time scaling (Muennighoff et al., 2025). In the current version of LAMBDA, we intentionally prioritize human-in-the-loop, so the system's response generation does not explicitly incorporate these techniques yet. As emphasized by Professor Xihong Lin, Professor Fan Zhou and Professor Bang Liu in the discussion, reasoning ability could substantially strengthen LAMBDA's capabilities. We are currently exploring approaches to incorporate reasoning modules into a fully end-to-end operational mode.

3.1 Incorporating a Reasoning Module Directly

An intuitive approach is to elicit explicit reasoning through prompt engineering or to introduce a dedicated reasoning agent. Recent advanced techniques such as Chain-of-Thought (CoT) (Wei et al., 2022), Tree of Thoughts (ToT) (Yao et al., 2023), and Graph of Thoughts (GoT) (Besta et al., 2024) can support complex tasks by enabling structured thinking, reflection, and exploration of multiple solution paths—particularly well-suited to open-ended data science workflows (Sun et al., 2025b). As Professor Xihong Lin noted, an analysis-planning agent could further enhance the system by refining research goals, mapping them to appropriate analytic procedures, and proposing a coherent, standards-aligned analysis pipeline.

Following this suggestion, we plan to extend the current two-agent system to a multi-agent framework that includes a planning agent and an analyst agent. The planning agent will parse the user's question, assess available data and environmental resources, perform high-level reasoning, and allocate subtasks to specialized sub-agents. These sub-agents, such as the analyst, will use the environment's tools to execute their subtasks step by step and return feedback to the planning agent, which will iteratively update the plan until the overall task is completed.

3.2 Reinforcement Learning for Statistical Reasoning

Reinforcement Learning (RL) has been widely used in reasoning LLM (Xu et al., 2025). Professor Fan Zhou and Professor Bang Liu propose framing data analysis as a reinforcement learning environment to enhance reasoning capabilities. LAMBDA's modular architecture provides a natural foundation for this, where agent actions and environment states can be clearly defined.

Recent advances, such as DeepAnalyze (Zhang et al., 2025), have demonstrated the efficacy of Reinforcement Learning in training agentic LLMs for autonomous data science. Motivated by this, we will adopt a statistics-oriented RL strategy to inject statistical reasoning into LAMBDA. The proposed framework comprises the following components:

- **Process-Oriented Action Space:** Standard LLMs often rush into code generation without adequate planning. To encourage fully automatic statistical analysis, the agent's action space must explicitly utilize control tokens to delineate reasoning steps. Inspired by DeepAnalyze, we introduce specific tokens such as `<Analyze>` for planning and self-verification, `<Code>` for generating executable codes, and `<Execute>` for executing the code in the sandbox. This structure enforces the LLM, ensuring that code generation is preceded by statistical deliberation.

- **Hybrid Reward Modeling:** In statistical analysis, code that executes without error is not necessarily statistically sound. Therefore, reliance on binary execution feedback is insufficient. We advocate for a hybrid reward model that combines rule-based feedback (checking execution success and formatting) with an "LLM-as-a-judge" mechanism. This judge evaluates the output based on qualitative metrics such as `soundness`, `interpretability`, and `richness`. Optimizing against these metrics incentivizes the agent to prioritize deep analytical insights over superficial execution.

The benchmark dataset we are constructing and other high-quality data synthesized via distillation from advanced proprietary models or adapted from open-source repositories (e.g., DataScience-Instruct-500K) are critical for the success. By implementing this RL framework, we aim to transform LAMBDA from a static code generator into an autonomous researcher capable of adaptive and rigorous statistical reasoning.

4 Expanding Roles for Collaboration

While LAMBDA's current dual-agent architecture (Programmer and Inspector) serves as a robust baseline, several discussants rightly point out that complex, real-world data science projects require a richer assembly of specialized roles. Professor Xiao-Li Meng, Professor Fan Zhou, Professor Bang Liu, Dr. Sammi Tang and Dr. Xuewei Wang advocate for expanding specialized roles such as a Data Engineer for preprocessing, Model Builder for modeling, Visualizer/Reporter/Result Agent to produce figures and interpret results, etc. However, we approach this expansion with caution.

Although agent frameworks like AutoGen (Wu et al., 2024) and MetaGPT (Hong et al., 2023) demonstrate the power of specialized multi-agent systems, recent work, LIMI (Less Is More for Agency) show that agentic capability does not improve simply by increasing the amount of supervision data. We believe this hypothesis similarly applies to the number of agents; mastering agency requires understanding its essence, not scaling the role of agents Xiao et al. (2025). We contend that data science differs fundamentally from general software engineering or scientific discovery, where modularity often allows agents to operate independently. Data analysis is inherently state-dependent and sequential: each step relies on the exact variable definitions, transformations, and memory state established previously. For example, a visualization at time $t + 1$ depends entirely on the preprocessing choices made at time t.

As noted in recent studies on multi-agent collaboration, domains requiring high context sharing and tight dependencies are often unsuited for large, fragmented agent teams. Introducing too many specialized agents (e.g., separating "Data Cleaning" from "Feature Engineering" into different personas) exacerbates the risk of "Context Confusion". In such scenarios, the overhead of serializing state and communicating context between agents can outweigh the benefits of specialization, leading to hallucinations where a downstream agent references a variable that an upstream agent modified or deleted.

Therefore, rather than an expansive village of agents, we propose a streamlined Atomic Team with skilled atomic agents that may be sufficient to handle the majority of data science workflows while maintaining tight context coherence.

- `The Planner`: Responsible for the high-level roadmap and reasoning (as discussed in Section 3.1). This agent holds the global view of the user's scientific intent.
- `The Data Scientist`: A unified execution role responsible for end-to-end coding. By keeping coding within a single "expert" persona, we ensure that variable state and logic remain consistent across steps, avoiding the fragmentation issues of passing code between a "Data Processor" and a "Modeler."
- `The Inspector`: The critical quality control layer that reviews code and outputs for both syntactic correctness and semantic validity, as emphasized by Dr. Sammi Tang, Dr. Xuewei Wang and Professor Xiao-Li Meng.
- `The Reporter`: Dedicated to the final synthesis, organizing files, and translating technical results into the narrative format required for human consumption.

This multi-agent framework is streamlined, avoids unnecessary agent roles, and facilitates clear and well-structured management of context.

5 Deepening Human-AI Collaboration

LAMBDA employs a human-in-the-loop mechanism to involve users in the decision-making process, allowing them to guide or adjust the agent's actions as needed. Dr. Sammi Tang and Dr. Xuewei Wang regard this as a valuable feature for enhancing transparency and control. In addition, Professor Fan Zhou and Professor Bang Liu, Mr. Mert Yuksekgonul, and Professor James Zou suggest that the current form can be more interactive. They advocate moving from a supervisor-worker dynamic to a genuine collaborative partnership, in which the agent actively supports and augments human intelligence rather than simply executing tasks.

5.1 Active Interaction and Intent Awareness

Current LLMs are optimized to be passive responders rather than active collaborators. When faced with ambiguity, they often make silent assumptions instead of seeking clarification. CollabLLM (Wu et al., 2025) addresses this via reinforcement learning, shifting the agent from silent guessing to active inquiry. For example, rather than unilaterally imputing missing data, the agent would ask, "Do you prefer simple mean imputation or a more robust KNN approach?" This promotes the nuanced, conversational interactions essential for rigorous analysis.

5.2 AI as "Mindware"

Professor Xiao-Li Meng introduces the concept of "Mindware Agents," tools designed not merely to generate outputs but to enhance users' data intelligence. This aligns with Mr. Mert Yuksekgonul and Professor James Zou's view of agents as educational tools that simulate research workflows. We embrace this perspective and aim to design LAMBDA to nudge users toward better statistical practice.

We plan to implement Professor Xiao-Li Meng's proposed "Data Minder" as a dedicated Quality Control Agent that uses "Reverse Prompting." Rather than waiting for user input, it proactively presents a checklist of data-quality questions, for example, "What is the provenance of this dataset?" and "Are there potential selection biases in the collection process?" By embedding these nudges into the workflow, data quality assessment becomes a routine, integral part of the analysis.

6 Engineering Considerations: Building Robust Infrastructure

While the conceptual design of agents is critical, Professor David Donoho points out issues in the system, such as off-target reports. In addition, Dr. Sammi Tang and Dr. Xuewei Wang noted that the transition from an academic prototype to a production-ready system requires rigorous engineering optimization.

6.1 Problem of Irrelevant Report

The observation that LAMBDA produced an irrelevant report for the Electricity Cost Prediction dataset[1] is informative. Professor David Donoho suggested that this may reflect an out-of-distribution (OOD) issue, since the dataset was released after the knowledge cutoff dates of the GPT-4/GPT-5 series. In our implementation, however, model outputs are driven primarily by the chat history, so we cannot rule out limitations in model capability, including possible OOD effects. In particular, we hypothesize that the irrelevance may also stem from deficiencies in context construction (for example, retrieval and prompt assembly).

To investigate this, we compared two approaches for structuring the reporting module's

[1]https://www.kaggle.com/datasets/shalmamuji/electricity-cost-prediction-dataset

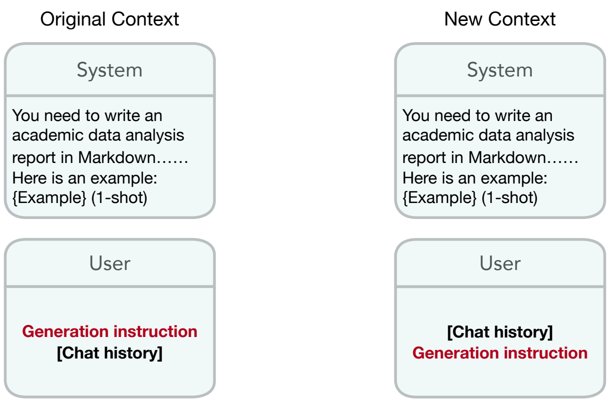

Figure 3: Re-organizing context for report generation.

Table 2: Results of report generation under different context configurations and models.

Model	Original Context	New Context
Gpt4o-mini	Sometimes irrelevant	Correct
Gpt4.1-mini	Correct	Correct
Gpt5-mini	Correct	Correct

context: placing the generation instructions above the chat history (the original setting) and placing the instructions below the chat history (the new setting), as illustrated in Figure 3. We tested multiple models on this dataset under both configurations. The results are summarized in Table 2.

The results indicate that the new context configuration improves the robustness of report generation. Placing the generation instruction at the end of the dialogue context appears to be a more effective strategy. However, factors such as the model's underlying knowledge, its instruction-following ability, and the overall length of the dialogue may also influence performance to a lesser extent. Since LAMBDA is designed to be compatible with most LLMs, using the latest and most capable models remains an effective way to mitigate such issues.

6.2 Scaling Environment Supporting

At present, the core of LAMBDA is implemented using the Jupyter Python kernel. Dr. Sammi Tang and Dr. Xuewei Wang noted that restricting LAMBDA to a Python-only environment limits its adoption within the broader statistical community, where R remains a dominant language. We acknowledge this limitation and share this concern. We are actively working on expanding the supported programming languages and software within the environment. In future versions, we will integrate a formal sandbox, which will enable LAMBDA to execute shell commands as well as Python, R, Julia, SQL, and perform file-retrieval actions.

Moreover, the hybrid programming workflow suggested by Dr. Sammi Tang and Dr. Xuewei Wang can also be naturally supported by this sandbox framework. For example, an agent could perform data preprocessing in Python and conduct statistical modeling in R (using specialized packages such as lme4) within the same session. Such capabilities would significantly enhance the flexibility and applicability of LAMBDA.

6.3 Academic Research and Industry Competition

We appreciate Professor David Donoho's perceptive analysis of the current commercial landscape. He rightly notes that the market for data-agent products is becoming highly competitive and is attracting venture capital. Indeed, venture-backed rhetoric often promises "frictionless" autonomy beyond present capabilities, intensifying competition. In this context, Professor Donoho's assistant, Dr. Elena Belogolovsky, compared LAMBDA with several commercial products on the analysis of Electricity Cost Prediction dataset and found that the Colab Data Science Agent performed best on this particular dataset and task, with LAMBDA ranking second.

Regarding the performance gap between LAMBDA and Google Colab on the Electricity Cost Prediction dataset, the underlying models are a key factor. Our evaluation used GPT-4o-mini as LAMBDA's base model, while the Colab Data Science Agent relied on the substantially more powerful Gemini 2.5 Pro. For example, GPT-4o-mini scores 40.2

on GPQA Benchmark Rein et al. (2024)[2], whereas Gemini 2.5 Pro achieves 84 on the diamond set[3]. This foundational capability gap likely explains much of the difference in analytical performance. Moreover, we have verified that LAMBDA completes the task successfully when equipped with comparably powerful models.

We acknowledge that academic resources are far more limited than those of venture-capital–backed startups building data-analysis agents, and that academia faces structural hurdles in matching industry-grade engineering and infrastructure. Even so, academic research has distinct strengths: it can originate novel ideas, serve the public interest, and set rigorous, transparent evaluation standards that shape the field. Moving forward, we will release open benchmarks and process-based evaluation suites for data agents; conduct user studies on collaboration and learning outcomes in data science education; and propose standards for data-quality audits and reproducible reporting. Our aim is to complement industry's scale with academic rigor, transparency, and public goods that benefit the entire ecosystem.

7 Future Directions and Open Challenges

While the current iteration of LAMBDA provides a strong foundation, the discussants have identified several important issues that remain to be addressed. In addition to the directions outlined above in response to their comments, we will address the following issues in future work.

Professor Fan Zhou and Professor Bang Liu point out that LAMBDA currently treats each task in isolation, failing to carry over knowledge from previous successes or failures. To address this, designing Self-Evolving Agents is a way. Future work could incorporate

[2]https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/

[3]https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#enhanced-reasoning

a long-term memory module. This would allow the agent to curate a personal library of “successful code skills” and “debugged error patterns,” enabling compound growth in capability and preventing the repetition of past mistakes across different sessions.

We acknowledge that agents with “Computer Use” capabilities pose significant risks, such as accidentally deleting critical user files. However, data agents like LAMBDA operate under a different risk profile. They generally do not require, nor should they be granted, direct control over the host operating system. A virtualized sandbox environment restricted to code execution is sufficient to meet analytical needs while isolating the system from destructive actions.

Furthermore, privacy remains a critical bottleneck for commercial or API-based agents. These systems often require data uploads or allow the LLM to inspect raw data contents (e.g., executing `data.head()`) (Sun et al., 2025b), creating inherent leakage risks. A trade-off solution is to deploy open-source LLMs locally, but this requires powerful hardware and incurs higher electricity costs. Thus, developing API-based solutions with privacy preservation remains an important research direction.

8 Conclusion

In summary, the discussants highlighted LAMBDA’s promise for automating data science and statistical analysis and offered many insightful suggestions. We have begun integrating their feedback to strengthen domain-knowledge integration, sharpen statistical and analytical reasoning, develop more reliable benchmarks, deepen human–AI collaboration, and enhance the robustness of our engineering infrastructure. We have conducted preliminary studies, addressed several issues in the current system, and will continue to incorporate these recommendations. We are optimistic that data analysis agents such as LAMBDA will help democratize statistics and data science, enabling broader participation in data-driven

inquiry. We sincerely thank the editor, Professor Hongtu Zhu, for convening the discussion and all the distinguished discussants for their constructive feedback.

Data Availability Statement

the author(s) of the manuscript have legitimate access to and permission to use the data used in this rejoinder, which are publicly available at `https://www.kaggle.com/datasets/shalmamuji/electricity-cost-prediction-dataset`, `https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence`, and `https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#enhanced-reasoning`. The source code of LAMBDA is available at the GitHub deposit `https://github.com/AMA-CMFAI/LAMBDA`.

Disclosure Statement

The authors report there are no competing interests to declare.

References

Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., et al. (2024). Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Wang, J., Zhang, C., Wang, Z., Yau, S. K. S., Lin, Z., et al. (2023). Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Hu, X., Zhao, Z., Wei, S., Chai, Z., Ma, Q., Wang, G., Wang, X., Su, J., Xu, J., Zhu, M., Cheng, Y., Yuan, J., Li, J., Kuang, K., Yang, Y., Yang, H., and Wu, F. (2024). Infiagent-dabench: Evaluating agents on data analysis tasks. *arXiv preprint arXiv:2401.05507*.

Huang, Q., Vora, J., Liang, P., and Leskovec, J. (2024). Mlagentbench: Evaluating language agents on machine learning experimentation. *arXiv preprint arXiv:2310.03302*.

Jiang, Z., Xu, F. F., Gao, L., Sun, Z., Liu, Q., Dwivedi-Yu, J., Yang, Y., Callan, J., and Neubig, G. (2023). Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. B. (2025). S1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20286–20332.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. (2024). Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Steck, H., Ekanadham, C., and Kallus, N. (2024). Is cosine-similarity of embeddings really about similarity? In *Companion Proceedings of the ACM Web Conference 2024*, pages 887–890. ACM.

Sun, M., Han, R., Jiang, B., Qi, H., Sun, D., Yuan, Y., and Huang, J. (2025a). Lambda: A large model based data agent. *Journal of the American Statistical Association*, pages 1–13.

Sun, M., Han, R., Jiang, B., Qi, H., Sun, D., Yuan, Y., and Huang, J. (2025b). A survey on large language model-based agents for statistics and data science. *The American Statistician*, pages 1–14.

Sun, M., Xie, Y., Wu, Y., Han, R., Jiang, B., Sun, D., Yuan, Y., and Huang, J. (2026). Dsaeval: Evaluating data science agents on a wide range of real-world data science problems. *arXiv preprint arXiv:2601.13591*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., et al. (2024). Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*.

Wu, S., Galley, M., Peng, B., Cheng, H., Li, G., Dou, Y., Cai, W., Zou, J., Leskovec, J., and Gao, J. (2025). Collabllm: From passive responders to active collaborators. *arXiv preprint arXiv:2502.00640*.

Xiao, Y., Jiang, M., Sun, J., Li, K., Lin, J., Zhuang, Y., Zeng, J., Xia, S., Hua, Q., Li, X., et al. (2025). Limi: Less is more for agency. *arXiv preprint arXiv:2509.17567*.

Xu, F., Hao, Q., Zong, Z., Wang, J., Zhang, Y., Wang, J., Lan, X., Gong, J., Ouyang, T., Meng, F., et al. (2025). Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36:11809–11822.

Zhang, S., Fan, J., Fan, M., Li, G., and Du, X. (2025). Deepanalyze: Agentic large language models for autonomous data science. *arXiv preprint arXiv:2510.16872*.