Federated RLHF Pipeline for Personalized Tutoring

Jérémy Chaverot
jeremy.chaverot@epfl.ch
Advised by Vinitra Swamy and Paola Mejia,
at the ML4ED lab led by Prof. Tanja Käser
EPFL, Switzerland

Abstract

TODO

ACM Reference Format:
Jérémy Chaverot. 2025. Federated RLHF Pipeline for Personalized Tutoring. In *Proceedings of ScholéAI.* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Developped in collaboration with EPFL's ML4ED research lab and recognized by the 2024 Learning Engineering Tools Competition, MIT Solve, and the EPFL Ignition Grant, Scholé AI is a startup committed to transforming the future of online education.

Traditional online platforms like Coursera and edX often suffer from high dropout rates in MOOCs, with figures reaching up to 90% [9]. Scholé adresses this challenge by offering adult learners personalized, flexible learning experiences. Through multimodal contents, it builds adaptative learning journeys tailored to each individual.

At the core of Scholé is Olé, an AI teaching assistant that designs personalized data science learning paths based on your team's specific context. By leveraging cutting-edge research in AI for education, Scholé redefines adult upskilling, and ensures learners are equipped for the challenges of tomorrow. However, transforming Olé into an effective AI tutor requires rigorous training and refinement.

Personalization, scalability, and alignment are critical challenges in educational AI. While large language models have demonstrated strong generative capabilities, aligning their outputs with learner preferences remains a difficult task, particularly when data is distributed and subject to strict privacy constraints.

To address this, we adopt FedBiscuit [16], a recent state-of-the-art framework for federated reinforcement learning with human feedback (RLHF), which we adapt for the context of personalized learning. By leveraging a federated setup, this approach enables preference alignment without centralizing sensitive learner data, ensuring both scalability and privacy.

As Scholé is a young startup with limited access to real user interaction data, we rely on large language models to generate high-quality synthetic data representing user preferences. We customize and deploy FedBiscuit on our cluster, applying it to this synthetic data for our personalized learning use case. Our focus is on generating preference pairs and tailored learning curriculums for diverse learner profiles. This work highlights the potential of federated RLHF for scalable, privacy-preserving personalization in low-data settings.

2 Related Work

LLMs in Education. The growing accessibility of large language models (LLMs) via platforms such as Hugging Face has opened up new possibilities in education for both learners and instructors [14]. Students increasingly rely on AI to improve learning efficiency, solve problems, and automate routine tasks, while educators explore ways to enhance teaching strategies and content delivery.

Adaptive learning environments aim to tailor educational content and learning strategies to individual learners' needs, preferences, and performance. To dynamically adjust the learning experience, prior research highlights the importance of user modeling, which means tracking learner behavior, progress, and characteristics. Systems typically adapt content difficulty, feedback, and instructional sequencing to improve engagement and learning outcomes. The integration of intelligent agents has proven effective in managing these personalized adjustments [5]. This foundational work also supports our project's focus on generating synthetic student profiles and aligning content with diverse learner modalities.

Decentralized Training. Federated learning (FL) [8] offers a privacy-preserving and communication-efficient solution for training LLMs on decentralized data. Instead of centralizing sensistive datasets, FL enables multiple clients to collaboratively fine-tune a shared model by exchanging only model updates.

Federated Averaging (FedAvg) [8], also refered as Local SGD [13], is the standard algorithm in FL. It consists of alternating between a few local stochastic gradient updates at client nodes, followed by a model averaging update at the server.

Recent frameworks like OpenFedLLM [17] have demonstrated the feasability of applying FL to instruction tuning and alignment, two essential components for adapting LLM behavior to user preferences. Using techniques like parameter-efficient fine-tuning (e.g., LoRA [6]), these models can be trained effectively across distributed nodes, and even outperforming centralized baselines in domain-specific tasks. In this project, we adopt this paradigm to align LLMs with learner preferences.

Preference Alignment. Aligning LLMs with human preferences is key to generating pedagogically effective outputs, especially in educational settings where hallucinations or misinterpretations can harm learning [15]. After supervised fine-tuning (SFT),

ScholéAI, team@schole.ai

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

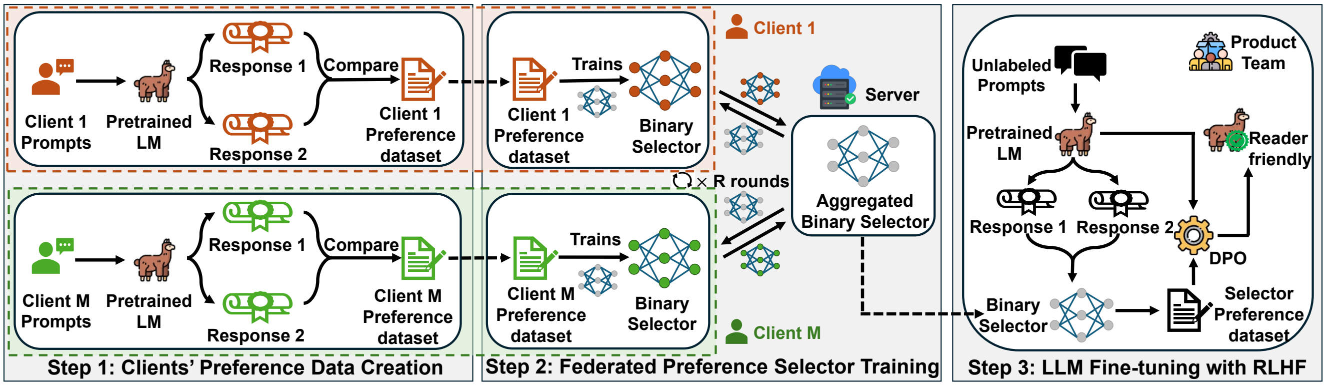

Figure 1: Overview of FedBiscuit.

this alignment is typically achieved through RLHF, a family of methods that optimize models based on preference data.

Two main approaches exist within RLHF. The classic RLHF pipeline involves training a reward model to mimic human preferences, followed by reinforcement learning using algorithms like Proximal Policy Optimization (PPO) [12]. While effective, this setup is complex and prone to instability.

In contrast, Direct Preference Optimization (DPO) [10] simplifies the process by eliminating the reward model. It directly fine-tunes the base model on preference pairs, optimizing a loss that favors preferred responses. This makes DPO more stable and computationally efficient, while still achieving strong alignment performance [3].

For our project, DPO is a better fit: it aligns models directly to learner feedback without requiring a fragile intermediary reward model, and is more suitable for federated training in low-data settings. Moreover, it only requires constructing a dataset of preference pairs (each consisting of a chosen and a rejected learning plan) rather than assigning numerical scores to individual plans, which is more subjective and less straightforward.

Synthetic preference and behavioral data TODO

3 FedBiscuit

TODO: Add that FedBiscuit is natively compatible with multi-GPU training and PEFT techniques, great for scalability

FedBiscuit is a state-of-the-art federated reinforcement learning framework designed to align language model behavior with user preferences while preserving user privacy and minimizing communication overhead.

3.1 Pipeline Overview

As observed in Figure 1, the pipeline consists of two main training stages. First, clients collaboratively train a binary selector model via federated learning to predict preferences between output pairs (e.g., learning plans). After convergence, the global selector is used to supervise the fine-tuning of the LLM through DPO. For each input, the LLM (e.g., Olé) generates two candidate outputs; the selector picks the preferred one, and DPO updates the LLM accordingly. This avoids the need for a fragile reward model and enables scalable alignment in privacy-sensitive contexts.

3.1.1 Federated System Design. As mentioned earlier, FedBiscuit follows a client-server architecture where, in each communication round, the server distributes the current global model to a subset of selected clients. Each client then fine-tunes the model locally using its own preference data. Once local training is complete, the clients send back their model updates, which the server aggregates to form an updated global model. This iterative process continues until convergence or until a predefined number of rounds has been reached.

3.1.2 Client-Side Operations. On the client side, each participant preprocesses their data to construct training samples in the form of preference pairs (i.e., chosen vs. rejected responses to a given prompt). In our adaptation for educational personalization, these preference pairs correspond to selected and discarded learning plans given the student behavioral data and preferences. Local models are trained as a classification task using the cross-entropy (CE) loss l_{CE}.

$$l_{CE} = \dots \tag{1}$$

3.1.3 Server-Side Aggregation. The central server aggregates model updates without ever accessing raw client data, only model weights or gradients are exchanged. This is enabled by a secure aggregation mechanism, which guarantees that the server can only see the final aggregated result [2]. The server also handles client sampling and adapts dynamically to slower or disconnected clients during training.

$$aggregation\ \ formula \tag{2}$$

3.1.4 LLM-as-a-Judge Evaluation. To evaluate alignment, FedBiscuit leverages Auto-J, a 13B-parameter generative language model specifically trained to assess other models through natural language critiques. Developed using large-scale user queries and LLM responses, Auto-J achieves competitive performance across 58 real-world evaluation scenarios. In the context of Scholé, Auto-J is particularly valuable: it provides an automated, scalable, and interpretable means of assessing how well Olé aligns with learner preferences, without relying on expensive human annotations or black-box evaluators.

3.2 Use Case and Limitations

3.2.1 TL;DR Summarization Task. To benchmark alignment performance, FedBiscuit uses the TL;DR summarization task, which consists in generating concise summaries of Reddit posts. This controlled setup helps evaluate preference learning and is later used to estimate the amount of data each client needs in educational settings like for Scholé.

3.2.2 Limitations. Despite its strengths, FedBiscuit happens to face some limitations. First, communication overhead remains a bottleneck, especially as the number of clients increases. Second, the system's effectiveness can degrade in the presence of extreme client heterogeneity, both in data quality and hardware capabilities. Finally, the current framework assumes clients have enough data to perform meaningful updates, which may not hold in sparse real-world scenarios.

In our specific setup, some of these challenges are mitigated. We assume client training occurs on the same server-side machine, removing issues related to hardware variation and network latency. While this setup compromises some of the privacy advantages offered by FedBiscuit, it enables us to establish a strong baseline with full control over the hardware environment. As for data sparsity, this work also introduces methods to generate synthetic data and augment existing datasets, providing a practical solution for low-data scenarios like those found for Scholé use case.

4 Synthetic Data Generation and Evaluation

As Scholé AI is still in its early development phase, we currently do not have access to real-world user data. To lay the foundation for experimentation and model training, we first established a formal typology of the types of data that will characterize student profiles. We categorized the data into two primary types: explicit data and implicit data.

Explicit Data. This category encompasses all directly stated learner preferences and declared learning goals. It includes information such as preferred learning modalities, self-reported motivation, goal orientation, or feedback explicitly given by the learner regarding content relevance or difficulty. These data points are directly interpretable. The detailed categories can be found in Appendix A.

Implicit Data. Also referred to as *behavioral data*, this category includes information derived from the learner's interaction patterns with the learning system. Behavioral data captures observable signals such as time spent on tasks, content navigation paths, click-through rates, quiz completion rates, and error frequencies. The detailed categories can be found in Appendix B.

4.1 Prompt Engineering

4.1.1 System prompt. The system prompt is responsible for guiding the model by specifying the overall structure, formatting, and style of the generated output. Unlike the user prompt, it remains constant across generations.

We implemented a synthetic data generation pipeline built upon prompt engineering techniques, following the methodology presented in [11]. We adopted the **in-context learning** (ICL) [4] with **one-shot prompting**, enabling the model to produce with simplicity structured and context-aware outputs.

In-context learning is a paradigm that allows language models to learn tasks given only a few examples in the form of demonstration. This allows the model to adapt its behavior based on the structure and semantics of the provided context. In *One-shot prompting*, the model is given a single example before generating a new output.

Each system prompt is structured as follows:

- A subgraph extracted from Scholé AI's main *knowledge graph* (KG) **TODO: add how it was extracted**, describing learning modules. This ensures that the model grounds its generation in the actual curriculum.
- A strict requirement to output data in a structured `JSON` format, which enables automatic validation and downstream processing.
- A combination of learning modality and student profile to guide personalization and diversity in the synthetic data.

4.1.2 User prompt. To introduce diversity in the generated data, we incorporate personalization into the user prompt with the learning modality and student profile. This prompt also specifies the number of generations to produce.

Learning Modalities. We defined the following learning modalities to reflect common cognitive styles according to the VARK model:

- *Visual*: prefers diagrams, illustrations, and video content.
- *Auditory*: favors spoken explanations, podcasts, and discussions.
- *Reading/Writing*: learns through text-based material such as articles and notes.
- *Kinesthetic*: benefits from interactive, hands-on experiences.

It's worth noting that these modalities can be somewhat controversial, which is why they should be used with caution [7]. Nonetheless, we adopt them in the early stages due to their simplicity.

Student Profiles. We define several learner types to reflect diverse engagement styles:

- *Goal-Oriented*: driven by milestones and performance tracking.
- *Curious*: explores optional content and asks open-ended questions.
- *Passive*: minimal engagement, avoids extra effort.
- *Fast-Paced*: skims content, prioritizes speed over depth.
- *Confused*: struggles with core concepts, needs clear guidance.
- *Struggling*: performs poorly, benefits from scaffolding and feedback.
- *Social*: learns through discussion and peer interaction.

In particular, for tasks such as data augmentation and generation of learning curriculum preferences, the model is explicitly instructed to interpret the given learner context and adjust its outputs accordingly. For example, a kinesthetic-exploratory learner may be recommended simulations and interactive labs, while a visual goal-oriented learner may receive structured video tutorials and milestone-based feedback.

TODO: Add diagram to summarize prompt structure

The latest versions of the prompts are available at this link.

4.2 Streamlit Interface for Prompt Interaction

To support fast iterations on prompt design and inference control, we developed an interactive interface using **Streamlit**.

Streamlit is an open-source Python framework that enables rapid development of web applications for data science and machine learning tasks. It allows users to manipulate model inputs and visualize outputs without requiring front-end development.

Our Streamlit interface provides the following features:

- Choosing the task type.
- Editing prompt content in real time.
- Selecting learning modalities and student profiles.
- Evaluating the generated synthetic data.

The website can be accessed at this link: https://scholeai-data-generation.streamlit.app/

TODO: Add screenshots

4.3 Evaluation Pipeline: Automated and Human-in-the-Loop

Once the synthetic data pipeline was functional, the next challenge was to assess the quality of the generated outputs. We implemented a two-tiered evaluation process combining automated validation and human annotation.

4.3.1 Automated Evaluation. Given the structured nature of the outputs, we first applied a series of automated validation tests to each generated `JSON` sample. These included the following checks, among others:

- All required fields were present and of the correct type.
- Categorical values (e.g., learning modality) belonged to the expected set.
- Text fields were non-empty where applicable.
- Timestamps respected ISO 8601 formatting and logical sequencing.
- Numeric ratings fell within predefined bounds (e.g., 1–5).

These tests served as an efficient first-pass filter, rejecting malformed samples before any human review.

4.3.2 Human Evaluation via Yes-No Questions. For semantic and pedagogical validity, we developed a human annotation interface embedded on the Streamlit web app. Annotators assessed each sample using a series of *yes-no questions*, such as:

- Is the proposed curriculum realistic for the given student profile?
- Does the plan match the specified learning modality?
- Is the behavioral data coherent and believable?

This human-in-the-loop component allowed for rapid, structured assessment without requiring full expert review. The feedback was used to iteratively improve prompt design and increase generation quality. The full list of questions is provided in Appendix C.

4.3.3 Visualization of Evaluation Results. To monitor performance and guide refinements, we used librairy Plotly to produce interactive dashboards summarizing evaluation statistics for automated checks and human questions. These summaries help understand what is working or not in the prompts and make it easier to track changes over time.

TODO: Add screenshot and some stats

4.4 Challenges in Synthetic Data Generation

The combined evaluation process surfaced several key insights. Using GPT-4o-mini as the generating model, automated validation revealed high structural correctness in most cases. However, as the number of generated samples increases, structural errors (such as malformed `JSON`) become more frequent. This is likely due to the verbosity of the syntax, where even a small mistake (e.g., a missing comma) can break the entire structure. Switching to a more lightweight format like `YAML` could help mitigate this. Alternatively, using a framework like LMQL [1], which enables constrained generation through embedded code logic, could enforce correct output structure regardless of sample count.

Beyond structural issues, semantic inconsistencies were more prevalent. These included unrealistic behavioral traces, such as improbable activity sequences or timestamp patterns that did not reflect plausible learner behavior.

These challenges highlight the limitations of LLM-based generation, particularly when prompt conditioning is weak or underspecified. They underscore the need for clear in-context examples for format control, and a multi-layer evaluation process to ensure data quality. Notably, LMQL could be a promising solution for addressing both structural and semantic reliability during generation.

TODO: Add examples of malformed samples

5 Experiment

- TL;DR summarization task as baseline for system validation.
- Objective: Identify optimal training dataset size and configuration.
- Simulation setup: client number, synthetic dataset distribution.
- Tested model architectures and size variations.
- Explored hyperparameters: federated rounds, learning rates, batch sizes.
- Hardware specifications
- Use of Weight&Biases to monitor trainings.
- Challenges

6 Results

- Quantitative metrics
- Comparative analysis of simulation configurations and dataset sizes
- Visualizations: graphs or barplots
- In-depth analysis: student preference alignment, edge cases, anomalies
- Interpretation of results in federated learning context and practical implications.

7 Conclusion and Future Work

- Summary of achieved goals and core contributions in federated RLHF for education
- What can be improved
- Potential next steps: deployment with real student data, model scalability improvements
- Directions for future research: (ideas: real-time feedback integration, adaptive learning paths, constrained inference for synthetic data)

References

[1] Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. 2023. Prompting Is Programming: A Query Language for Large Language Models. *Proceedings of the ACM on Programming Languages* 7, PLDI (June 2023), 1946âĂŞ1969. doi:10.1145/3591300

[2] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2016. Practical Secure Aggregation for Federated Learning on User-Held Data. arXiv:1611.04482 [cs.CR] https://arxiv.org/abs/1611.04482

[3] Stephen Casper and al. 2023. Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback. arXiv:2307.15217 [cs.AI]

[4] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A Survey on In-context Learning. arXiv:2301.00234 [cs.CL] https://arxiv.org/abs/2301.00234

[5] Hassan A. El-Sabagh. 2021. Adaptive e-learning environment based on learning styles and its impact on development students' engagement. *International Journal of Educational Technology in Higher Education* 18, 1 (2021), 53. doi:10.1186/s41239-021-00289-4

[6] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685 [cs.CL] https://arxiv.org/abs/2106.09685

[7] Ermira Idrizi, Sonja Filiposka, and Vladimir Trajkovik. 2018. VARK Learning Styles and Online Education: Case Study.

[8] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise AgÃijera y Arcas. 2023. Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv:1602.05629 [cs.LG] https://arxiv.org/abs/1602.05629

[9] Daniel F. O. Onah, Jane Sinclair, and Russell Boyatt. 2014. Dropout rates of massive open online courses : behavioural patterns, In 6th International Conference on Education and New Learning Technologies, L. Gómez Chova, A. López Martínez, and I. Candel Torres (Eds.). *EDULEARN14 Proceedings*, 5825–5834. https://wrap.warwick.ac.uk/id/eprint/65543/

[10] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. *ArXiv* abs/2305.18290 (2023).

[11] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2025. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. arXiv:2402.07927 [cs.AI] https://arxiv.org/abs/2402.07927

[12] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG] https://arxiv.org/abs/1707.06347

[13] Sebastian U. Stich. 2019. Local SGD Converges Fast and Communicates Little. arXiv:1805.09767 [math.OC] https://arxiv.org/abs/1805.09767

[14] Shen Wang, Tianlong Xu, Hang Li, Chaoli Zhang, Joleen Liang, Jiliang Tang, Philip S. Yu, and Qingsong Wen. 2024. Large Language Models for Education: A Survey and Outlook. arXiv:2403.18105 [cs.CL]

[15] Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023. Aligning Large Language Models with Human: A Survey. arXiv:2307.12966 [cs.CL]

[16] Feijie Wu, Xiaoze Liu, Haoyu Wang, Xingchen Wang, Lu Su, and Jing Gao. 2025. Towards Federated RLHF with Aggregated Client Preference for LLMs. arXiv:2407.03038 [cs.CL] https://arxiv.org/abs/2407.03038

[17] Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. 2024. OpenFedLLM: Training Large Language Models on Decentralized Private Data via Federated Learning. arXiv:2402.06954 [cs.LG] https://arxiv.org/abs/2402.06954

A Explicit Data Categories

The following list describes the categories of explicit data used to characterize Scholé AI learners. These data points are directly provided by the user or derived from user feedback and preferences.

- **ratings_on_modules**: User ratings on the effectiveness and quality of learning modules.
- **approval_of_content_modifications**: Whether the user accepted or rejected system-suggested changes.
- **explicit_learning_goals**: Stated learning goals or objectives provided by the user.
- **initial_curriculum_state**: Ordered list of module names representing the system-suggested curriculum before any user modifications.
- **drag_and_drop_curriculum_edits**: Reordering of the learning path via drag-and-drop (track index changes).
- **curriculum_editing_feedback**: Feedback provided after modifying the suggested curriculum.
- **preferred_content_format**: User preference among text, video, or audio content.
- **reflection_inputs**: Written explanations for curriculum modifications or preferences.
- **satisfaction_surveys**: Survey responses about overall platform satisfaction.
- **skill_self_assessments**: Self-evaluated skill level before and after learning sessions.
- **relevance_feedback**: Feedback about whether the content matched the user's real-world needs (Likert scale 1-5).
- **difficulty_feedback**: Perceived difficulty level of content (Likert scale 1-5).
- **trust_feedback**: Degree of trust in the platform's tutoring and recommendations (Likert scale 1-5).

B Implicit Data Categories

The following list describes the categories of implicit data collected from user interactions on Scholé AI. These signals are inferred from behavior and engagement patterns.

- **timestamped_clicks**: List of all clicks with timestamps (e.g., button presses, navigation).
- **scrolling_behavior**: Scrolling depth, speed, and frequency during content consumption.
- **time_on_task_per_module**: Duration a user spends on each learning module or page.
- **skipped_modules**: List of modules that were skipped by the user.
- **engagement_metrics**: Completion rates, frequency of activity, interaction levels.
- **pace_tracking_signals**: Estimated learning speed from knowledge tracing (e.g., fast vs slow pace).
- **drop_off_events**: Points where users abandon a module or quit mid-session.
- **content_adaptation_requests**: Requests made by users to adjust difficulty or format.
- **memory_usage_patterns**: Tracking usage of personalized memory features (e.g., saved preferences).
- **interactions_with_tutor**: Timestamps, questions, and responses during tutor interactions.
- **number_of_retries_on_quizzes**: Number of attempts needed to successfully complete quizzes.
- **response_times**: Time taken to answer questions or to interact after prompts.

C Human Evaluation Questions

The following yes/no questions were used by human evaluators to assess the quality and realism of the synthetic student data:

- **QH1.** Do the user behaviors feel realistic and consistent with the assigned student profile?
- **QH2.** Does the sequence of actions and timestamps reflect plausible learning behavior over time?
- **QH3.** Is the generated content (e.g., goals, reflections, preferences) coherent and appropriate given the context?
- **QH4.** Do the interactions with the AI tutor sound natural and relevant to the user's learning progress?
- **QH5.** Does the data reflect meaningful variation across different users or profiles?
- **QH6.** Is there any repetition or redundancy that feels unnatural or overly templated?
- **QH7.** Do the initial curriculum and the user's edits (e.g., module reordering or removals) make sense and align with the learner's profile?

- **QH8.** Would this data be convincing if presented as part of a real learner's interaction log?

Received 06 June 2025.