NeuralNexus at BEA 2025 Shared Task: Retrieval-Augmented Prompting for Mistake Identification in AI Tutors

Numaan Naeem **Sarfraz Ahmad** **Momina Ahsan** **Hasan Iqbal**
MBZUAI
{numaan.naeem, sarfraz.ahmad, momina.ahsan, hasan.iqbal}@mbzuai.ac.ae

Abstract

This paper presents our system for **TRACK 1: MISTAKE IDENTIFICATION** in the BEA 2025 SHARED TASK ON PEDAGOGICAL ABILITY ASSESSMENT OF AI-POWERED TUTORS. The task involves evaluating whether a tutor's response correctly identifies a mistake in a student's mathematical reasoning. We explore four approaches: (1) an ensemble of machine learning models over pooled token embeddings from multiple pretrained language models (LMs); (2) a frozen sentence-transformer using [CLS] embeddings with an MLP classifier; (3) a history-aware model with multi-head attention between token-level history and response embeddings; and (4) a retrieval-augmented few-shot prompting system with a large language model (LLM) i.e. GPT 4O. Our final system retrieves semantically similar examples, constructs structured prompts, and uses schema-guided output parsing to produce interpretable predictions. It outperforms all baselines, demonstrating the effectiveness of combining example-driven prompting with LLM reasoning for pedagogical feedback assessment. Our code is available at https://github.com/NaumanNaeem/BEA_2025.

1 Introduction

Conversational AI systems are increasingly being used for educational applications, particularly in the form of AI-powered tutors that can engage students in instructional dialogues. While recent advances in LLMs have made it possible to generate fluent and context-aware responses, evaluating whether these responses exhibit true pedagogical ability remains a fundamental challenge. Traditional dialogue evaluation metrics, such as fluency, coherence, or BLEU-like scores, fall short in capturing educational effectiveness, such as whether the tutor correctly identifies a student's mistake or provides helpful, targeted feedback.

The BEA 2025 SHARED TASK ON PEDAGOGICAL ABILITY ASSESSMENT OF AI-POWERED TUTORS (Kochmar et al., 2025) addresses this gap by introducing a standardized evaluation benchmark and taxonomy to assess pedagogical abilities in AI-generated tutor responses. In particular, TRACK 1: MISTAKE IDENTIFICATION focuses on determining whether a tutor's response correctly detects and communicates an error in the student's reasoning within a mathematical dialogue. The benchmark used in this task is based on MRBENCH (Maurya et al., 2025), which includes *192* dialogues and over *1,500* responses from human and LLM tutors, annotated across *eight* pedagogical dimensions grounded in learning sciences.

2 Methodology

We tackle TRACK 1: MISTAKE IDENTIFICATION, which involves determining whether a tutor's response correctly identifies a student's mistake in a multi-turn mathematical dialogue. Given the subtle and varied nature of student errors and tutor feedback, this task demands both contextual understanding and pedagogical sensitivity. To address this, we developed and evaluated *four* distinct approaches: three baseline models leveraging traditional classification techniques and transformer embeddings, followed by a final retrieval-augmented few-shot classification technique using LLMs.

2.1 Layered Embedding Extraction with Classical ML Ensemble

In our first baseline, we designed a layered ensemble approach by extracting embeddings from several pre-trained transformer models, including BERT, ROBERTA, XLNET, T5, and GPT-2. To handle this flexibly, we developed a unified LM_EMBED class that tokenizes and encodes both conversation history and tutor responses using each model's specific configuration. We ap-

Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications, pages 1254–1259

plied average pooling over the token embeddings to produce fixed-length vectors for each input, and then averaged the conversation and response vectors to create the final input representation. Using these features, we trained a diverse set of traditional classifiers i.e. SVM, Decision Tree, Random Forest, Logistic Regression, Naive Bayes, KNN, AdaBoost, and MLP, each optimized using GRIDSEARCHCV with 10-fold cross-validation. We then built a meta-classifier by stacking the prediction probabilities from these base models and training a logistic regression model on top. This ensemble strategy allowed us to combine the strengths of different embedding models and classifiers, leading to more stable and accurate predictions compared to using any single model alone.

2.2 Token-Level Attention with History-Aware Model

In our second baseline, we modeled the interaction between the conversation history and tutor response using a token-level attention mechanism without any pooling during embedding extraction. We used a transformer encoder (`sentence-transformers/all-mpnet-base-v2`) to obtain full token-level representations for both the conversation history and the tutor's response. These representations were then passed into a custom multi-head attention module. Specifically, we treated the response as the query (Q) and the history as both the key (K) and value (V) in a standard multi-head attention setup. The output of the attention layer was mean-pooled along the sequence length dimension, and a small feedforward network mapped the pooled vector to three output classes. The model was trained using cross-entropy loss with the ADAMW optimizer, and predictions were generated by taking the argmax over the logits. This architecture allows the model to explicitly attend to relevant parts of the history when interpreting the tutor's response, resulting in a more nuanced classification of pedagogical mistakes.

2.3 Frozen Sentence-Transformer with MLP Classifier

Our third baseline models the pedagogical mistake identification task as a supervised classification problem using fixed sentence embeddings. We use a frozen sentence-level transformer model (`sentence-transformers/all-mpnet-base-v2`) to independently encode the conversation history and the tutor's response, extracting the `[CLS]` token from the final hidden state as a dense representation. These embeddings are projected through two separate linear layers and concatenated to form a joint feature vector, which is passed to a shallow feedforward neural network to predict one of three mistake identification categories. We trained this model using cross-entropy loss and the ADAMW optimizer, keeping the encoder frozen throughout training. To improve efficiency, we cached the embeddings as `.npz` files. The final output was restructured to match the original JSON format for evaluation, preserving conversation IDs, model names, tutor responses, and predicted mistake annotations.

2.4 Retrieval-Augmented Few-Shot Classification with LLM-as-a-Judge

Our final and most effective approach tackles the mistake identification task as a judgment problem, using a retrieval-augmented few-shot prompting strategy powered by large language models (LLMs). Instead of training a traditional classifier, we designed a modular pipeline built with LangChain. At its core, the system takes the full conversation history and the tutor's response, then prompts an LLM, specifically, `GPT-4o` to assess whether the tutor has correctly identified a mistake in the student's reasoning.

Figure 1 outlines the system architecture. We begin by embedding the conversation history and tutor responses from the MRBENCH training set using the `OpenAI Embedding Model`. These embeddings are stored in a persistent vector database using `ChromaDB`. With this setup, we construct a few-shot prompt template and use the LLM itself as a "judge" on the test data. At inference time, the system retrieves the top-k semantically similar examples and integrates them into the prompt.

Each prompt includes detailed labeling instructions, definitions for all possible labels (`Yes`, `No`, `To some extent`), and the full dialogue context. (see Appendix B for more information). To ensure clear and structured outputs, we use a `PydanticOutputParser` that enforces a strict schema and reliably extracts the label from the LLM's response. The pipeline also supports retries and incremental saving, making it robust and efficient for large-scale processing.

By combining relevant examples with a powerful instruction-following model, this method allows for nuanced mistake identification beyond simple clas-

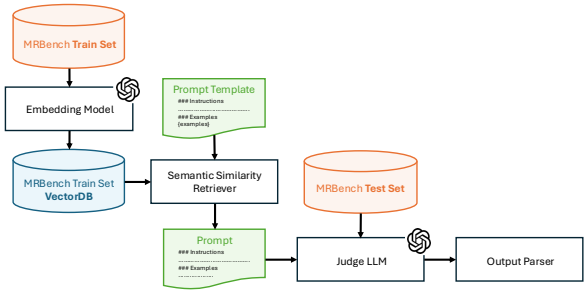

Figure 1: Pipeline of our final approach for mistake identification. The system takes tutor–student dialogue as input, retrieves relevant examples, constructs a structured prompt, and uses LLM to predict whether a mistake is identified. The output is parsed and saved.

sification. It requires no fine-tuning, generalizes well to new inputs, and showed improvements in both accuracy and qualitative evaluations compared to baseline methods. This highlights the effectiveness of prompt-based, retrieval-augmented approaches in educational and feedback-driven NLP tasks.

3 Dataset

We use the dataset introduced by Maurya et al. (2025), which includes both development and test splits. The development set consists of *300* dialogues from Macina et al. (2023) and Wang et al. (2024), each ending with a student utterance that reflects confusion or a mistake. Tutor responses, generated by seven large language model systems and human tutors (one in MathDial Macina et al. (2023), expert and novice in Bridge), are annotated along four pedagogical dimensions: (1) Mistake Identification, (2) Mistake Location, (3) Providing Guidance, and (4) Actionability. In total, the development set includes over *2,480* annotated responses.

The test set contains *200* dialogues with the same structure, but tutor identities are anonymized (for example, Tutor_1, Tutor_2), and no annotations are provided. This allows for blind evaluation of system outputs under the shared task setting.

3.1 Pre-processing

For the baseline systems, we apply extensive preprocessing to both the conversation history and tutor response texts. This includes converting text to lowercase, removing punctuation, stripping emojis, and cleaning URLs, HTML tags, and contractions. We also remove stopwords using the NLTK stopword list. All texts are passed through a unified normalization pipeline to reduce noise and ensure consistency. The labels for Mistake Identification are mapped to numeric values as follows: `No` $\rightarrow$ 0, `Yes` $\rightarrow$ 1, and `To Some Extent` $\rightarrow$ 2.

For our final approach, we additionally preprocess both the development and test sets so that each dialogue is reformatted into evenly paired exchanges between tutor and student, preserving the integrity of the back-and-forth interaction. During this process, we addressed two key issues. First, some conversations included greetings or closing phrases (e.g., “Hi”, “Thank you”) that did not contribute to the reasoning process. These were removed to maintain focus on educational content. Second, a few dialogues contained erroneous segments where the tutor responded to its own utterance without student input. These cases were consistently found to follow a correctly structured exchange and were manually removed (see Appendix A for examples).

This pre-processing step ensured a clean and consistent input format, enabling reliable downstream processing and model evaluation.

4 Evaluation and Results

We evaluated all four approaches on the **Track 1: Mistake Identification** test set using two evaluation schemes: **Strict** and **Lenient**, each reporting both *Macro F1* and *Accuracy*. In the strict setting, only exact matches with the gold labels are considered correct. In contrast, the lenient setting provides partial credit by treating `To some extent` as aligning with `Yes`, reflecting the fuzzy nature of pedagogical judgments in borderline cases. Table 1 summarizes the results.

As expected, the first baseline using pooled token embeddings and an ensemble of traditional classifiers (Approach 1) offered a modest starting point. This method, while straightforward, lacked the capacity to fully capture the nuances in dialogue-based reasoning.

Introducing token-level attention in Approach 2 led to a notable jump in performance. This suggests that modeling fine-grained interactions between the student’s dialogue and the tutor’s response helps the model better identify whether a mistake was correctly addressed. However, while this approach added depth to the representation, it still relied on relatively shallow modeling of the context.

Approach 3, which used frozen `[CLS]` embeddings from a sentence transformer combined with

Approach	Strict F1	Strict Acc	Lenient F1	Lenient Acc
Approach 1 (ML Ensemble)	0.446	0.657	0.637	0.754
Approach 2 (Token-Level Attention)	0.571	0.765	0.777	0.865
Approach 3 (CLS + MLP)	0.583	0.809	0.805	0.888
Approach 4 (Few-shot LLM + Retrieval)	**0.584**	**0.827**	**0.814**	**0.897**

Table 1: Performance of all four approaches on the BEA 2025 Mistake Identification test set under strict and lenient evaluation settings.

an MLP classifier, further improved performance. This indicates that sentence-level semantic representations, especially when paired with a focused classification head, can offer a stronger understanding of the overall pedagogical intent.

Our final method, Approach 4, which frames the task as a retrieval-augmented prompting problem with `GPT-4o`, achieved the best performance across all metrics. By retrieving semantically similar examples and using detailed, schema-guided prompts, the system benefited from both contextual grounding and the powerful instruction-following capabilities of modern LLMs. Notably, it showed strong results in both strict and lenient settings, highlighting its ability to make fine distinctions while still handling ambiguity in borderline cases effectively. Our final submission, achieved an official leaderboard rank of **37$^{\text{th}}$** among all participants.

5 Conclusion

We developed and evaluated four approaches for the BEA 2025 Shared Task **Track 1: Mistake Identification**, culminating in a retrieval-augmented few-shot prompting system using GPT-4o. While our initial baselines used traditional classifiers over pretrained embeddings, the final system reframed the task as a structured judgment problem, combining semantically retrieved examples, instruction-driven prompting, and schema-constrained output parsing.

This approach consistently outperformed all baselines in both strict and lenient evaluations, achieving a strict Macro F1 of *0.584* and a lenient accuracy of *0.897*. It was particularly effective at capturing nuanced pedagogical feedback, highlighting the strength of LLM-based reasoning when guided by relevant context. Our submission ranked 37$^{\text{th}}$ on the official leaderboard, demonstrating the competitiveness of our method.

These results show that retrieval-augmented prompting offers a scalable and effective solution for assessing complex teaching behaviors in AI tutors. Future work could explore more adaptive example selection, multi-turn consistency, and alignment with broader goals such as helpfulness and instructional fairness.

Limitations

While our final system achieved the best performance among all submitted approaches, it still has several limitations that suggest promising directions for future work.

Limited Diversity in Retrieved Examples The effectiveness of our retrieval-augmented prompting pipeline depends heavily on the quality and coverage of the example pool. Since we rely on a fixed set of annotated training examples, the system may struggle with out-of-distribution dialogues or question types that are underrepresented in the retrieval set. Moreover, retrieval is based solely on static embedding similarity from OpenAI embeddings, without adapting to the context or emphasizing specific pedagogical traits.

Lack of Multi-Turn Dialogue Modeling Each input is treated as a standalone conversation-response pair, with no memory of earlier tutor turns or evolving dialogue context. This limits the system's ability to track learning progression or take prior feedback into account. Modeling dialogue history explicitly—through dialogue state tracking or memory-based retrieval—could improve consistency and pedagogical depth in multi-turn interactions.

Simplified Output Format Although the use of a structured parser ensures consistency, it restricts the model to selecting a single label per example. It does not capture uncertainty, nuanced justifications, or cases where multiple labels might apply. Extending the output to include rationales or confidence

scores could make evaluations more informative and reflective of real-world ambiguity.

Scalability and Cost Constraints Inference with frontier models like GPT-4o is computationally intensive and dependent on external APIs, which introduces latency, cost, and rate-limit challenges. These constraints pose barriers to deployment in low-resource settings or real-time tutoring applications, where efficiency and scalability are critical.

References

Ekaterina Kochmar, Kaushal Kumar Maurya, Kseniia Petukhova, KV Aditya Srivatsa, Anaïs Tack, and Justin Vasselli. 2025. Findings of the BEA 2025 shared task on pedagogical ability assessment of AI-powered tutors. In *Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications*.

Jakub Macina, Nico Daheim, Sankalan Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. MathDial: A dialogue tutoring dataset with rich pedagogical properties grounded in math reasoning problems. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5602–5621, Singapore. Association for Computational Linguistics.

Kaushal Kumar Maurya, Kv Aditya Srivatsa, Kseniia Petukhova, and Ekaterina Kochmar. 2025. Unifying AI tutor evaluation: An evaluation taxonomy for pedagogical ability assessment of LLM-powered AI tutors. pages 1234–1251, Albuquerque, New Mexico. Association for Computational Linguistics.

Rose Wang, Qingyang Zhang, Carly Robinson, Susanna Loeb, and Dorottya Demszky. 2024. Bridging the novice-expert gap via models of decision-making: A case study on remediating math mistakes. pages 2174–2199, Mexico City, Mexico. Association for Computational Linguistics.

A Preprocessing Examples

This appendix list some issues which are fixed during pre-processing of dataset.

In the following example from the development set, the initial student message is a casual greeting that disrupts the expected alternating structure of the dialogue. To maintain structural integrity and ensure an even number of turns between tutor and student, such non-essential messages are removed during preprocessing.

```
[
'Student: okey',
'Tutor: What is 25 minus 18?',
'Student: 8'
]
```

In the example below, the final tutor response erroneously mimics the student's explanation, as if the tutor is responding to itself rather than engaging with the student. This type of error breaks the natural flow of the dialogue and was manually identified and removed during preprocessing to ensure accurate tutor-student interaction.

```
[
'Tutor: Hi, could you please provide a
↪  step-by-step solution for the question
↪  below? The question is ...',
'Student: Samantha buys 4 toys at $12.00 each.
↪  For each pair of toys...',
'Tutor: I added the two amounts together to get
↪  a total of $36.00 + $6.00 = $42.00.'
]
```

In cases like the example below, the tutor's prompt is split across multiple turns, breaking the intended question into separate messages. To preserve the coherence of the dialogue and maintain a consistent turn-taking structure, such fragmented tutor responses are merged into a single utterance by concatenating the strings.

```
[
'Tutor: Hi, could you please provide a
↪  step-by-step solution for the question
↪  below? The question is: Tyson decided to
↪  make muffaletta sandwiches for ...,
'Tutor: How many pounds of meat are needed for
↪  each sandwich?',
'Student: Each sandwich requires 1 pound of
↪  meat and 1 pound of cheese.',
'Tutor: What is the cost of 1 pound of meat?',
↪  'Student: The cost of 1 pound of meat is
↪  $7.00.'
]
```

B Prompt Engineering

```
"""
You will be shown a short educational "Conversation" between a tutor and a student, including the
↪  student's solution and the tutor's follow-up "Response". Your task is to judge whether the
↪  tutor's response successfully **identifies a mistake** in the student's reasoning.

### Instructions
1. Read the entire dialogue to understand the context of the student's solution.
2. Focus on whether the tutor's response explicitly or implicitly calls out an error.
3. Reply **only** with one of the labels: `Yes`, `To some extent`, or `No`.

### Labels
- `Yes`: The mistake is clearly identified/recognized in the tutor's response. The tutor implicitly
↪  or explicitly points out the error in the student's reasoning.
- `No`: The tutor's response does not identify any mistake in the student's reasoning. The tutor's
↪  response is either irrelevant or does not address the student's solution.
- `To some extent`: The tutor's response suggests that there may be a mistake, but it sounds as if
↪  the tutor is not certain.

### Format Instructions:
{format_instructions}
Return only the classification label without any additional commentary or extraneous details.

### Examples
{examples}

## Mistake Identification
### Conversation
{conversation}

### Response
{response}
"""
```

Figure 2: Prompt for LLM which is used a judge in Retrieval-Augmented Few-shot classification approach