BD at BEA 2025 Shared Task: MPNet Ensembles for Pedagogical Mistake Identification and Localization in AI Tutor Responses

Shadman Rohan Ishita Sur Apan Muhtasim Ibteda Shochcho Md Fahim
Mohammad Ashfaq Ur Rahman AKM Mahbubur Rahman Amin Ahsan Ali
Center for Computational & Data Sciences, Independent University, Bangladesh (IUB)
{shadmanrohan, ishitasurapan, sho25100, fahimcse381}@gmail.com
{imashfaqfardin}@gmail.com, {akmmrahman, aminail}@iub.edu.bd

Abstract

We present Team BD's submission to the BEA 2025 Shared Task on Pedagogical Ability Assessment of AI-powered Tutors, under Track 1 (Mistake Identification) and Track 2 (Mistake Location). Both tracks involve three-class classification of tutor responses in educational dialogues – determining if a tutor correctly recognizes a student's mistake (Track 1) and whether the tutor pinpoints the mistake's location (Track 2). Our system is built on MPNet, a Transformer-based language model that combines BERT and XLNet's pre-training advantages. We fine-tuned MPNet on the task data using a class-weighted cross-entropy loss to handle class imbalance, and leveraged grouped cross-validation (10 folds) to maximize the use of limited data while avoiding dialogue overlap between training and validation. We then performed a hard-voting ensemble of the best models from each fold, which improves robustness and generalization by combining multiple classifiers. Our approach achieved strong results on both tracks, with exact-match macro-F1 scores of approximately 0.7110 for Mistake Identification and 0.5543 for Mistake Location on the official test set. We include comprehensive analysis of our system's performance, including confusion matrices and t-SNE visualizations to interpret classifier behavior, as well as a taxonomy of common errors with examples. We hope our ensemble-based approach and findings provide useful insights for designing reliable tutor response evaluation systems in educational dialogue settings.

1 Introduction

Effective intelligent tutoring systems need to be able to recognize and address student mistakes during interactions. To evaluate such capabilities in automated systems, the BEA 2025 Shared Task introduced a multi-dimensional assessment of AI tutor responses. In particular, Track 1 focuses on whether a tutor's response identifies the student's mistake, and Track 2 on whether it locates the mistake in the student's answer. Each track is framed as a three-way classification: the tutor either fully recognizes/locates the error ("Yes"), partially or uncertainly does so ("To some extent"), or fails to do so ("No"). These pedagogically motivated categories draw from prior frameworks in educational dialogue analysis—for example, Mistake Identification corresponds to the student understanding dimension in Tack and Piech's schema (Tack and Piech, 2022b) and correctness in other tutoring evaluation schemata, reflecting how well the tutor acknowledges the student's misconception.

Assessing tutor responses along such dimensions is challenging due to the nuanced and subjective nature of pedagogical feedback. For instance, different studies have used varied measures (e.g., "speaking like a teacher," "understanding the student," etc.) to judge tutor responses. The BEA 2025 shared task addresses this gap by defining clear categories and metrics for evaluation (Kochmar et al., 2025). However, even with a fixed taxonomy, classifying responses correctly remains non-trivial: tutors may implicitly acknowledge an error without stating it outright, or they might hint at the error's location in vague terms. Distinguishing between a definite "Yes" and a tentative "To some extent" thus requires subtle interpretation of language.

In this paper, we describe Team BD's ensemble-based MPNet system for automating the annotation of mistake identification and mistake location in AI-tutor responses. MPNet, a pretrained Transformer model that uses masked and permuted language modeling to capture token dependencies, was chosen as our backbone for its strong generalization capabilities compared to earlier models like BERT, XLNet, and RoBERTa. To address the limited size of the labeled data (approximately 2.5 K examples) and inherent class imbalance, we fine-tuned MPNet with a class-weighted cross-entropy loss and trained ten separate models using grouped cross-

Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications, pages 1266–1277
July 31 - August 1, 2025 ©2025 Association for Computational Linguistics

validation—grouped by dialogue to prevent context leakage—and then combined the top-performing model from each fold through hard-voting. This ensemble strategy greatly improved robustness and generalization, leading to high accuracy and macro-F1 scores on both the mistake identification and mistake location tracks. Our error analysis using confusion matrices and t-SNE visualizations revealed consistent misclassification patterns, notably confusing fully recognized with partially acknowledged mistakes. We created a taxonomy of common error types with examples to aid future refinements.

2 Related Work

Evaluation of Tutor Responses: The task of judging tutor or teacher responses in educational dialogues has recently garnered attention. Tack and Piech (Tack and Piech, 2022a) introduced the AI Teacher Test to measure the pedagogical ability of dialogue agents, proposing dimensions such as whether the agent understands the student's error and provides helpful guidance. Following this, the BEA 2023 Shared Task (Tack et al., 2023) focused on generating AI teacher responses (rather than classification), where models like GPT-3 and Blender were challenged to produce tutor-like feedback. The BEA 2025 Shared Task (Kochmar et al., 2025) moves a step further by creating a benchmark dataset of tutor responses annotated along multiple pedagogical dimensions. The dataset leverages dialogues from **MathDial** (Macina et al., 2023) and **Bridge** (Maurya et al., 2025), two collections of student-tutor interactions in the math domain. Each tutor response in these dialogues was labeled by experts as to whether it identifies the student's mistake, pinpoints the mistake's location, provides guidance, and offers actionable next steps. Such multi-faceted annotation of tutor feedback is relatively novel; it connects to earlier work on dialogue act classification (Maurya et al., 2025) in that both involve categorizing utterances, but here the labels are pedagogical quality ratings rather than communicative intent.

Ensemble Methods in NLP: Classic studies, such as Dietterich's work on ensemble methods, demonstrated that an ensemble of diverse classifiers can correct individual models' errors and reduce variance (Dietterich, 2000). For instance, (Ovadia et al., 2019) and (Gustafsson et al., 2020) found that deep ensembles improve reliability under dataset shift. In shared task and kaggle competitions, top teams often resort to model ensembling to squeeze out some additional performance. These benefits come at the cost of increased computational overhead. Our approach aligns with this trend, as we build an ensemble of 10 MPNet-based classifiers (from cross-validation folds) to tackle the classification of tutor responses.

Dialogue and Educational NLP: Related to our work is research on grammatical error detection and correction, where systems identify mistakes in student-written text. Notably, (Ng et al., 2014) and (Bryant et al., 2019) have contributed significantly to this field. However, our task differs in that the "mistakes" are conceptual or procedural errors in a problem solution, and we are evaluating the tutor's response to those errors rather than directly analyzing the student's text. Another line of relevant work is on student response analysis in tutoring systems, where the goal is to classify student answers as correct, incorrect, or incomplete. (Dzikovska et al., 2013) explored this in the context of the SemEval-2013 Task 7. In our case, the roles are reversed—we classify the tutor's replies. We also draw on insights from educational dialogue analysis: studies like (Daheim et al., 2024) examined tutor responses for targetedness and actionability, which correspond to our Track 2 and Track 4 tasks. These studies emphasize the subtle linguistic cues that indicate whether a tutor has pinpointed an error (e.g., referencing a specific step in the student's solution) or just given generic feedback.

In summary, our work is situated at the intersection of dialogue evaluation and text classification. We build upon the shared task's provided taxonomy (SIGEDU, 2025) and prior educational NLP research, employing modern Transformer models and ensemble techniques known to be effective in such tasks.

3 Data and Task Definition

Task Definition: Tracks 1 and 2 are classification tasks applied to tutor responses in a dialogue. Based on the previous conversation history between students and tutors, in Track 1 (Mistake Identification), the system must determine if the tutor's response indicates recognition of the student's mistake. In Track 2 (Mistake Location), the system judges if the tutor points out the specific location or nature of the mistake in the student's solution. Both tasks share the same label set: **Yes, To some extent,**

Model	Macro-F1 Score
BERT-large	0.6851
DeBERTa	0.6845
MPNet (selected)	**0.6975**

Table 1: 10 fold Cross-validation Macro-F1 scores for different Transformer models on the track 1 development set. MPNet achieves the highest score.

or **No**. Because these categories can be nuanced, the shared task also defined a lenient evaluation where "Yes" and "To some extent" are merged, but our system is trained on the full 3-class distinction (exact evaluation).

Dataset: The training (development) data provided by the organizers consists of annotated educational dialogues in mathematics, drawn from the MathDial and Bridge datasets. Each dialogue includes a student's attempt at a math problem (containing a mistake or confusion) and one or more tutor responses (from either human tutors or various LLMs such as Mistral, Llama, GPT-4, etc. acting as tutors). Each tutor response is annotated with the three-class labels for all four dimensions (Tracks 1–4). In total, the development set contains 300 conversation history and over 2,480 tutor responses with annotations. On average, each dialogue context yields 8–9 different tutor responses (one from each of several tutor sources), which were all annotated. The test set is constructed in the same way but uses held-out dialogues and responses—both the ground-truth labels and the tutors' identities are hidden.

The development set for both **Track 1 (Mistake Identification)** and **Track 2 (Mistake Location)** consists of the same 300 dialogues and 2,476 tutor responses. However, the label distributions differ between tracks due to the nature of the classification tasks. The underrepresentation of the *To some extent* class in both tracks poses challenges for model learning.

4 Methodology

4.1 Preprocessing

All tutor responses and conversation histories were first lowercased (while preserving punctuation) to ensure consistent casing.

To standardize and sanitize the responses, we applied a series of targeted cleaning steps:

- **Extra Info Removal**: Eliminated any metadata or annotations not part of the tutor's actual reply.
- **Appended Dialogue Trimming**: Removed follow-up conversational turns that were appended after the original tutor response (e.g., speculative follow-up questions or acknowledgments).
- **Code Abstraction**: Replaced Python code blocks with the placeholder «python code» to retain structural intent while abstracting away executable details.
- **Punctuation Cleanup**: Stripped redundant or mismatched punctuation (e.g., extraneous quotes or dashes) that might confuse the tokenizer or the model.

Table 4 provides a summary of how many instances were affected by each category. We observed that models such as **Phi-3** and **Llama-3.1-405B** required the most extensive preprocessing.

Finally, each input example—consisting of the conversation history, cleaned response, and separator tokens—was constrained to a maximum of 512 MPNet tokens. In cases where the input exceeded this limit, we removed the low-value content (e.g., greetings or small talk) from the conversation history to retain the most relevant context.

4.2 Language Model Finetuning

In our experiments, we utilize transformer-based pretrained language models (LMs). Since these models may lack task-specific contextual knowledge, we fine-tune them on our target tasks to improve performance.

To begin, we consider a pretrained language model denoted as ϕ_{LM}. Each tutor's response after preprocessing T is input to the model, yielding a sequence of tokens $T = \{t_{\text{[CLS]}}, t_1, t_2, \dots, t_n\}$ along with their corresponding layer-wise hidden representations $H^l = \{h^l_{\text{[CLS]}}, h^l_1, h^l_2, \dots, h^l_n\}$.

In our setup, we use the hidden representation of the [CLS] token from the final layer as the sentence-level representation of the input T, defined as:

$$h_T = \phi_{\text{LM}}(T)^L_{\text{[CLS]}} = H^L_{\text{[CLS]}}$$

This representation h_T is then passed through a classification head to produce the prediction. The classification head consists of a dropout layer *Drop* followed by a linear transformation:

$$p = W \cdot \text{Drop}(h_T) + b$$

Finally, we use a cross-entropy loss function to update the parameters of the language model ϕ_{LM} during training.

4.3 Grouped Cross-Validation

We employ *group cross-validation* to ensure robust evaluation and mitigate overfitting. In this approach, each dialogue (or group of dialogues) is entirely assigned to either the training or validation set within each fold, preventing shared context between the training and validation sets.

For each fold $f \in \{1, 2, \ldots, k\}$, we define the training and validation sets as $\mathcal{G}_{\text{train}}^{(f)}$ and $\mathcal{G}_{\text{val}}^{(f)}$, respectively, where each set contains whole dialogues (or groups) with no overlap. We monitor the model's performance on the validation set using the *macro-averaged F1 score* (macro-F1), which provides a balanced measure of performance across classes. For each fold, we save the model checkpoint that achieves the highest macro-F1 score on the validation set.

The final performance of the model is computed by aggregating the macro-F1 scores across all k folds.

4.4 Ensembling Strategy

To enhance model performance, we employed an *ensembling strategy* where the top-performing models from each fold were combined using hard voting. Specifically, for each track, we had a total of $N = 10$ models (one from each fold).

Let $\hat{y}_i^{(f)}$ denote the prediction of the model from fold f for the i-th sample, where $f \in \{1, 2, \ldots, N\}$. The final prediction $\hat{y}_i$ for each sample i was determined by majority vote:

$$\hat{y}_i = \text{mode}(\hat{y}_i^{(1)}, \hat{y}_i^{(2)}, \ldots, \hat{y}_i^{(N)})$$

In the case of a tie, the tie-breaking rule was based on the average softmax confidence across all models. Let $s_i^{(f)}$ denote the softmax output (confidence) of the f-th model for the i-th sample. If a tie occurs, the final prediction is chosen as:

$$\hat{y}_i = \arg\max \left(\frac{1}{N} \sum_{f=1}^{N} s_i^{(f)} \right)$$

Ensembling helps to reduce variance and correct individual model biases, leading to more robust predictions. Our ensembling approach improved the macro-F1 score by 2–3 points over the performance of individual models.

5 Experimental Setup

5.1 Implementation Details

Model Selection In our experiments, we compared several such models—including BERT-large, DeBERTa, and MPNet—on a held-out subset of the training data. Among these, MPNet achieved the best macro-F1 score (see Table 1), and was thus selected as our backbone. For implementation details, including software, packages, and hardware setup, see Appendix A.

Model Hyperparameters

We used the AdamW optimizer with a learning rate of 2×10^{-5}, selected through preliminary experiments on a held-out validation set. This setting outperformed alternative learning rates such as 1×10^{-5} and 3×10^{-5} in terms of macro-F1. A linear learning rate decay schedule was used, along with early stopping based on validation macro-F1 (patience = 5 epochs). We trained with a batch size of 32 and applied a dropout rate of 0.1 in the classification head. No gradient accumulation was used.

Handling Class Imbalance

To mitigate class imbalance, we used a class-weighted cross-entropy loss, where the weight for each class c was computed as:

$$w_c = \frac{N}{K \cdot n_c}$$

with N being the total number of samples, K the number of classes, and n_c the count for class c. This formulation emphasizes underrepresented classes without overly penalizing frequent ones.

For Track 1 (Mistake Identification), class distributions were skewed toward "Yes" (1932), compared to "No" (370) and "To some extent" (174). We thus used the weight vector:

$$[w_{\text{No}}, w_{\text{Some}}, w_{\text{Yes}}] = [1.0, 3.0, 0.5]$$

to boost recall for the rare "Some extent" class and mildly down-weight the majority class.

In Track 2 (Mistake Location), the frequencies were: "Yes" (1504), "No" (732), and "To some extent" (240). Based on this, we used:

$$[w_{\text{No}}, w_{\text{Some}}, w_{\text{Yes}}] = [0.8, 2.2, 0.9]$$

These weights, derived from inverse class frequencies and lightly tuned, improved macro-F1 by reducing systematic underprediction of minority classes. Although not extensively optimized, this approach provided consistent performance gains across both tracks.

Track	Macro F1	Accuracy
Track 1 – Mistake Identification		
Best (BJTU)	0.718	0.862
Ours (Test)	0.711	0.877
Ours (CV aggregate)	0.685	0.869
Track 2 – Mistake Location		
Best (BLCU-ICALL)	0.598	0.768
Ours (Test)	0.554	0.714
Ours (CV aggregate)	0.560	0.700

Table 2: Comparison of our system's macro-F1 and accuracy with top leaderboard scores on both tracks.

5.2 Evaluation Metrics

Following the shared task guidelines, we report both Accuracy and Macro F1. Macro F1, the unweighted average of per-class F1 scores, is emphasized due to class imbalance. We monitored performance using these metrics on the validation set during training and evaluated on the aggregated development set using cross-validation predictions. Final test metrics were provided by the organizers. We focus on exact 3-class classification; lenient 2-class metrics (merging "Yes" with "To some extent") were higher but are omitted here for brevity.

6 Result and Analysis

6.1 Main Result

To contextualize our system's performance, we compared it against the top submissions from the official shared task leaderboard. On Track1 (Mistake Identification), our model achieved a macro-F1 of 0.711 on the test set, placing 5th out of 44 participating teams. The top-ranked system (BJTU) achieved a macro-F1 of 0.718, indicating that our system performs competitively, within 0.7 points of the best result. For Track2 (Mistake Location), our system scored 0.554 macro-F1 on the test set, ranking 7th out of 31 teams. The highest score on this track was 0.598, obtained by BLCU-ICALL. While our model trails behind the top result by approximately 4.4 points in macro-F1, it still exceeds the median leaderboard performance.

Our system achieved higher accuracy than the top Track 1 system (0.877 vs. 0.862), suggesting stronger performance on dominant classes, albeit with slightly lower balance across all classes.

Even though our system performs well, a closer examination of its errors provides insights into its decision-making and the task's inherent difficulty. We carried out an error analysis on the development set predictions, focusing on confusion patterns and the nature of misclassified cases.

6.2 Class-Level Performance Analysis

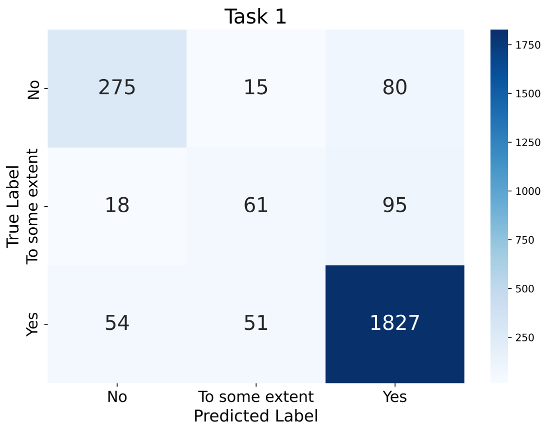

Figure 1: Confusion matrix for Track 1 (Mistake Identification) on the development set. The model shows strong performance on the "Yes" class but has difficulty distinguishing partial acknowledgment ("To some extent").

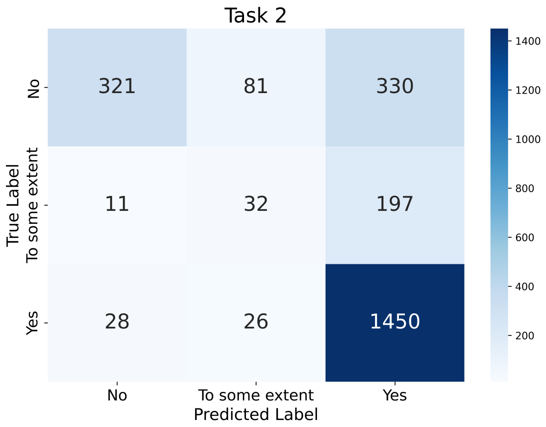

Figure 2: Confusion matrix for Track 2 (Mistake Location). The model maintains high accuracy on explicit localizations ("Yes") but misclassifies many "To some extent" and "No" cases, highlighting the subtlety of location inference.

To gain insight into how well our system distinguishes among the three pedagogical feedback categories, we analyze confusion matrices for both tasks. Figures 1 and 2 visualize model predictions against gold labels on the development set for Track 1 (Mistake Identification) and Track 2 (Mistake Location), respectively.

In Track 1 (Figure 1), the model performs strongly on the "Yes" class, correctly identifying 1,827 instances, with relatively low misclassification into the "No" (54) and "To some extent" (51) classes. The "No" class is also well captured with 275 correct predictions and few false positives. The model struggles more with the "To some extent" category: 61 were correctly predicted, but 113 were misclassified as either "No" or "Yes." This aligns with our earlier claim that “To some extent” lies on a subjective continuum and is more difficult to pin down categorically.

For Track 2 (Figure 2), a similar trend emerges. The model again shows high accuracy on “Yes” (1,450 correct), but struggles to distinguish “To some extent,” which is often misclassified as “Yes” (197 cases) or “No” (11 cases). Notably, the “No” class is less cleanly separated in Track 2 compared to Track 1, with 330 examples misclassified as “Yes.” This may suggest that tutors sometimes appear to reference an error without pinpointing its location, confusing the model’s judgment.

Overall, these confusion matrices illustrate the asymmetric difficulty across classes. "Yes" responses are most reliably predicted due to their clearer, more direct language. "To some extent" predictions remain a challenge, particularly when tutors use indirect or hedging phrasing that blurs the line between partial and full error acknowledgment or localization.

6.3 Embedding Space Insights

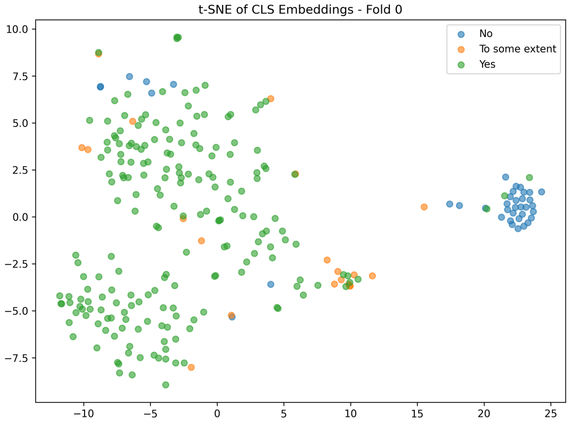

Figure 3: t-SNE projection of [CLS] embeddings from the held-out fold (Fold 0) for Track 1 (Mistake Identification), colored by true label. “Yes” and “To some extent” examples are scattered and intermixed, whereas “No” forms a more compact cluster, indicating lower intra-class variation.

To better understand the internal representations learned by our model, we applied t-SNE (van der Maaten and Hinton, 2008) to the [CLS] embeddings from the final Transformer layer. These projections reveal how the model organizes tutor responses in the embedding space across folds and classes.

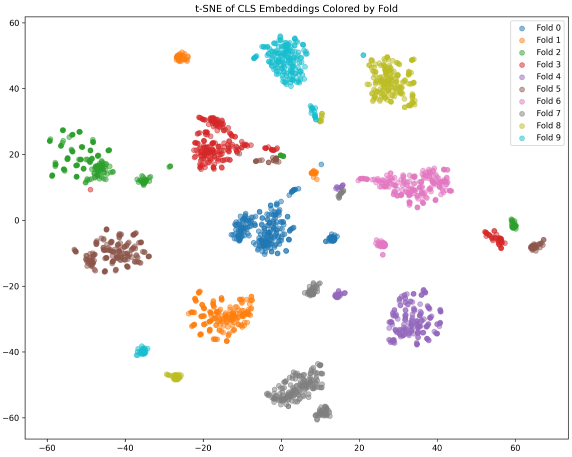

Figure 4: t-SNE projection of [CLS] embeddings from MPNet models across all 10 cross-validation folds for Track 1 (Mistake Identification). Each point represents a tutor response from a held-out fold, colored by fold ID. The emergence of distinct clusters suggests that each fold-specific model learns a consistent but fold-specific embedding subspace, reflecting representational diversity across the ensemble.

Figure 4 shows the t-SNE projection of the [CLS] embeddings across all ten cross-validation folds, with points colored by fold ID. We observe that embeddings from each fold tend to form compact, well-separated clusters. This indicates that while training on different subsets, each fold-specific model learns fold-consistent but distinct representations. The tightness of these clusters also suggests good embedding stability and coherence across training runs.

Figure 3 presents the t-SNE visualization for the held-out fold (Fold 0), this time colored by the true label. Unlike the per-fold visualization, class-level structure is less distinct: the “Yes” and “To some extent” responses are widely dispersed and often intermingle, suggesting overlapping semantic characteristics. In contrast, the “No” class forms a more compact group, indicating that tutor responses with no recognition of error share more consistent linguistic patterns. This aligns with our earlier findings that “Yes” and “To some extent” are harder to separate, as they exist on a continuum of acknowledgment.

Together, these visualizations support our earlier confusion matrix results and highlight the challenge of distinguishing nuanced pedagogical feedback categories based solely on language.

6.4 Error Taxonomy

To better understand where the model fails, we analyzed misclassified responses from both tasks and developed a taxonomy of recurring error types, summarized in Table 3. These categories reflect systematic issues in how the model interprets pedagogical language.

False Negatives (Missed Signal). These errors occur when the model fails to recognize that the tutor has identified or located a mistake, typically labeling the response as “No” or “To some extent” instead of “Yes.” Such cases often involve subtle cues like rhetorical questions or light correction phrasing (e.g., “Can you check the multiplication again?”), which the model may under-interpret.

False Positives (Over-interpretation). Here, the model predicts “Yes” even when the tutor does not provide evidence of error recognition. This often results from over-interpreting generic encouragement (e.g., “Let’s try another one.”) or positive sentiment as pedagogical feedback.

Partial–Full Confusion. A frequent source of confusion is the distinction between full and partial identification or localization. Indirect language such as “You’re close, just verify your subtraction” may be intended as partial feedback, but the model may treat it as a complete identification.

Hedged Language Confusion. Tutors often use polite or indirect language (e.g., “Maybe revisit the earlier step?”), especially in educational settings. Such hedging may obscure intent, leading the model to underestimate the strength of the feedback signal.

Contextual Miss. Some misclassifications stem from failing to use conversational history. For instance, if a tutor’s comment refers to an earlier incorrect step, the model may mislabel it when that context is not incorporated effectively.

Template Bias. We also observed that the model sometimes over-relies on surface patterns seen during training. For example, statements like “Great work!” may be incorrectly classified as “Yes” due to template bias, even when no mistake is acknowledged.

These error categories offer valuable insight into the linguistic and contextual challenges of the task. They suggest that improvements in discourse modeling, uncertainty handling, and pragmatic language understanding could further enhance performance.

From the above taxonomy, we see that many of the model’s mistakes correspond to understandable difficulties. False negatives often involved indirect tutor feedback—the tutor recognized the mistake but phrased it as a question or hint, requiring inference to identify it as an acknowledgment of error. Our model sometimes took such tentative language at face value and labeled it as if the tutor did nothing. False positives, on the other hand, were cases where the tutor’s response had reassuring or neutral language that the model mistook for a sign of recognizing a mistake. For example, tutors might say “Let’s double-check that” even when the student was correct (encouraging the student, not pointing an error), and the model erroneously flagged it as identifying an error.

The partial vs. full confusion category was the most prevalent error type. This reflects the inherent ambiguity of the “To some extent” class—even human annotators might differ on these in some cases. Our model would sometimes collapse it into one of the binary decisions (“Yes” or “No”) depending on slight wording differences. In some cases, the model predicted “To some extent” when the tutor had actually pinpointed the error but perhaps in a subtle way; in others, it predicted “Yes” for a tutor response that was only hinting. This suggests that improving the model’s understanding of nuanced language (perhaps via better context usage or training on more examples of hedging) could help.

We also found that ambiguous wording and polite phrasing (common in educational settings) posed challenges. Phrases like “Maybe check that again” require contextual understanding—they might indicate an error without explicit wording. Our model did catch many of these, but not all. Some errors could be attributed to the model’s lack of world knowledge or reasoning; for example, if a tutor says “Remember the formula for area,” the model needs to infer that the student likely made a mistake related to area calculation and that the tutor is hinting at it—a level of reasoning beyond surface text.

In summary, the error analysis reveals that while our ensemble is effective, there is room for improvement in handling borderline cases and understanding implicit signals. These findings guided us in considering potential enhancements, as discussed next.

Error Type	**Description**	**Example Scenario**
False Negative (Missed Signal)	Tutor indicates or locates a mistake, but the model predicts "No" or "To some extent."	*Tutor:* "Can you check the multiplication again?" Gold: Yes → Pred: To some extent
False Positive (Over-interpretation)	Model predicts "Yes" despite the tutor giving no error feedback.	*Tutor:* "Let's try another one." Gold: No → Pred: Yes
Partial–Full Confusion	Confuses indirect hints as full identification, or subtle localization as partial.	*Tutor:* "You're close, just verify your subtraction." Gold: To some extent → Pred: Yes
Hedged Language Confusion	Tutor's suggestion is misread due to polite phrasing or indirect cues.	*Tutor:* "Maybe revisit the earlier step?" Gold: Yes → Pred: To some extent
Contextual Miss	Misclassification caused by ignoring or misusing multi-turn context.	*Tutor:* Feedback depends on an earlier step, but the model misses the reference.
Template Bias	Model favors phrases resembling training-time patterns, even when semantically incorrect.	*Tutor:* "Great work!" with no correction. Model assumes this implies error recognition.

Table 3: Taxonomy of common misclassification errors in both tasks, with representative examples.

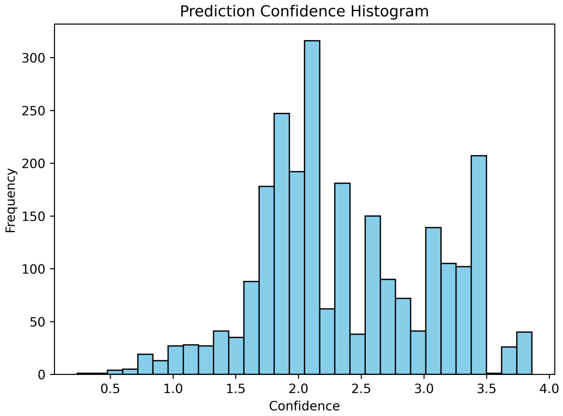

Figure 5: Histogram of prediction confidence values for Track 1 (Mistake Identification). Most predictions fall within a mid-confidence range.

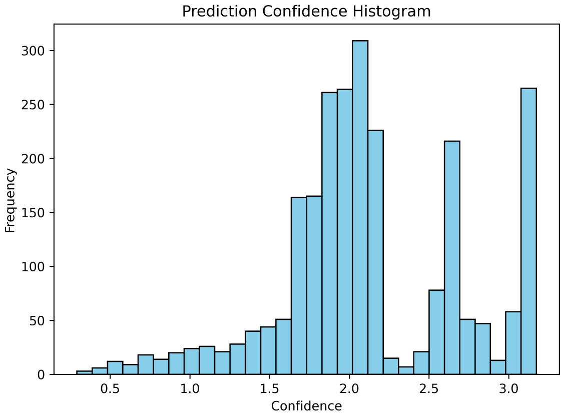

Figure 6: Histogram of prediction confidence values for Track 2 (Mistake Location). A similar mid-range clustering pattern is observed, with some extreme confidence peaks.

6.5 Confidence Distribution and Calibration

To further investigate the model's decision-making behavior, we analyzed its prediction confidence across classes and tasks. Figures 5 and 6 present histograms of predicted confidence scores for Track 1 and Track 2, respectively. These reflect the model's certainty in its predictions across the development set.

In both tasks, the confidence distribution is skewed toward the middle range (1.5–3.0), with multiple local peaks. This suggests that while the model often makes moderately confident predictions, it does not frequently commit to extremely low or high confidence outputs. The spiked clusters in Track 2 (Figure 6) hint at calibration artifacts possibly introduced by ensemble averaging. Despite ensemble smoothing, we still observe confidence saturation for some predictions near 3.5, particularly on easier instances.

To better understand class-specific behavior, we examined boxplots of prediction confidence grouped by predicted label (Figures 7 and 8). In both tasks, predictions labeled as "No" tend to have higher median confidence compared to "To some extent," reflecting that the model is more certain when asserting a complete absence of error. Predictions for "To some extent" exhibit both lower median confidence and greater spread—supporting earlier findings that this category is harder to classify due to its inherent ambiguity. Interestingly, in Track 1, "Yes" predictions also show relatively high confidence, indicating that the model treats full error recognition as a more decisive signal than partial acknowledgment.

These confidence trends are broadly aligned with our confusion matrix analysis: "To some extent" is not only the most frequently confused class but also the one with the least confident predictions. This

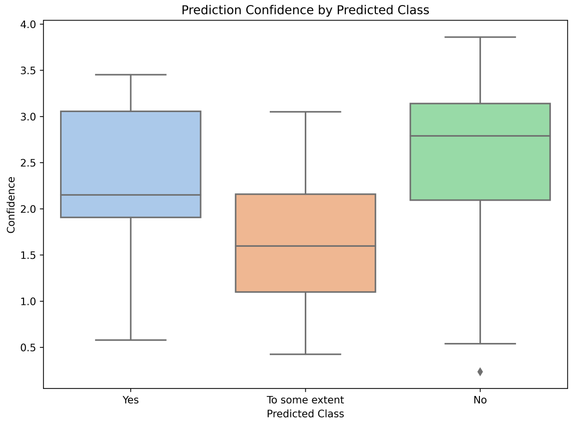

Figure 7: Boxplot of confidence by predicted class (Track 1). Predictions labeled "To some extent" tend to have lower median confidence.

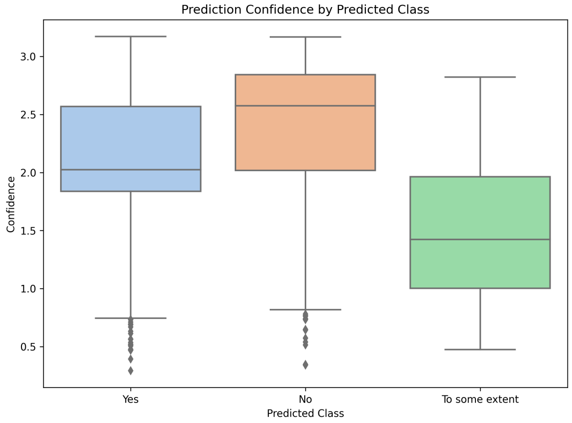

Figure 8: Boxplot of confidence by predicted class (Track 2). "No" and "Yes" predictions show higher confidence than "To some extent."

highlights a key challenge in pedagogical feedback modeling—the need to model uncertainty explicitly, especially in borderline cases. Future work could explore temperature scaling or Bayesian ensembling to better calibrate prediction confidence, particularly for interpretability in high-stakes educational settings.

7 Conclusion

This paper presents Team BD's ensemble-based MPNet system for the BEA 2025 Shared Task on Mistake Identification and Location in tutor responses. By fine-tuning MPNet with class-weighted loss and grouped cross-validation, we addressed data imbalance and maximized the use of training data, achieving high accuracy and macro-F1 scores on both Track 1 and Track 2. Extensive analyses show that, while the model reliably handles clear-cut error recognition, it struggles with borderline cases involving partial acknowledgment, as evidenced by embedding-space visualizations and a taxonomy of common errors. Future work will explore multi-task learning across evaluation dimensions, leverage larger language models or adapter-based methods to incorporate LLM knowledge, and improve calibration and domain-specific contextual understanding to enhance system reliability and interpretability.

8 Limitations

Despite the strong results achieved by our ensemble MPNet-based system, several limitations warrant discussion:

Confidence Calibration: Our ensemble exhibits poor calibration, often assigning high confidence to incorrect predictions—problematic for intervention-triggering systems. We did not apply calibration methods due to time constraints. Adaptive Temperature Scaling (ATS), a recent post-hoc technique, improves token-level calibration by 10–50% across benchmarks (Xie et al., 2024), and merits future exploration.

Label Ambiguity: The line between "Yes" and "To some extent" is subjective, with some errors stemming from annotation uncertainty rather than model failure, thus limiting performance. Modeling the task as ordinal or probabilistic may better capture this continuum; ordinal methods have been proposed for similar label structures (Zhang et al., 2023).

Model Scope and Efficiency: MPNet-base lacks domain-specific specialization for educational dialogue, which may limit its ability to handle nuanced interactions. Exploring a larger, domain-adapted backbone or a multitask learning setup could enhance performance and is a promising direction for future work.

Acknowledgments

We thank the BEA 2025 Shared Task organizers for their efforts and Center for Computational & Data Sciences(CCDS) for supporting this work. Our code and models are available at: https://github.com/ShadmanRohan/team-bd-bea25

References

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The bea-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75. Association for Computational Linguistics.

Nico Daheim, Jakub Macina, Tanmay Sinha, and Mrinmaya Sachan. 2024. Stepwise verification and remediation of student reasoning errors with large language model tutors. *arXiv preprint arXiv:2407.09136*.

Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems*, pages 1–15. Springer.

Myroslava Dzikovska, Rodney Nielsen, Chris Brew, Claudia Leacock, Danilo Giampiccolo, Luisa Bentivogli, Peter Clark, Ido Dagan, and Hoa Trang Dang. 2013. Semeval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 263–274. Association for Computational Linguistics.

Fredrik Gustafsson, Martin Danelljan, and Thomas B. Schön. 2020. Evaluating scalable bayesian deep learning methods for robust computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 169–170.

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*, Vienna, Austria. Association for Computational Linguistics.

Jakub Macina, Nico Daheim, Sankalan Pal 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. *arXiv preprint arXiv:2305.14536*.

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. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1234–1251, Albuquerque, New Mexico. Association for Computational Linguistics.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14. Association for Computational Linguistics.

Yaniv Ovadia, Elad Fertig, Jae Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, and Balaji Lakshminarayanan. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*.

Nils Reimers and Iryna Gurevych. 2020. sentence-transformers/all-mpnet-base-v2. `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`.

SIGEDU. 2025. Bea 2025 shared task: Pedagogical ability assessment of ai-powered tutors. `https://sig-edu.org/sharedtask/2025`.

Anaïs Tack, Ekaterina Kochmar, Zheng Yuan, Serge Bibauw, and Chris Piech. 2023. The bea 2023 shared task on generating ai teacher responses in educational dialogues. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*.

Anaïs Tack and Chris Piech. 2022a. The ai teacher test: Measuring the pedagogical ability of blender and gpt-3 in educational dialogues. In *Proceedings of the 15th International Conference on Educational Data Mining (EDM)*.

Anaïs Tack and Chris Piech. 2022b. An evaluation taxonomy for pedagogical ability assessment of llm tutors. *arXiv preprint arXiv:2412.09416*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Johnathan Xie, Annie S. Chen, Yoonho Lee, Eric Mitchell, and Chelsea Finn. 2024. Calibrating language models with adaptive temperature scaling. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yue Zhang, Wei Wang, and Xiaojun Wan. 2023. Boosting language-driven ordering alignment for ordinal classification. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS)*.

Appendix

A Software and Package Details

We conducted all experiments using Python 3.9, PyTorch 1.13, and the Hugging Face Transformers library (version 4.37.2) (Wolf et al., 2020). Specifically, we fine-tuned the `sentence-transformers/all-mpnet-base-v2` model available on the Hugging Face Model Hub (Reimers and Gurevych, 2020). Tokenization was performed using MPNet's tokenizer, with inputs truncated to a maximum length of 300 tokens.

All models were trained on a single NVIDIA RTX 3090 GPU (24 GB). Each fold took approximately 2–4 minutes per epoch to train, with convergence typically reached within 3 epochs (i.e., 6–12 minutes per model). Full ensemble training (10 models for Track 1 and 7 for Track 2) completed in under 3 hours. Despite the ensemble size, inference was efficient: classifying the entire test set (several hundred responses) took under 30 seconds.

B Training Configuration

Class Weights. To mitigate class imbalance, we applied inverse frequency class weighting in the cross-entropy loss function:

$$w_c = \frac{1}{\log(f_c + \epsilon)},$$

where f_c is the frequency of class c and $\epsilon = 1.05$.

Hyperparameter Search. We performed grid search over learning rates {1e-5, 2e-5, 3e-5} and batch sizes {8, 16}. The best configuration was selected based on average macro-F1 over the cross-validation folds.

Reproducibility. We fixed all random seeds to 42 and set PyTorch to deterministic mode. Our code will be made publicly available upon publication.

C Preprocessing Frequency Across Models

Table 4 summarizes the frequency of manual cleanup operations required across models.

D Additional Training Results

Table 5 reports additional macro-F1 scores for Mistake Identification and Mistake Location tasks across various models. For non-Transformer models, we used TF-IDF representations as input features.

Category	**Phi3**	**Mistral**	**Llama-3.1-8B**	**Llama-3.1-405B**	**GPT-4**	**Total**
Extra Info	1	0	1	11	1	14
Appended Dialogue Trimming	19	0	0	0	0	19
Code Abstraction	2	0	0	0	0	2
Punctuation Cleanup	3	2	0	0	0	5
Totals	**25**	**2**	**1**	**11**	**1**	**40**

Table 4: Model-specific frequencies of manual cleanup operations on tutor responses.

Model	**Mistake Identification**	**Mistake Location**
BERT	**0.8703**	**0.7025**
RoBERTa	0.7816	0.6551
DeBERTa	0.8576	**0.7025**
ELECTRA	0.8513	0.6266
MPNet	0.8639	0.6203
NeoBERT	0.8513	0.6677
Logistic Regression	0.7880	0.6139
Random Forest	0.8260	0.6551
Gradient Boosting	0.8418	0.6519
SVM	0.7785	0.6110
LightGBM	0.8418	0.6551
XGBoost	0.8386	0.6646
CatBoost	0.8196	0.6582

Table 5: Macro-F1 scores for Mistake Identification and Mistake Location tasks across Transformer models and TF-IDF + traditional classifiers. Best results per column are bolded.