AI3703: Natural Language Processing

Evaluating AI Tutor Effectiveness: A Classification Approach to Pedagogical Abilities

Donal Loitam (AI21BTECH11009)
Sai Pradeep (AI21BTECH11013)
Suraj Kumar (AI21BTECH11029)

April 27, 2025

Abstract

This report details a project focused on evaluating the effectiveness of AI tutors by analyzing their pedagogical abilities through natural language processing. We utilized a provided JSON dataset containing tutor-student conversations and associated annotations across four pedagogical dimensions: Mistake Identification, Mistake Location, Providing Guidance, and Actionability. Exploratory data analysis revealed class imbalances and informed our choice of modeling strategies. We developed a preprocessing pipeline and experimented with various transformer-based models (DistilBERT, BERT, RoBERTa), task-handling strategies (single-task vs. multi-task), input formulations (with/without history), and loss functions (Cross-Entropy, Weighted Cross-Entropy, Focal Loss). Our results indicate that single-task models generally outperform multi-task approaches, and Focal Loss effectively handles class imbalance. Ablation studies on model layers provide insights into optimal representational depth for different models. The best performing configuration (RoBERTa-base, single-task, Focal Loss, no history) was evaluated on a held-out test set. Finally, model interpretability was explored using LIME to understand key textual features driving classification decisions for each pedagogical dimension. Additionally, we explore a better model ELECTRA with the best performing configurations identified during experimentation.

Contents

1 Introduction

Artificial Intelligence (AI) tutors hold significant promise for personalized education. Evaluating their effectiveness is crucial for development and deployment. Beyond measuring task completion or correctness, understanding the *pedagogical strategies* employed by AI tutors offers deeper insights. This project aims to automatically assess AI tutor effectiveness by classifying tutor responses along four key pedagogical dimensions using natural language processing techniques. We frame this as a text classification problem, exploring various model architectures, training strategies, and preprocessing steps to build robust classifiers for evaluating pedagogical abilities inherent in tutor dialogue.

2 Dataset and Exploratory Data Analysis (EDA)

2.1 Dataset Overview

The primary data source for this project was a provided JSON dataset containing tutor-student conversational data.

- **Features Used:** The core features utilized for modeling were `conversation_history`, `tutor_response` text, and the four annotation labels (`Mistake_Identification`, `Mistake_Location`, `Providing_Guidance`, `Actionability`).
- **Shape:** The dataset contains 2476 instances and 8 columns.
- **Schema:** The dataset columns and types are summarized below (inferred from slide):

```
0 conversation_id          2476 non-null object
1 conversation_history     2476 non-null object
2 tutor                    2476 non-null object
3 response                 2476 non-null object
4 Mistake_Identification   2476 non-null int64
5 Mistake_Location         2476 non-null int64
6 Providing_Guidance       2476 non-null int64
7 Actionability            2476 non-null int64
```

2.2 Exploratory Data Analysis

2.2.1 Label Distribution

The distribution of labels for each of the four tasks was examined. As shown in Figure 1, there is a noticeable class imbalance, particularly with the "To some extent" and "No" categories often being minority classes compared to "Yes".

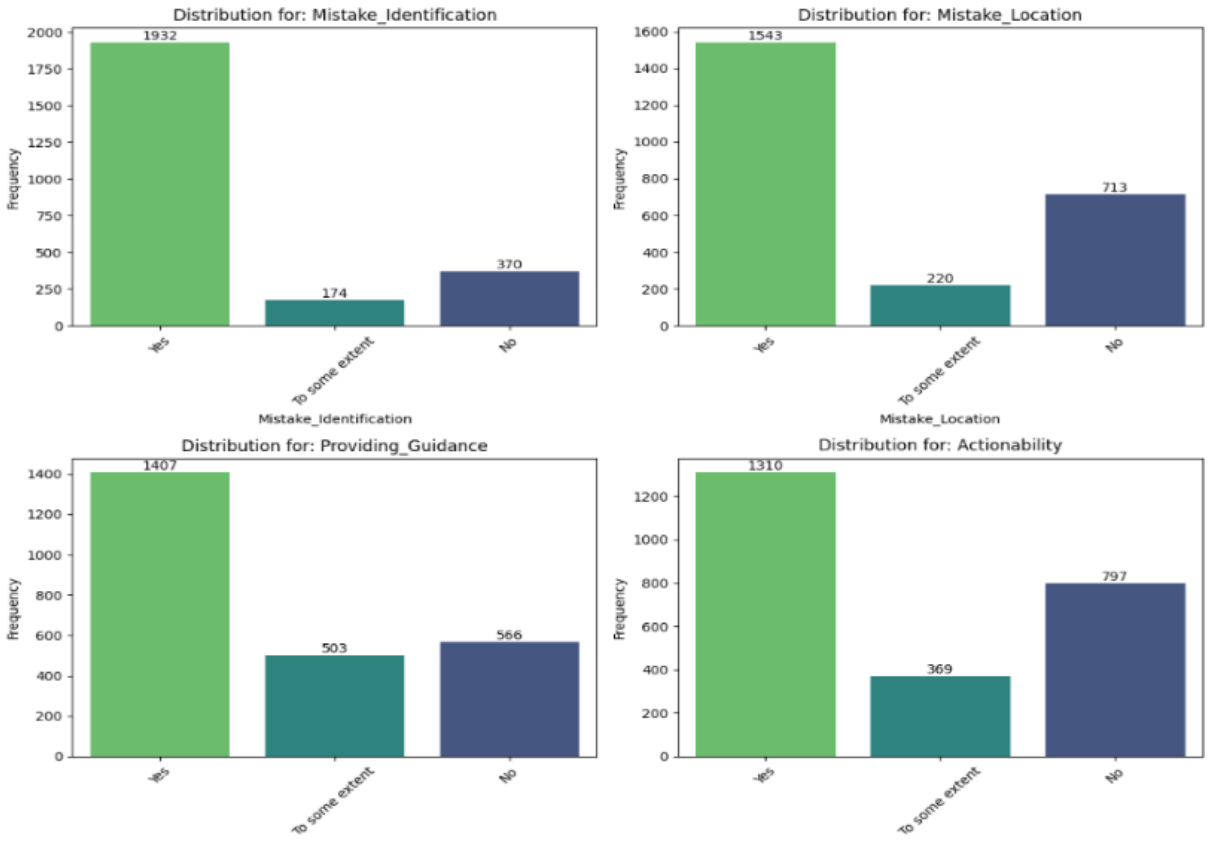

Figure 1: Distribution of labels for the four pedagogical tasks.

2.2.2 Text Length Distribution

We analyzed the length of both the tutor responses and the conversation history, measured in characters and words (Figures 2 and 3). Response lengths are typically short, while history lengths show a wider, more varied distribution.

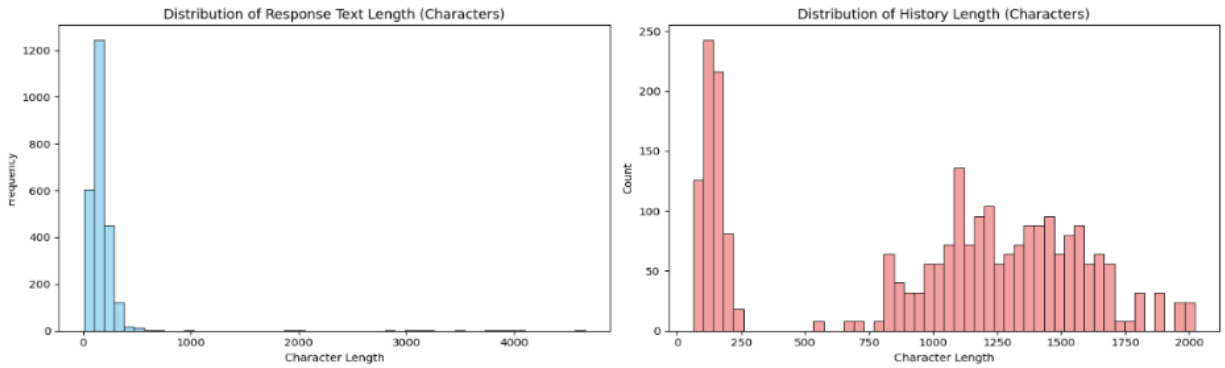

Figure 2: Distribution of Response and History Length (Characters).

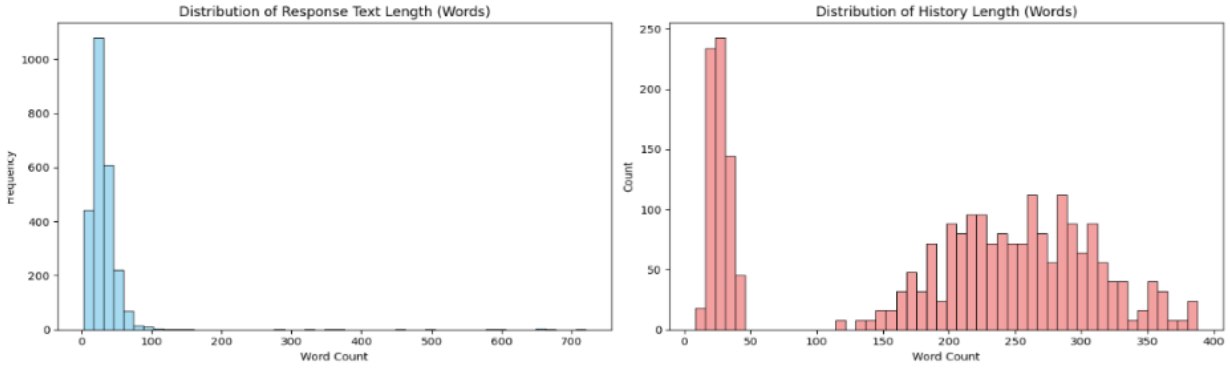

Figure 3: Distribution of Response and History Length (Words).

Descriptive statistics (Table 1) further quantify these lengths. Notably, 90% of responses are under 49 words, and 90% of histories are under 313 words. This analysis informed the choice of maximum sequence length (`MAX_LEN`) for model inputs, balancing the need to capture sufficient context against computational cost and truncation effects. Values of 256 and 384 were considered.

Table 1: Descriptive Statistics for Text Lengths (Words).

Statistic	response_word_count	history_word_count
count	2476.000000	2476.000000
mean	32.339257	192.002019
std	44.575923	113.093320
min	2.000000	8.000000
50%	27.000000	221.000000
90%	49.000000	312.500000
95%	59.000000	348.000000
99%	99.000000	374.000000
max	721.000000	388.000000

2.3 EDA Takeaways

The EDA highlighted several key points influencing the modeling approach:

- **Class Imbalance:** The observed imbalance necessitates strategies to prevent models from being biased towards the majority class.
- **Loss & Metrics:** Weighted loss functions (like Focal Loss or class-weighted Cross-Entropy Loss) are suitable for addressing imbalance. Macro-averaged F1 score is appropriate for balanced evaluation across classes.
- **Stratified Splitting:** Using stratified train-validation-test splits is crucial to maintain representative label distributions in each set.

3 Methodology

3.1 Preprocessing Pipeline

A configurable preprocessing pipeline was developed:

1. **Input Formulation (Configurable):** Two main formulations were tested:
   - Using only the tutor `response` text.
   - Concatenating `conversation_history` + `[SEP]` token + `response`.
2. **Text Cleaning (Configurable):**
   - **Tokenizer:** Hugging Face's `AutoTokenizer` was used, corresponding to the chosen base model (`distilbert-base-uncased`, `bert-base-uncased`, `roberta-base`).
   - **Lowercasing:** Applied consistently.
   - **Other options (Not Used):** Punctuation and stopword removal were included as configurable options but ultimately not used in the final reported experiments based on initial validation.
3. **Label Handling:** Text labels ("Yes", "To some extent", "No") were mapped to numerical representations (e.g., 0, 1, 2).
4. **Data Splitting:** The dataset was split into training, validation, and test sets using an 80-10-10 ratio with stratification based on the labels for each task.

3.2 Model Architecture and Task Strategy

- **Base Models:** Pre-trained Transformer models from the Hugging Face library formed the core architecture:
  - `distilbert-base-uncased`
  - `bert-base-uncased`
  - `roberta-base`

  Standard pre-trained language model weights were used without further pre-tuning on domain-specific data.
- **Task Handling Strategies:** Two primary strategies were evaluated:
  - **Single-Task:** Four independent models were trained, one dedicated to each pedagogical task.
  - **Multi-Task:** A single shared base model was used, with four separate classification heads (one per task) attached on top.

3.3 Training and Fine-Tuning

- **Optimizer:** AdamW optimizer was used with a learning rate of `2e-5`.
- **Loss Functions:** Three loss functions were experimented with:
  - Standard Cross-Entropy Loss.
  - Cross-Entropy Loss with class weights (inversely proportional to class frequency).
  - Custom Focal Loss (with gamma = 2.0) to address class imbalance by down-weighting easy examples.
- **Training Duration:** Models were trained for a maximum of 5 epochs.
- **Early Stopping:** Implemented with a patience of 3 epochs, monitoring the average validation lenient F1 score across the four tasks. Training stopped if the score did not improve for 3 consecutive epochs.
- **Hardware:** Training was conducted on Google Colab using T4 GPUs.

4 Experiments and Results

4.1 Experimental Setup

A total of 36 distinct experiments were conducted by varying the following parameters:

- **Base Model (3):** `distilbert-base-uncased`, `bert-base-uncased`, `roberta-base`
- **Task Strategy (2):** `single_task`, `multi_task`
- **Preprocessing (2):** Including conversation history (`True`/`False`)
- **Loss Function (3):** `CrossEntropyLoss`, `WeightedCrossEntropyLoss`, `FocalLoss`

(Calculation: $3 \times 2 \times 2 \times 3 = 36$)

4.2 Validation Results Analysis

The performance across these 36 experiments was evaluated on the validation set. Key trends observed include:

- **Best Metrics:** Tables 2 and 3 show the configurations achieving the highest score for each metric on the validation set for tasks 1-2 and 3-4, respectively. Different configurations excel depending on the specific task and metric.
- **Strategy Comparison (Figure 4):** Single-task models generally achieved slightly higher median and overall average lenient F1 and Accuracy scores compared to multi-task models on the validation set.
- **Model Comparison (Figure 4):** RoBERTa and BERT tended to perform slightly better and more consistently than DistilBERT, although differences were often small.
- **History Inclusion (Figure 5):** Models trained without conversation history generally performed better than those that included it, based on average lenient scores.
- **Loss Function Comparison (Figure 5):** Focal Loss and standard Cross-Entropy Loss generally outperformed Weighted Cross-Entropy loss, with Focal Loss often achieving the highest median performance, likely due to its effectiveness in handling class imbalance.

Table 2: Best Validation Metrics for Tasks 1 (Mistake Identification) and 2 (Mistake Location).

Task	Metric Type	Max Value	Model	Strategy	Loss	Include History
Mistake_Identification	exact_acc	0.899	distilbert-base-uncased	single_task	FocalLoss	False
Mistake_Identification	exact_f1	0.749	roberta-base	single_task	FocalLoss	False
Mistake_Identification	lenient_acc	0.960	bert-base-uncased	single_task	CrossEntropyLoss	True
Mistake_Identification	lenient_f1	0.977	distilbert-base-uncased	single_task	FocalLoss	False
Mistake_Location	exact_acc	0.774	distilbert-base-uncased	single_task	CrossEntropyLoss	False
Mistake_Location	exact_f1	0.609	distilbert-base-uncased	single_task	CrossEntropyLoss	False
Mistake_Location	lenient_acc	0.847	distilbert-base-uncased	single_task	CrossEntropyLoss	False
Mistake_Location	lenient_f1	0.892	distilbert-base-uncased	single_task	CrossEntropyLoss	False

Table 3: Best Validation Metrics for Tasks 3 (Providing Guidance) and 4 (Actionability).

Task	Metric Type	Max Value	Model	Strategy	Loss	Include History
Providing_Guidance	exact_acc	0.701	roberta-base	multi_task	CrossEntropyLoss	False
Providing_Guidance	exact_f1	0.596	roberta-base	single_task	CrossEntropyLoss	False
Providing_Guidance	lenient_acc	0.847	bert-base-uncased	single_task	CrossEntropyLoss	False
Providing_Guidance	lenient_f1	0.907	bert-base-uncased	single_task	CrossEntropyLoss/FocalLoss	False
Actionability	exact_acc	0.741	roberta-base	single_task	FocalLoss	False
Actionability	exact_f1	0.617	distilbert-base-uncased	single_task	WeightedCrossEntropyLoss	False
Actionability	lenient_acc	0.879	distilbert-base-uncased	multi_task	FocalLoss	False
Actionability	lenient_f1	0.913	distilbert-base-uncased	multi_task	CrossEntropyLoss/FocalLoss	False

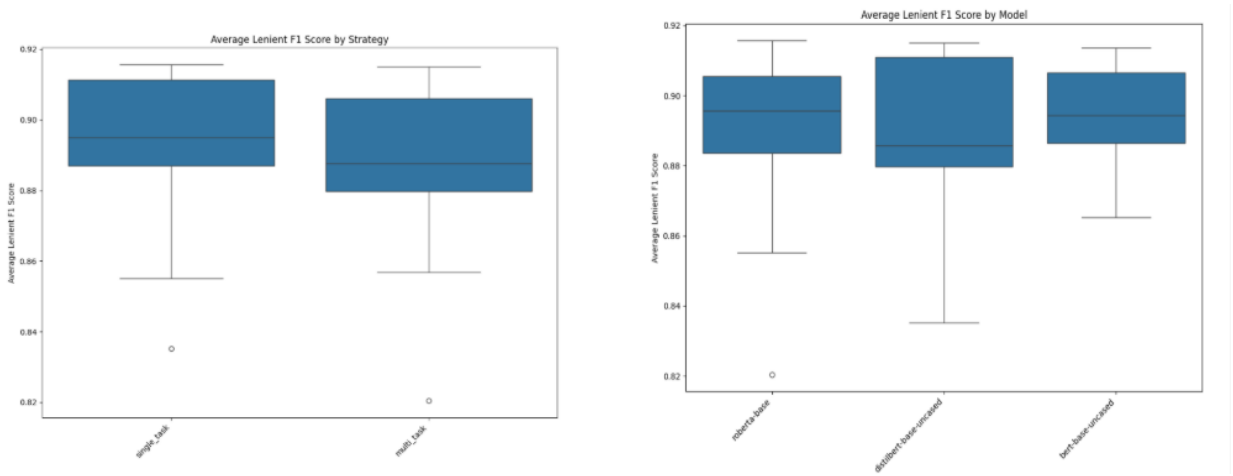

Figure 4: Average Lenient F1 Score by Strategy (Left) and Model (Right) on Validation Set.

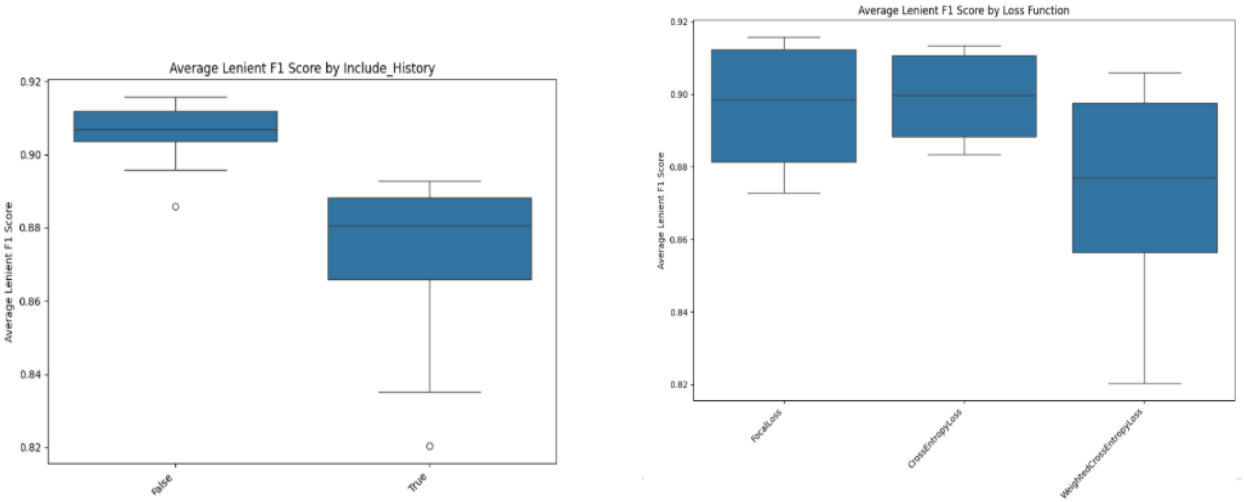

Figure 5: Average Lenient F1 Score by History Inclusion (Left) and Loss Function (Right) on Validation Set.

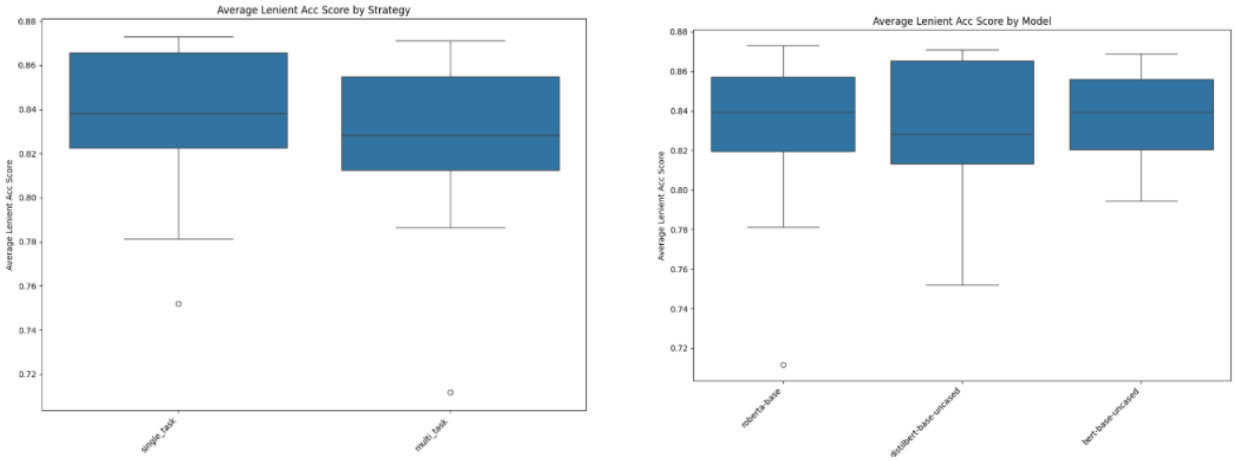

Figure 6: Average Lenient Accuracy Score by Strategy (Left) and Model (Right) on Validation Set.

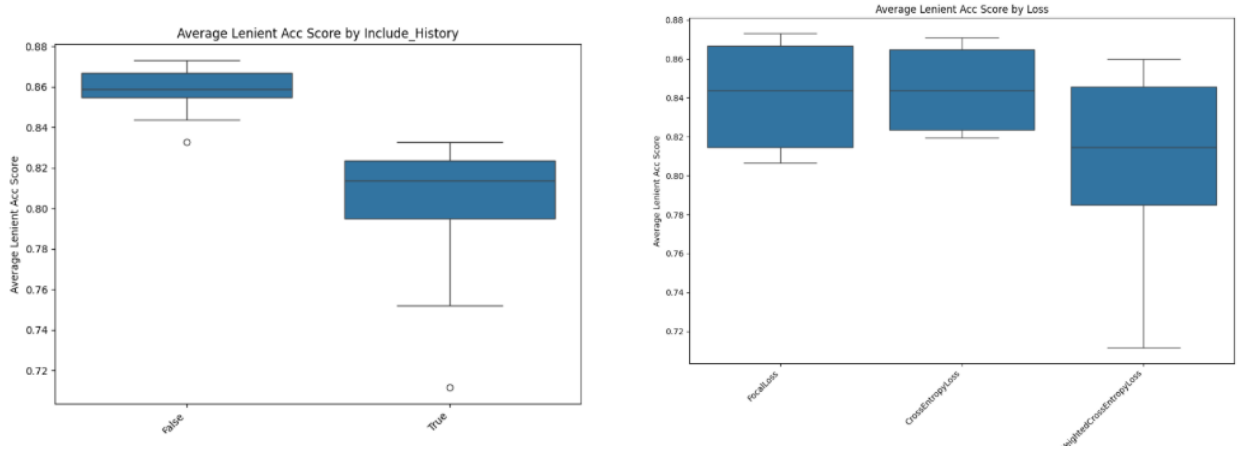

Figure 7: Average Lenient Accuracy Score by History Inclusion (Left) and Loss Function (Right) on Validation Set.

4.3 Final Test Set Evaluation

Based on the validation results, the best performing configuration was identified as:

- **Model:** `roberta-base`
- **Strategy:** `single_task`
- **Loss Function:** `FocalLoss`
- **Input:** No conversation history (`include_history = False`)

This configuration was evaluated on the held-out test set (size: 248 instances, derived from a stratified split of the 2476 total instances, with 2228 used for train+validation). The performance on the test set is shown in Table **??**.

Table 4: Final Test Set Performance for the Best Model Configuration.

Task	Exact Acc	Exact F1	Lenient Acc	Lenient F1
Mistake_Identification	0.8750	0.6727	0.9395	0.9649
Mistake_Location	0.6815	0.4783	0.7863	0.8616
Providing_Guidance	0.6250	0.5924	0.7944	0.8661
Actionability	0.7298	0.6637	0.8629	0.9000
Average(over tasks)	**0.7278**	**0.6018**	**0.8458**	**0.8982**

4.4 Ablation Studies: Impact of Layer Choice

To understand the contribution of different layers within the transformer models, ablation studies were conducted using representations from specific layers or combinations. The metrics are averaged across all 4 tasks for simplicity.

- **Setup:** We compared performance using representations from:
  - Layer 10 (Layer 4 for DistilBERT)
  - Layer 11 (Layer 5 for DistilBERT)
  - Concatenation of Layers 11 and 12 (Layers 5 and 6 for DistilBERT)
  - Final Layer (Layer 12 for RoBERTa/BERT, Layer 6 for DistilBERT)

  Average scores across all four tasks were reported for both Exact and Lenient metrics.

- **DistilBERT Results (Table 5):** Combining layers 5 and 6 yielded the highest average Lenient F1 (0.9181) and Accuracy (0.8770). Layer 5 alone provided the best Exact F1 (0.5964) and Accuracy (0.7671).

- **BERT Results (Table 6):** The final layer (Layer 12) achieved the best performance across all average metrics (Lenient F1 0.9137, Lenient Acc 0.8690, Exact F1 0.5761). Combining layers 11-12 offered a slight lenient accuracy boost but underperformed Layer 12 alone on exact metrics.

- **RoBERTa Results (Table 7):** Layer 11 provided the highest average Lenient F1 (0.9196) and Accuracy (0.8760). Combining layers 11-12 yielded the best Exact F1 (0.6064), while the final Layer 12 achieved the best Exact Accuracy (0.7691, though Layer 11 was close at 0.7671).

Table 5: Ablation Study Results: distilbert-base-uncased

Layer	Avg Lenient F1	Avg Lenient Acc	Avg Exact F1	Avg Exact Acc
Layer 4	0.9096	0.8639	0.5790	0.7541
Layer 5	0.9149	0.8730	0.5964	0.7671
Layer 5-6	0.9181	0.8770	0.5905	0.7631
Layer 6	0.9118	0.8659	0.5803	0.7621

Table 6: Ablation Study Results: bert-base-uncased

Layer	Avg Lenient F1	Avg Lenient Acc	Avg Exact F1	Avg Exact Acc
Layer 10	0.9049	0.8558	0.5578	0.7419
Layer 11	0.9071	0.8599	0.5641	0.7510
Layer 11-12	0.9089	0.8629	0.5650	0.7450
Layer 12	0.9137	0.8690	0.5761	0.7520

Table 7: Ablation Study Results: roberta-base

Layer	Avg Lenient F1	Avg Lenient Acc	Avg Exact F1	Avg Exact Acc
Layer 10	0.9166	0.8730	0.5971	0.7530
Layer 11	0.9196	0.8760	0.5656	0.7671
Layer 11-12	0.9185	0.8770	0.6064	0.7480
Layer 12	0.9157	0.8730	0.5938	0.7691

5 Analysis and Discussion

5.1 Overall Performance Trends

The experimentation revealed several key insights:

- **Model Choice:** `roberta-base` and `bert-base-uncased` consistently performed slightly better than `distilbert-base-uncased`, though the differences were generally minor.
- **Task Strategy:** Single-task models outperformed the multi-task approach. This suggests potential negative interference between tasks during multi-task learning, hindering overall performance for these specific pedagogical dimensions. Optimizing separate models may be more effective.
- **Input Formulation:** Counter-intuitively, models performed better without including the conversation history. This could be because the history introduces noise or dilutes the model's focus from the critical information present in the tutor's immediate response.
- **Loss Function:** Focal Loss demonstrated superior performance compared to standard Cross-Entropy and significantly better than Weighted Cross-Entropy. This highlights its effectiveness in addressing the observed class imbalance by focusing learning on harder-to-classify examples (like the "To some extent" category).

5.2 Interpretability with LIME

To gain insights into model predictions and increase transparency, we employed LIME (Local Interpretable Model-agnostic Explanations). LIME explains individual predictions by approximating the complex model's behavior locally with a simpler, interpretable model (like linear regression) and highlighting the input features (words) that most influenced the specific outcome.

Figures 8 through 11 show LIME explanations for an example prediction for each of the four tasks, using the best performing model.

- **Mistake Identification (Figure 8):** Words like "confusion", "subtract", and "Try" positively contribute to identifying a mistake. The highlighting also shows the model focusing on the tutor referencing incorrect values (245, 60) and suggesting a correction.
- **Mistake Location (Figure 9):** Similar terms contribute, but LIME highlights the model's focus on pinpointing the specific misused value ("not from 60").
- **Providing Guidance (Figure 10):** Key contributing words include modal verbs ("should", "could") and instructional terms ("subtract", "total time", "300 minutes"), emphasizing the corrective instruction provided.
- **Actionability (Figure 11):** Tokens indicating explicit next steps ("Try", "what", "subtract", "should") are highlighted, showing the model recognizes the suggestion for the student to rework the problem.

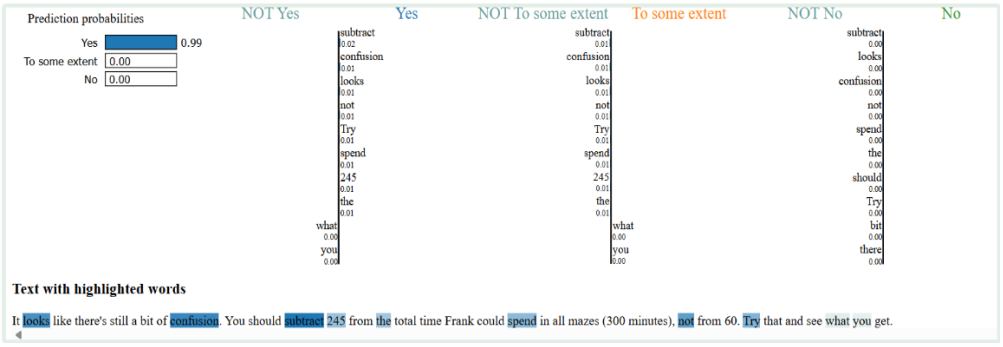

Figure 8: LIME Explanation for Mistake Identification.

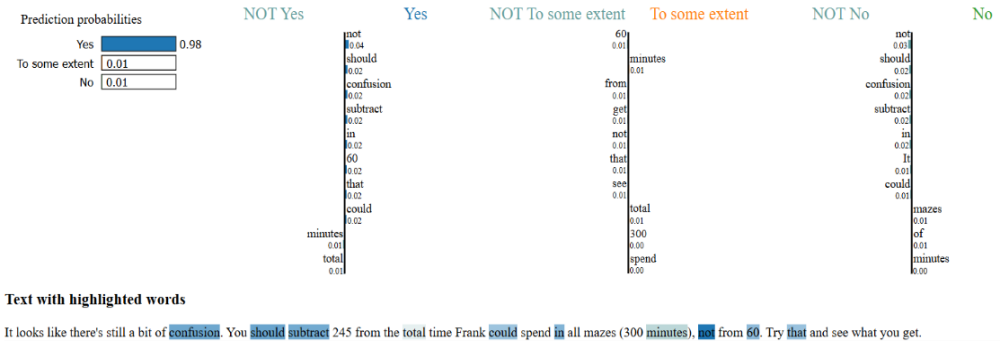

Figure 9: LIME Explanation for Mistake Location.

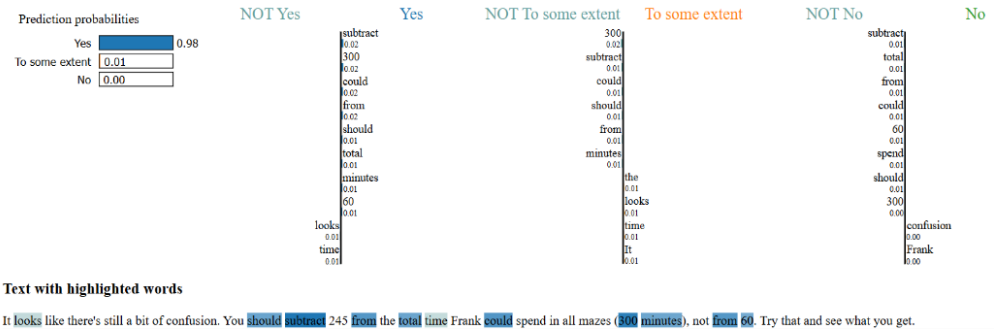

Figure 10: LIME Explanation for Providing Guidance.

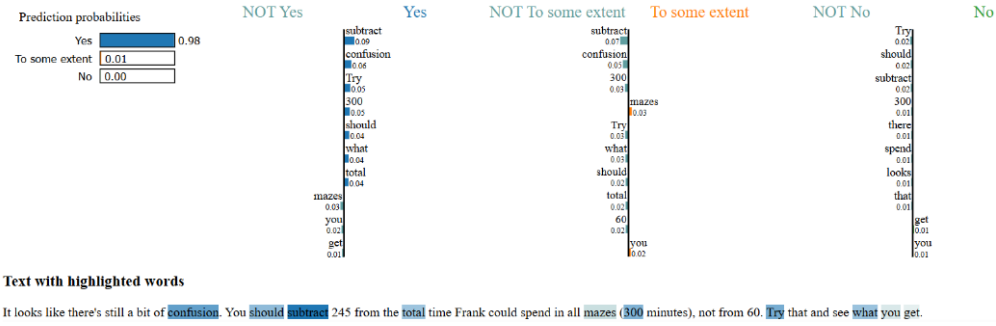

Figure 11: LIME Explanation for Actionability.

6 Experimentation with ELECTRA Model

6.1 Motivation

Following the initial experiments with models such as DistilBERT, BERT, and RoBERTa, we sought to explore alternative transformer architectures known for their efficiency and performance on various NLP tasks. ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) was selected as a promising candidate. Its pre-training objective, Replaced Token Detection (RTD), differs significantly from Masked Language Modeling (MLM) used by BERT/RoBERTa and is often cited for its sample efficiency. The goal was to evaluate if ELECTRA could achieve comparable or superior performance on our pedagogical assessment tasks compared to the previously tested models, particularly RoBERTa which had shown strong results.

6.2 Experimental Setup

For this phase of the experiment, we focused on evaluating the ELECTRA model using the configuration identified as potentially optimal or representative from prior runs, while substituting the base model.

6.2.1 Configuration

The specific configuration used for the final ELECTRA evaluation was:

- **Base Model**: `google/electra-base-discriminator`

- **Strategy**: Single-Task (Separate models trained for each of the four tasks)
- **Loss Function**: Focal Loss (Gamma = 2.0, default alpha)
- **Preprocessing**:
  - Include Conversation History: False
  - Remove Punctuation: False
  - Remove Stopwords: False
- **Epochs for Final Training**: 5
- **Optimizer**: AdamW
- **Learning Rate**: 2×10^{-5}
- **Max Sequence Length**: 384
- **Batch Size**: 16 (Reduced from 32 due to GPU memory constraints observed during training)

6.2.2 Dataset Split

The dataset was split into training and test sets using the same stratified split and random seed (SEED=42) as in previous experiments to ensure comparability.

6.3 Results

The ELECTRA model, trained using the single-task strategy and Focal Loss for 5 epochs on the combined training/validation set, yielded the following performance on the final test set:

6.3.1 Average Metrics Across Tasks

- **Average Test Lenient F1 Score**: 0.8863
- **Average Test Exact F1 Score**: 0.4773

6.3.2 Per-Task Metrics

Table 8: Test Set Performance Metrics for ELECTRA (Single-Task, Focal Loss)

Task	Exact Acc	Exact F1	Lenient Acc	Lenient F1
Mistake_Identification	0.6976	0.6006	0.9355	0.9619
Mistake_Location	0.6653	0.4797	0.7540	0.8310
Providing_Guidance	0.5323	0.2316	0.7782	0.8753
Actionability	0.6331	0.5974	0.8266	0.8768

6.4 Findings and Discussion

An interesting observation arose when comparing the ELECTRA results to those obtained with RoBERTa under a similar configuration (single-task, Focal Loss, include_history=False).

- **Training Loss vs. Test Performance**: During the final retraining phase, the ELECTRA models consistently achieved lower average training losses per epoch compared to the RoBERTa models.
- **Generalization Gap**: Despite lower training loss, ELECTRA's performance on the unseen test set was notably lower than RoBERTa's, particularly in terms of the Average Exact F1 score (ELECTRA: 0.4773 vs. RoBERTa: 0.6018) and also slightly lower in Average Lenient F1 (ELECTRA: 0.8863 vs. RoBERTa: 0.8982).

This discrepancy suggests that the ELECTRA model, under this specific configuration and training regime, might have been overfitting to the training data more than RoBERTa. Potential contributing factors include:

- **Overfitting**: ELECTRA might have memorized training set nuances rather than learning generalizable pedagogical patterns.
- **Batch Size Impact**: The necessary reduction in batch size for ELECTRA (due to memory) could have introduced more noise during training.
- **Model Architecture/Pre-training Bias**: The inherent differences in pre-training objectives (RTD vs. MLM) might make RoBERTa's learned representations slightly more suitable for the nuances of these pedagogical assessment tasks.

The key takeaway is that training loss is not the ultimate measure of model quality; test set performance is. ELECTRA demonstrated it could fit the training data very well, but RoBERTa proved more effective at generalizing its learning to new, unseen examples in this specific setup.

7 Conclusion

This project successfully implemented and evaluated a pipeline for classifying AI tutor responses based on four pedagogical dimensions. Through systematic experimentation with different models, training strategies, input formats, and loss functions, we identified effective configurations for this task. Key findings include the superiority of single-task models over multi-task learning for these distinct dimensions, the counter-intuitive benefit of excluding conversation history, and the effectiveness of Focal Loss in handling class imbalance. Layer-wise ablation studies provided insights into the representational power of different layers within transformer models for these tasks. Furthermore, model interpretability was explored using LIME, demonstrating its utility in understanding which textual features drive specific pedagogical classifications. The best performing model (`roberta-base`, single-task, Focal Loss) provides a strong baseline for future work in automated pedagogical assessment of AI tutors.

8 Future Work

Several avenues exist for future exploration:

- **Hyperparameter Tuning:** Further tuning, such as exploring different learning rates or increasing the `MAX_LENGTH` based on text length analysis, could yield improvements.
- **Error Analysis:** A deeper dive into misclassifications could reveal systematic errors and guide model refinement.
- **Advanced Features:** Incorporating external knowledge bases or more sophisticated linguistic features might enhance performance.
- **Advanced Models/Ensembles (explored *)**: Exploring larger or more recent transformer architectures or employing ensemble techniques could boost accuracy.
- **History Encoding:** Investigating alternative methods for incorporating conversation history, such as prioritizing recent turns or using hierarchical encoding, might prove more effective than simple concatenation.