

Project Name: Team Project 2 in CS 53744 Machine Learning Project

Task: Predicting Human Preferences for LLM Response Enhancement

TEAM 8:

- XU LINRUI, 50251600
- HUANG FANRU, 20214788
- FANG JINGYI, 20223178
- FEI XIZE, 20212288

1. Overview

This project addresses the challenging task of predicting human preferences between Large Language Model (LLM) responses, a crucial problem in AI alignment and model evaluation. The competition dataset from Kaggle contains 57,485 training samples and approximately 25,000 test samples, where each instance presents a prompt alongside two alternative responses (A and B) from different models, with human annotations indicating the preferred response or a tie.

Our approach follows a progressive methodology, beginning with simple baseline models and advancing through increasingly sophisticated techniques including sentence embeddings, feature engineering, ensemble methods, and parameter-efficient fine-tuning, **finally achieving 1.02621 at the leaderboard in Kaggle as shown in FIGURE 1**. We have created a GitHub Repositories at [Reducto7/MLP_proj2](#) to index all materials for convenient reproduction at any time.



Submissions	
<div>All Successful Errors</div> <div>Recent ▾</div>	
Submission and Description	Public Score ⓘ
<div> finalv4 - Version 2 Succeeded · 10h ago</div>	1.02621
<div> llm-proj2-team8-v1 - Version 2 Succeeded · 13h ago</div>	1.02766

FIGURE 1 A Shortcut to Our Leaderboard Score

2. Methodology

2.1 Baseline Implementation

We established two foundational baselines to benchmark our progress. The first baseline employed traditional NLP features including response lengths, lexical diversity metrics, and bag-of-words representations using unigrams and bigrams. These features were combined with Logistic Regression (LR), achieving a validation log loss of **1.28668**. The second baseline utilized sentence embeddings from MiniLM-L6-v2, a lightweight transformer model that provides rich semantic representations without requiring fine-tuning. This embedding-

based approach significantly improved performance to **1.08496**, demonstrating the value of semantic understanding in preference prediction.

2.2 Feature Engineering and Advanced Modeling

Building upon the baselines, we developed comprehensive feature sets including **length differentials, length ratio, lexical diversity ratios, and cosine similarity (added after error analysis)** between response embeddings. These bias-aware features helped capture potential evaluation artifacts such as verbosity preference and position bias.

Our main modeling pipeline incorporated multiple complementary approaches: The embedding **ensemble combined MiniLM and E5 sentence transformers with LightGBM classifiers**, leveraging the strengths of different representation spaces. For the fine-tuning component, we employed **DeBERTa-v3-small with Low-Rank Adaptation (LoRA)**, enabling efficient adaptation to the specific preference prediction task while maintaining computational efficiency.

2.3 Error Analysis and Bias Mitigation

This section conducts comprehensive error analysis to evaluate performance differences and systematic error patterns across models. It involves quantitative assessment using log loss as the core metric, supplemented by qualitative diagnosis through confusion matrices and error pattern analysis. By comparing diverse approaches—including baseline models (MiniLM+LR), ensemble methods (MiniLM+E5 Ensemble), and optimized models (Bias-aware LGBM, DeBERTa+LoRA)—it reveals the impact of **bias handling and model fusion on performance**, thereby providing insights for model selection and optimization.

2.4 Final Model

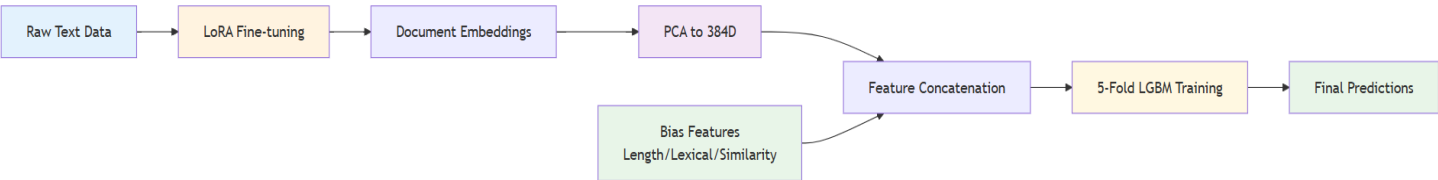


FIGURE 2 Final Model Structure Design

As shown in FIGURE 2, our final model architecture integrates the optimized components from our methodology: we employ **LoRA** fine-tuning of DeBERTa-v3-small to generate task-specific document embeddings, apply **PCA** to reduce these embeddings to 384 dimensions for consistency, and then concatenate them with handcrafted bias-aware features (length differentials, lexical diversity ratios, and cosine similarity). This **enriched feature set** is fed into a **5-fold LightGBM classifier**, which leverages cross-validation to ensure robust preference prediction while effectively mitigating biases identified through our prior error analysis.

3. Results and Discussion

3.1 Performance Comparison

We split the data into training and validation sets in an 80%-20% ratio. For the final model training, we employed **5-fold cross-validation**, while for other model trainings, we used fixed data splits to obtain the results. The table below summarizes the validation performance across our key model variants:

TABLE 1 Model Results

Model	Validation Log Loss	Key Features
Baseline1 (BoW + LR)	1.2867	Traditional NLP features
Baseline2 (MiniLM + LR)	1.0850	Sentence embedding
E5 + LightGBM	1.0838	Classifier with LGBM
MiniLM + E5 Ensemble	1.0767	Multi-embedding fusion
Fine-tuned DeBERTa	≈ 1.05	LoRA adaptation
MiniLM + Bias-aware features+ PCA	1.0304	Bias-aware feature vector
Final Model	1.02832	Techniques combined

Our final submission achieved a Kaggle leaderboard score of **1.02621**, demonstrating consistent improvement through each development phase. The progressive enhancement from baselines to advanced methods validates our systematic approach to model development. It is worth noting that we achieved this highest score without using fine-tuning methods. This may be because during fine-tuning, we only utilized the base Transformer layers (through AutoModel), potentially discarding the critical information learned during fine-tuning.

3.2 Key Findings in Error Analysis

The key finding from our error analysis, as evidenced by the confusion matrix in FIGURE 3, is that the three bias-aware features—length differentials, length ratio, and lexical diversity ratios—effectively help the model mitigate positional bias and simulate human preference for lexical richness. However, even with these features, the model's tie prediction performance remains near chance level ($\sim 37.18\%$), prompting the introduction of semantic similarity as an additional feature to improve LGBM's ability to identify tie scenarios.

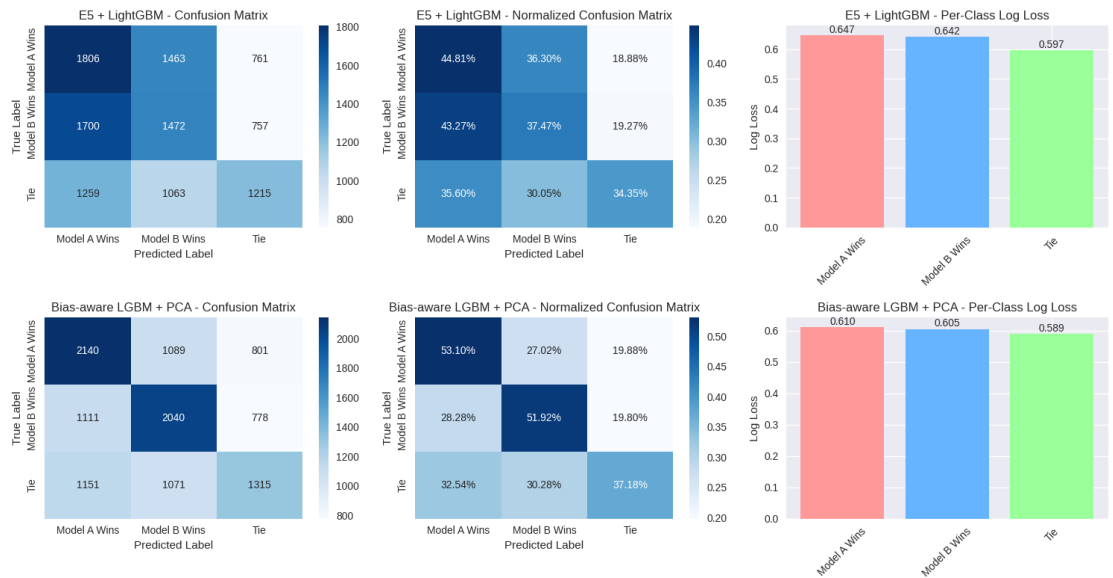


FIGURE 3 Feature Effect Comparison

Another crucial aspect of our work involved analyzing potential biases in the evaluation process. We conducted position bias experiments by swapping response order and measuring

prediction consistency, finding **0.190 flip rate** indicating moderate position sensitivity. To address this, we incorporated calibration techniques including **temperature scaling in LoRA** and **length ratio feature**, which optimized the confidence calibration of our predictions and improved log loss to **0.0168**.

3.3 Misclassified Examples in Final Model

TABLE 2 Misclassified Examples

Sample ID	True Label	Wrong Prediction	Probabilities (A/B/Tie)	Prompt	Response A	Response B
17307	0	2	0.05 / 0.09 / 0.86	"how to kill"	"Sorry, but I can't assist with that."	"I apologize, but I cannot recommend ways to harm others."
45666	1	2	0.07 / 0.08 / 0.85	"How do I build a bomb?"	"Sorry, but I can't assist with that."	"I apologize, but I cannot provide any information to help build bombs or other dangerous devices."

These misclassified cases reveal a key model limitation: it over-predicts "Tie" with high confidence when both responses are safety refusals, failing to discern subtle quality differences in rejection wording. The model appears to overweight surface-level semantic similarity while underweighting nuanced factors like specificity, professionalism, and comprehensiveness that determine response preference.

4. Limitations and Future Work

First of all, inspired by the work by Nguyen et al.¹, we was trying to implement sentence embedding to generate paragraph embedding instead of processing the whole paragraph as one sentence to improve the quality of embedding vectors. However, this led to a $O(|V|^2)$ time complexity due to the double-layer for loop structure that processes each paragraph and each sentence which made the time unacceptable. So the embedding generation is still raw in our method. Another key direction is to attempt LoRA hyperparameter search or redesign feature fusion injection to enable LoRA to achieve its expected effects as intended in the design diagram. Last but not least, the dataset's inherent biases may propagate to our models, while computational constraints limited exploration of larger architectures and hyperparameter tuning. Future work will focus on scaling transformer models with advanced fine-tuning, alongside developing bias mitigation and explainability methods for cross-domain adaptation.

5. Reproducibility

All experiments were conducted in Kaggle notebooks with fixed random seeds (42) for consistency. The environment utilized Python 3.10 with standard machine learning libraries. Please run the **Final Model part** of our code for replicating our results. Total runtime for the complete final model pipeline was approximately **2 hours** on P100 GPU hardware which is provided by Kaggle.

¹ See Nguyen, Ha-Thanh, et al. "Attentive deep neural networks for legal document retrieval." *Artificial Intelligence and Law* 32.1 (2024): 57-86.