Team Project 2

CS 53744 Machine Learning Project

20216793	20225121	20241383	20191635
Junseob Kim	**Soyoung Hyun**	**Hanbin Oh**	**Dongjun Shin**

Github link

https://github.com/thdudgus/LLM-Response-Enhancement

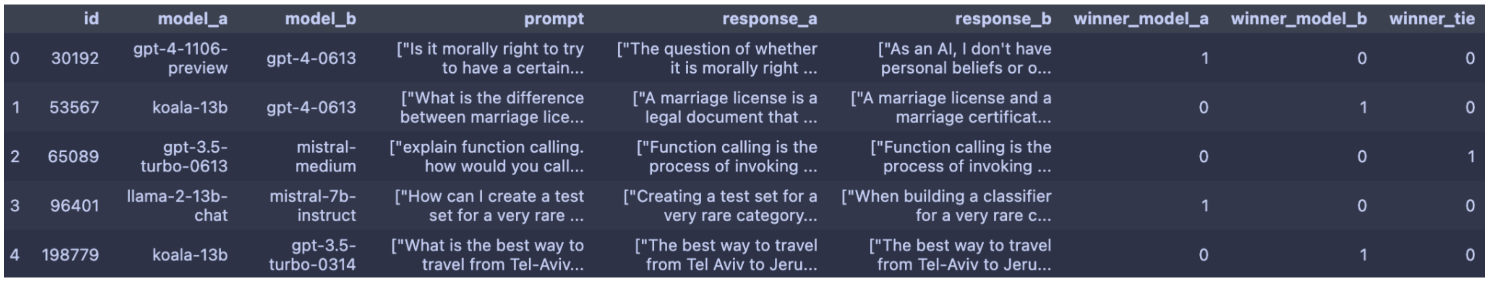
Description of dataset and task

[task]

In Kaggle's **"LLM Classification Finetuning"** competition, the goal is to develop a multi-class classifier that predicts human preferences for LLM-generated texts. Given a prompt, the model must evaluate two different responses (A and B) generated by LLMs and predicts one of three outcomes: **Response A wins**, **Response B wins**, or **a tie**.

[data description]

train.csv: Contains 57,477 training samples. Each sample includes a *prompt*, *response_a*, and *response_b*. The target is represented by three binary columns: *winner_model_a*, *winner_model_b*, and *winner_model_tie*.
test.csv: Contains the test set for which predictions must be made. It does not include the winner columns.
submission.csv: Requires the id and the predicted probabilities for each of the three target classes (A, B, Tie).



	id	model_a	model_b	prompt	response_a	response_b	winner_model_a	winner_model_b	winner_tie
0	30192	gpt-4-1106-preview	gpt-4-0613	["Is it morally right to try to have a certain...	["The question of whether it is morally right ...	["As an AI, I don't have personal beliefs or o...	1	0	0
1	53567	koala-13b	gpt-4-0613	["What is the difference between marriage lice...	["A marriage license is a legal document that ...	["A marriage license and a marriage certificat...	0	1	0
2	65089	gpt-3.5-turbo-0613	mistral-medium	["explain function calling. how would you call...	["Function calling is the process of invoking ...	["Function calling is the process of invoking ...	0	0	1
3	96401	llama-2-13b-chat	mistral-7b-instruct	["How can I create a test set for a very rare ...	["Creating a test set for a very rare category...	["When building a classifier for a very rare c...	1	0	0
4	198779	koala-13b	gpt-3.5-turbo-0314	["What is the best way to travel from Tel-Aviv...	["The best way to travel from Tel Aviv to Jeru...	["The best way to travel from Tel-Aviv to Jeru...	0	1	0

Figure 1. First five rows of train.csv.

Performance Comparison Analysis by Model (TF-IDF, MiniLM, DeBERTa)

The basic idea behind the baseline model and the subsequent models is to use the existing data and outcomes to select, for each new input, the **'most similar'** outcome.

[TF-IDF + Logistic Regression]

As the most basic model, we transformed the text into 150 key word frequency (**TF-IDF**) features, which we entered into a **multi-logical regression model** to classify the winners.

Using Sklearn's `TfidfVectorizer`, we constructed a dictionary of the top 150 most frequent words (max_features=150) among the words appearing in *prompt*, *response_a*, and *response_b* in the training data (df_train). Based on this dictionary, each of the three texts was transformed into a 150-dimensional TF-IDF vector. Using `scipy.sparse.hstack`, these three vectors were concatenated horizontally, resulting in a total of 450 dimensions of feature vectors (X_train_tfidf) per sample as the final input.

We evaluated the performance of our model with 20% verification data (X_val_tfidf, y_val) not used for training. `TfidfVectorizer` fit only with df_train, and LogisticRegression models are also trained only with X_train_tfidf.
Validation Accuracy Score: 0.4568
Validation Log Loss (Competition Metric): 1.0472

This verification score (1.0472) is almost identical to the actual Kaggle submission score of 1.07, indicating that the model is not overfitted. Analysis of classification_report showed that the model predicted the 'Tie' class relatively well (recall high) but had difficulty distinguishing between 'A win' and 'B win' (recall low). This suggests that 150 simple word frequencies alone are limited in capturing the subtle differences in preference of the two answers.

[Embedding based model: MiniLM + Logistic Regression]
We used the pre-trained all-MiniLM-L6-v2 model as a "feature extractor." (In the one above, TF-IDF played that role.) This approach does not train the MiniLM model itself, but only uses it to transform the meaning of the text into a fixed number vector (embedding).
We passed the prompt, response_A, and response_B into the embedding model separately, concatenated the resulting 384-dimensional embeddings, and used the 1,152-dimensional vector as the feature vector.
Using these final feature vectors as inputs for the logistic regression model, we trained them to classify three classes (A/B/Tie).

We loaded the all-MiniLM-L6-v2 model with the SentenceTransformers library to obtain embeddings. We applied L2 normalization to each embedding and then concatenated them to reduce bias toward any single input segment.
As above, we held out 20% as a validation set.
Validation accuracy: 0.4422
Validation log loss: 1.0589 (Submit: 1.065)

There are several limitations to the above approach. No matter how sophisticated our feature extraction is, the final classification still relies on a logistic regression algorithm. This problem isn’t simple enough for a linear model to solve well; therefore, using classifiers fit to more complex problems, we could improve our accuracy of our task.

[Model extensions: End-to-End Finetuning Model Based on DeBERTa V3]
The previous two methods (TF-IDF, MiniLM) were separated from the step of extracting features from text and the step of classifying them by entering them into a linear model, logistic regression. Logistic regression models had clear limitations in learning complex non-linear relationships that distinguish subtle differences in preferences between the two answers.
To overcome these limitations, this approach fine-tuned DeBERTa V3 (deberta_v3_extra_small_en), a pre-trained transformer model using the KerasNLP library and JAX backend, in an end-to-end manner.

The model does not separate the feature extractor from the classifier, and the DeBERTa model itself is trained simultaneously with the classification head. In particular, it adopts a weight-sharing structure, similar to the Siamese Network.

Although we do not embed 'Prompt', 'Response A', and 'Response B' separately like the MiniLM model, we have constructed the following two text pairs for comparison.

input 1: "Prompt: {prompt}\n\nResponse: {response_a}"

input 2: "Prompt: {prompt}\n\nResponse: {response_b}"

These two inputs pass through one and the same DeBERTa V3 Backbone, respectively. Because of the weight-sharing, the model evaluates both inputs on the same basis and generates embedding vectors whose contextual meaning is compressed.

And the two output vectors that have passed through DeBERTa are combined into one in the concatenate layer. This combined vector compresses the contextual information through GlobalAveragePooling1D, resulting in the Dense layer (Softmax activation function) finally outputting probabilities for three classes: 'A win', 'B win', and 'A draw'.

Unlike previous methods, the model trains the entire model, including the weights of the DeBERTa backbone, in a full fine-tuning manner. This allows the model to optimize DeBERTa's ability to understand the context for certain tasks that determine 'what answers are better'.

We used 20% of the training data separated by the validation set and monitored the val_log_loss of the validation set during the training process. We used ModelCheckpoint callback to store the optimal weight with the lowest validation loss and utilize it for final inference.

Beyond the bounds of linear models, this approach has the advantage of being able to learn the complex and subtle differences of two answers nonlinearly.

Performance comparison table (Baseline vs. Final Model)

Model	Validation Loss	Validation Accuracy	Final Score
TF-IDF + Logistic Regression	1.0472	0.4568	1.07049
MiniLM + Logistic Regression	1.0589	0.4422	1.06506
End-to-End Finetuning Model Based on DeBERTa V3	1.0322	0.4723	1.03606

Figure 2. Performance comparison of the baseline models and the final DeBERTa V3-based end-to-end finetuning model in terms of validation loss, validation accuracy, and final competition score.

Error and bias analysis

1. Initial Model (TF-IDF, MiniLM)

TF-IDF + LR:

Kaggle Final Score: 1.07049

Silhouette Score: -0.0040

MiniLM + LR:

Kaggle Final Score: 1.06506

Silhouette Score: -0.0019

The main problem with the early two models was the biased feature space bias. Both models recorded negative Silhouette Scores, meaning that in the vector spaces generated by TF-IDF (450 dimensions) and MiniLM (1152 dimensions), the three classes 'A win', 'B win', and 'winless' are not clustered at all but mixed.

This bias in the feature space caused a catastrophic error in the final classifier, Logistic Regression. The classifier did not find a linear boundary in the mixed data, resulting in a low recall predicting a "no-win" without being able to distinguish the subtle difference between "A win" and "B win." This led to a high Log Loss score of 1.06 to 1.07.

2. Error-correcting with DeBERTa

DeBERTa V3 (End-to-End):

Kaggle Final Score: 1.03606

Silhouette Score: -0.0111

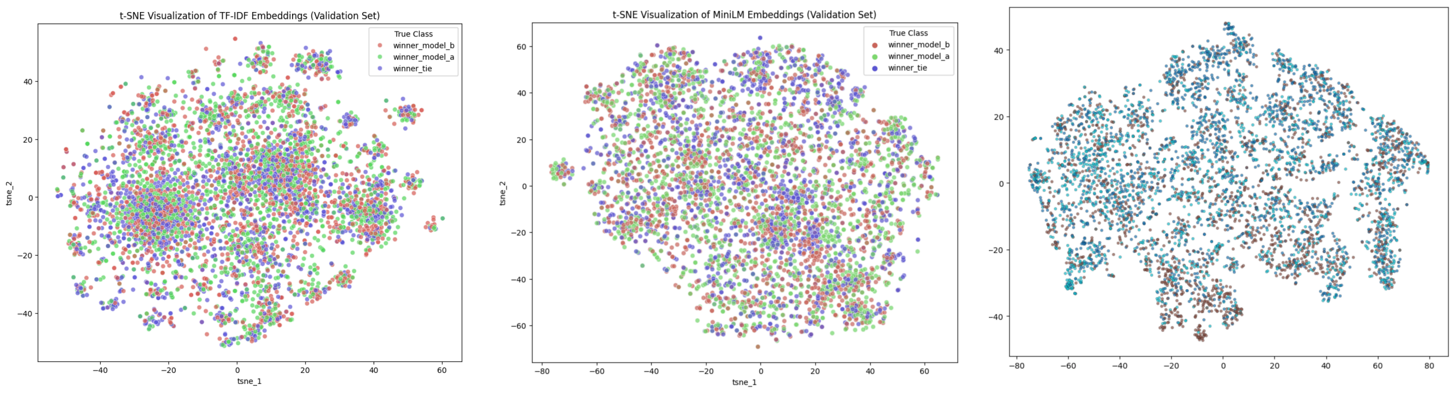
The DeBERTa model tried to solve this fundamental error by reconstructing the vector space.

Although the Silhouette Score remained negative (-0.0111), this seems to be due to the inherent ambiguity of the "Tie" class and the loss of information in the process of projecting high-dimensional data in two dimensions. However, looking at the t-SNE visualization results, we find that the distinction between classes in DeBERTa's embedding space becomes much more visually distinct compared to TF-IDF or MiniLM.

Through end-to-end fine-tuning, DeBERTa learned a new nonlinear relationship optimized for the task without relying on a predefined feature (TF-IDF) or a fixed embedding (MiniLM). Learning by directly comparing the two answers via a Siamese structure, we were able to calibrate the vector space more effectively.

This improvement in vector space has made it possible for the final Dense Layer to differentiate the class more clearly, which directly demonstrates its effectiveness by achieving the lowest Kaggle Log Loss score (1.03606).

[t-SNE Visualization]



TF-IDF + Logistic Regression

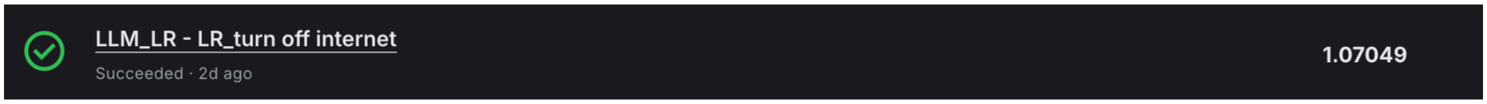
MiniLM + Logistic Regression

End-to-End Finetuning Model Based on DeBERTa V3

Figure 3. t-SNE visualizations of the feature matrices and corresponding labels for each model.

Kaggle leaderboard score

[TF-IDF + Logistic Regression]

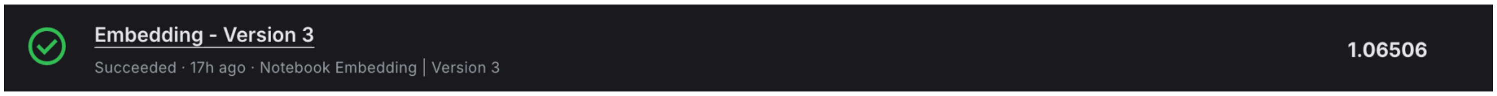


LLM_LR - LR_turn off internet

Succeeded · 2d ago

1.07049

[Embedding based model: MiniLM + Logistic Regression]

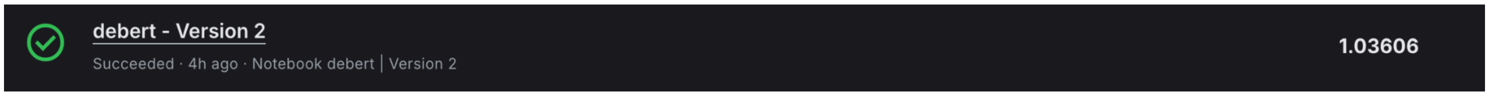


Embedding - Version 3

Succeeded · 17h ago · Notebook Embedding | Version 3

1.06506

[Model extensions: End-to-End Finetuning Model Based on DeBERTa V3]



debert - Version 2

Succeeded · 4h ago · Notebook debert | Version 2

1.03606