Personalized Recommendations Based on MBTI Classification

Kuang, Yenting, kuang.ye@northeasterm.edu
Gao, Siyuan, gao.siyuan1@northeastern.edu
Balswamy, Keerthana, balswamy.k@northeastern.edu

Abstract

Social media posts offer insights into users' personalities, preferences, and behaviors,facilitating people's interest in exploring the Myers-Briggs Type Indicator (MBTI). This enhances recommendation systems by enabling them to deliver more personalized and context-aware suggestions. Therefore, this paper aims to utilize MBTI insights to optimize the efficiency of recommendation systems, reducing the time users spend searching for relevant products and improving their experience. By implementing three models—Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and Bidirectional Encoder Representations from Transformers (BERT), with LSTM and GRU integrating with Word2Vec—the results show that BERT achieves the highest accuracy, reaching 90.96%. Additionally, we found that using the Synthetic Minority Over-sampling Technique (SMOTE) with LSTM and GRU to address imbalanced data improves the F1 score, bringing it to 84%, and prevents the models from overfitting to the majority class.

Key words: *Gated Recurrent Unit; Long Short-Term Memory; Bidirectional Encoder Representations from Transformers; Natural Language Processing; MBTI; Personalized Recommendations*

1. Introduction

Personality classification methods provide essential insights into individual preferences and behaviors. Common approaches include the Big Five Personality Traits, which emphasize quantitative and continuous dimensions for nuanced behavioral analysis, and DISC, which focuses on four behavioral types suitable for workplace communication and leadership. Among these, the Myers-Briggs Type Indicator (MBTI) stands out for its combination of depth and simplicity, categorizing personalities into 16 types based on four dichotomies: introversion vs. extraversion (I/E), sensing vs. intuition (S/N), feeling vs. thinking (F/T), and perceiving vs. judging (P/J). This paper adopts the MBTI method due to its clear structure and practical relevance to personalized analysis.

- Introversion (I) vs. Extraversion (E): ‘I’ is more likely to get energy from the internal environment, while ‘E’ is more likely from the external environment.
- Sensing (S) vs. Intuition (N): ‘S’ is more likely to process information through the five senses, while ‘N’ is more likely through intuition.
- Feeling (F) vs. Thinking (T): ‘F’ is more likely to decide based on emotion, while ‘T’ is more likely to decide based on objective principles.
- Perceiving (P) vs. Judging (J): ’P‘ prefers to do things flexibly, while ‘J‘ prefers to make and execute plans.

The motivation for making this classification model comes from two aspects. The first aspect is that the user experience can be enhanced through personalized product recommendations, as it could save users time in the process of finding relevant products. For example, an extroverted user might receive an interactive or social type of product, and an introverted user might receive a content-centric product. The second aspect is that analyzing the MBTI distribution among high value customer groups can provide better guidance for product optimization and marketing strategies in anticipation of better business results in A/B testing of products.

2. Backgrounds

Ryan et al. (2023) [1] aimed to predict MBTI personality types using multiple machine learning techniques and Synthetic Minority Over-sampling Technique (SMOTE) to address class imbalance. In the phase of pre-processing, the researchers used the Word2Vec model for vectorization and applied SMOTE to balance the data. The machine learning models include logistic regression, linear support vector classification, stochastic gradient descent (SGD), random forest, the extreme gradient boosting classifier, and the cat boosting classifier. With the introduction of SMOTE, their findings have significantly improved the f1 score results, making them better at predicting MBTI.

Mohan et al. (2023) [2] focused on improving MBTI personality predictions by analyzing language patterns from social media posts. They used an LSTM network to capture text sequences, Word2Vec for word embedding, sequence padding for model training, and grid search for hyperparameter tuning. The results demonstrate the effectiveness in capturing complex relationships within textual data, with applications in personalized marketing and psychological analysis.

Ontoum and Chan (2022) [3] also predicted MBTI types from social media text using machine and deep learning. They transformed text using Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) for keyword relevance. In addition, they employed Naive Bayes, Support Vector Machines (SVMs) classifiers, and Recurrent Neural Networks (RNN) for the classification. Furthermore, the authors added Bidirectional Long Short-Term Memory (BiLSTM) layer to improve the performance of the RNN model, which outperformed among the other two classifiers. The study emphasized considering other social networks and soft skills for better personality classification.

The approach proposed in this paper was to perform MBTI personality prediction using word embedding with GRU and LSTM. For imbalance data, we use SMOTE to oversample and randomly undersample the majority class. Additionally, we employed the BERT for natural language processing (NLP) to capture the contextual bidirectional dependencies of text, further improving classification accuracy and generalization.

3. Approach

Several tasks were implemented in this study to develop the model effectively and achieve the research goals. These tasks included a deep dive into understanding the corpus, preparing the data through feature encoding, cleaning, and normalization, and processing the dataset with techniques such as regular expressions, tokenization, and vectorization. The models were then created and trained, and its performance was enhanced using SMOTE. Finally, the model was evaluated by comparing results based on metrics such as F1 score and accuracy.

3.1 Lemmatization

Lemmatization is the process of reducing words to their base or root form (lemma) while ensuring the resulting word is a valid one found in the dictionary. Unlike stemming, which often produces root forms that may not be real words, lemmatization uses linguistic knowledge such as morphology and vocabulary. For instance, "running" becomes "run", and "better" becomes "good". This process reduces data dimensionality by normalizing different inflections of a word, helping models generalize better. In this paper, lemmatization helps improve the accuracy of our models by unifying word variants into a single representative form, reducing noise in textual data.

3.2 Text vectorization

Text vectorization converts textual data into numerical representations so machine learning algorithms can process it. Techniques for vectorization include:

- Bag-of-Words (BoW): Represents text as a collection of word counts, disregarding grammar and word order.
- TF-IDF (Term Frequency-Inverse Document Frequency): Adjusts word frequency by its importance in the corpus, reducing the weight of common words.
- Word Embeddings: These represent words in a continuous vector space, where semantically similar words have similar vectors. For instance, Word2Vec generates dense vectors and captures contextual and semantic meaning.

In this paper, we utilized Word2Vec as the word embeddings model. There are two primary architectures for Word2Vec: Skip-Gram and Continuous Bag of Words (CBOW). Each uses different formulations to learn word representations.

- Skip-Gram Model: Predicts context words w_c given a target word w_t. The objective function is as follow:

$$max \sum_{t=1}^{T} \sum_{c \epsilon C(t)} logP\left(w_c | w_t\right)$$

Where w_t is the target word at position t, $C(t)$ is set of context words within a window around w_t. The probability is as follow:

$$P\left(w_c | w_t\right) = \frac{exp\left(v_c^T v_t\right)}{\sum_{w=1}^{V} exp\left(v_w^T v_t\right)}$$

Where v_t is the vector representation of the target word, v_c is the vector representation of the context word, and V is the vocabulary size.

- Continuous Bag of Words (CBOW) Model: Predicts the target word w_t from context words $C(t)$. The objective function is as follow:

$$max \sum_{t=1}^{T} logP\left(w_c | C(t)\right)$$

The probability is as follow:

$$P\left(w_t | C(t)\right) = \frac{exp\left(v_t^T \overline{v_c}\right)}{\sum_{w=1}^{V} exp\left(v_w^T \overline{v_c}\right)}$$

Where $\overline{v_c}$ is the average of context word vectors within the window.

3.3 Imbalanced Data Processing

SMOTE (Synthetic Minority Over-sampling Technique) is an oversampling technique used to handle imbalanced datasets. It generates synthetic samples for the minority class by interpolating between existing samples. The steps are:

1. Select a minority class sample.
2. Identify its k-nearest neighbors.
3. Randomly select one of these neighbors.
4. Create a synthetic sample along the line segment connecting the original point and the neighbor.

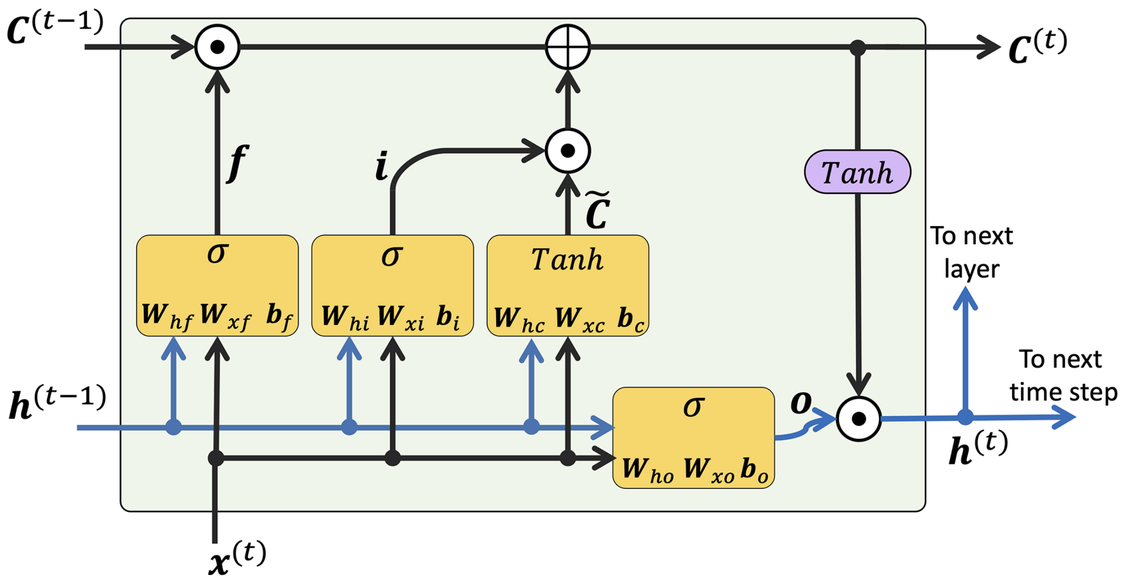
For instance, assume X and $X_{neighbor}$ are feature vectors:

$$X_{synthetic} = X + \lambda \times \left(X_{neighbor} - X\right)$$

where λ is a random number between 0 and 1.

SMOTE helps mitigate class imbalance, improving model performance and stability by ensuring the model does not bias toward the majority class.

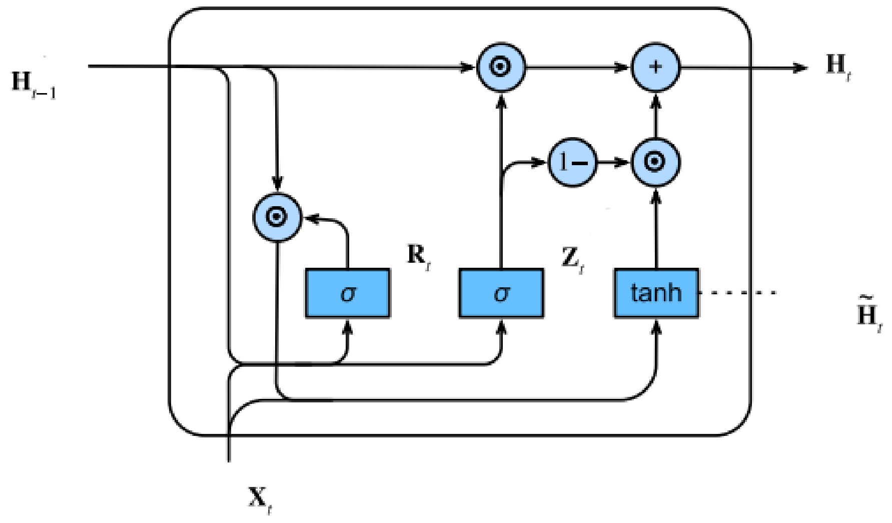
3.4 LSTM



LSTM (Long Short-Term Memory) networks are specialized RNNs designed to capture long-term dependencies in sequential data by using a memory cell and gating mechanisms. The forget gate decides which information to discard from the previous cell state, the input gate determines what new information to add, and the output gate regulates what part of the updated cell state contributes to the output. This structure allows LSTMs to retain relevant information over long time intervals while mitigating vanishing gradient issues. In this study, we use Adam as the optimizer [2], the equation is as follows:

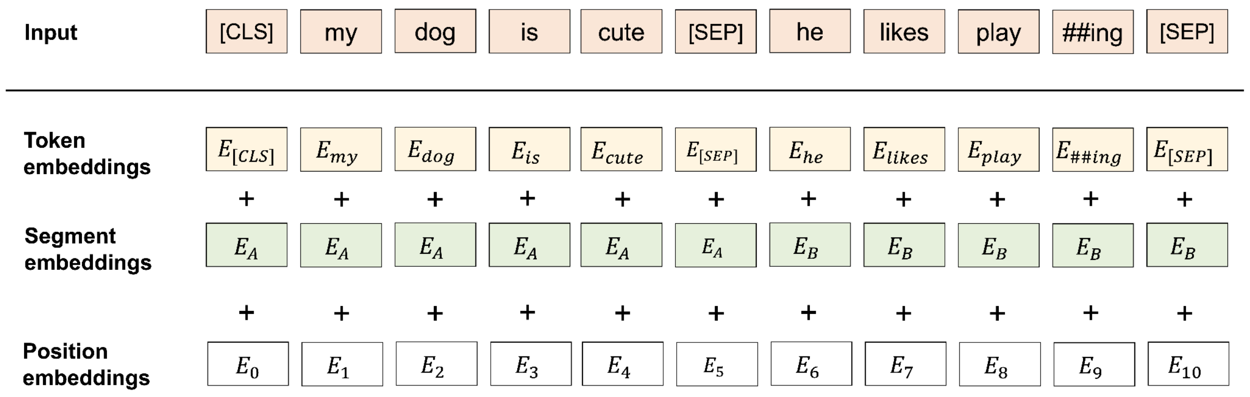
$$w(t+1) = w(t) - learning\ rate \times \frac{\widehat{m}(t+1)}{\sqrt{\widehat{v}(t+1)} + \epsilon} \quad (1)$$

3.5 GRU



GRU (Gated Recurrent Unit) is a type of RNN that simplifies LSTM by combining the forget and input gates into a single update gate and using a reset gate to control the flow of information. The update gate decides how much of the past information needs to be carried forward, while the reset gate determines how much of the previous state should be ignored.

3.6 BERT



BERT (Bidirectional Encoder Representations from Transformers) is a transformer-based model that pre-trains on large text corpora using a masked language modeling objective and next sentence prediction. It encodes input text into token embeddings, enhanced by segment and positional embeddings, enabling a deep understanding of word context in both directions.

4. Results and Discussion

4.1 Corpus Description

The "MBTI 500" dataset used in this study is publicly available on Kaggle [4] and contains over 106K rows of text posts. Each row consists of two columns: the user's MBTI personality type and their social networking posts from Reddit, typically containing 500 words or fewer. Users first complete a questionnaire to determine their MBTI type, which is then associated with their publicly shared posts or forum discussions.

The distribution of MBTI personality types in the dataset is illustrated in Figure 1. It is evident that there is a significant data imbalance across the corpus, which could cause the model to overfit certain categories when making predictions, where we decided to use SMOTE. To mitigate this, we applied SMOTE and treated each of the four MBTI dimensions independently, transforming the original sixteen personality types into four separate binary classification tasks. We then trained four distinct binary classification models, each corresponding to one MBTI dimension.

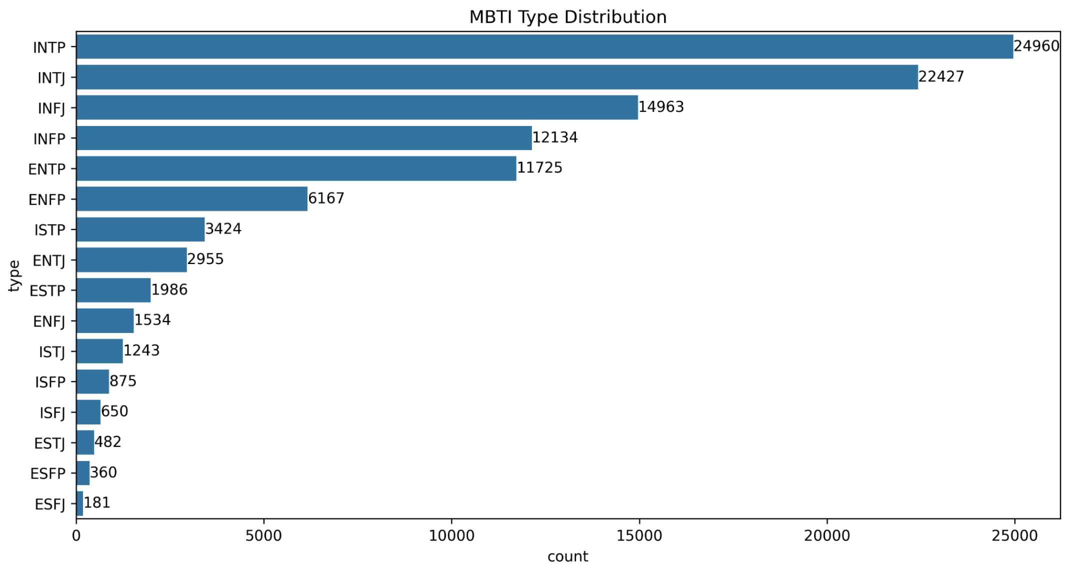


Figure 1. Distribution of MBTI personality types

4.2 Data Pre-processing

The textural data in the corpus needed to be pre-processed to enhance the overall performance in the following model training step. Through these methods, duplicates, null values, and meaningless text can be filtered out, and important information can be retained to maintain the robustness of the model.

- Applied regular expression: By removing URLs, and filtering out non-alphabetic characters such as punctuation, numbers, and special characters with a regular expression, we ensure that the model will not be distracted by irrelevant patterns, leading to simplified text and reduced dimensionality. We also removed the user ID just in case he might have biased or confused the dataset.
- Removed explicitly mention MBTI types: As we processed the texts, we found that some users would explicitly mention personality types in their posts, like "I'm an INFP,". Removing these terms ensures that the model learns from the content, not from direct mentions of MBTI labels, to prevent the model from remembering this information rather than learning patterns in the context.
- Elimination of stop words: Stop words are often used terms in a language such as “the,” “is,” and “and”, that have little or no sense in a sentence. As they can generate noise and increase the computational load without adding much to the prediction task, these words are eliminated from the text data.
- Lowercasing: Converts all text to lowercase for uniformity, preventing the model from treating "INFJ" and "infj" as different words.
- Tokenization and lemmatization: Tokenization splits text into individual words, and lemmatization reduces words to their base forms to minimize variability while preserving semantics. These steps standardize text, ensuring the model focuses on meaningful linguistic patterns crucial for MBTI classification.

Additionally, we used text vectorization to convert unstructured text into numerical vectors, enabling deep learning models to process and analyze the data. This is crucial for capturing semantic and contextual information, allowing models to identify linguistic patterns. We utilized the Word2Vec technique because it can capture relationships between words, even if they were not seen during training [1]. We adopted Google's pre-trained Word2Vec model, which generates high-dimensional word vectors based on the distributional hypothesis. Compared to methods like Bag-of-Words (BOW) or TF-IDF, Word2Vec approach retains semantic information while reducing the computational cost of high-dimensional features. For words not present in the pre-trained model, zero vectors are used to avoid data loss.

To address class imbalance in the corpus, we used SMOTE to generate synthetic samples for underrepresented classes by interpolating between existing instances. This approach prevents bias toward majority classes, promotes balanced decision boundaries, and enhances the model's ability to generalize across all MBTI types. We set a target of 1000 samples per MBTI type and use a pipeline combining SMOTE and RandomUnderSampler method to simultaneously oversample minority classes and undersample majority classes. The resampling ensures balanced class distribution for robust training. Figure 2 shows that after resampling, we encoded MBTI types into four binary dimensions (I/E, S/N, F/T, J/P) of equal data size 8000.

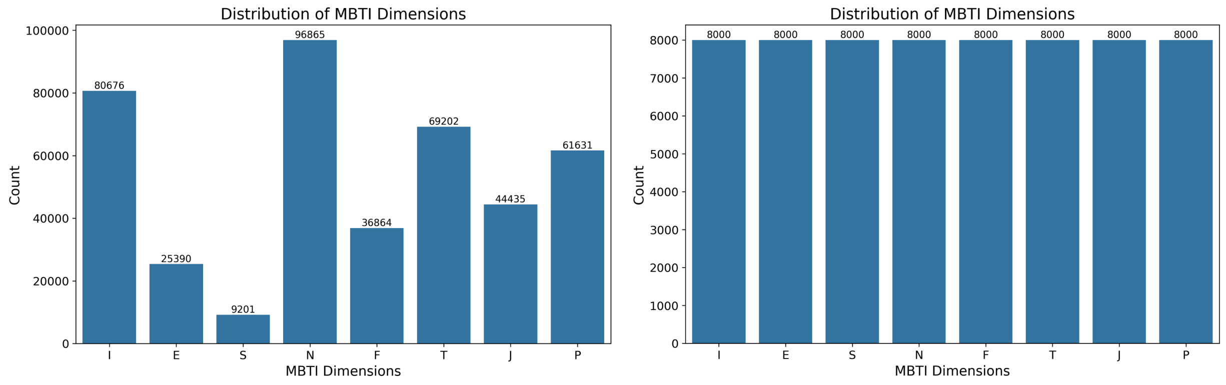


Figure 2. Distribution of MBTI eight dimensions

4.3 Exploratory Data Analysis (EDA)

Distribution of word counts in posts:
We could see that the number of words in most of the posts is between 450 with 500 from Figure 3.

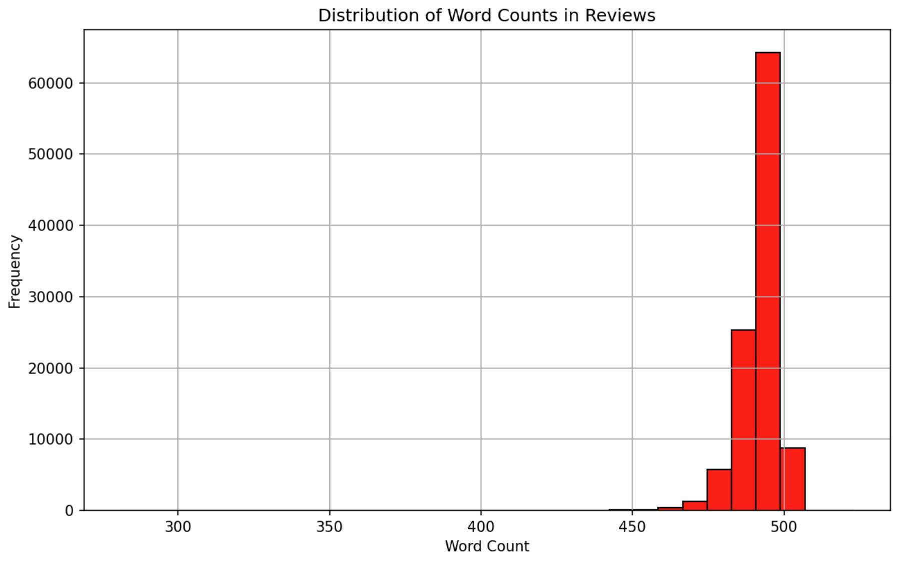


Figure 3. Distribution of Word Counts in Posts

Plot of comparison of sample numbers of two categories in four dimensions:
We can see from Figure 4. that there is indeed a problem of data imbalance in the data of the four dimensions, where the comparison of data volume between the two types of S/N classification reaches about 1:9.

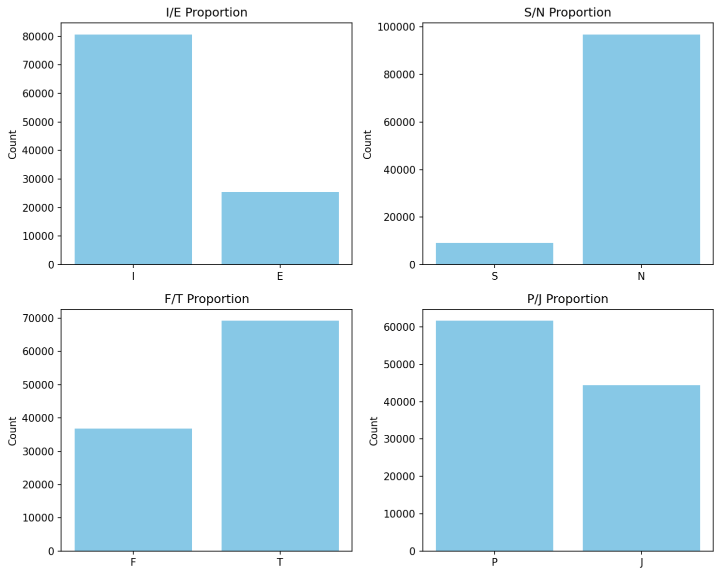


Figure 4. Comparison of sample numbers of two categories in four dimensions

WordCloud visualization:
We can see from Figure 5. regardless of the MBTI personality type, the six most common words are: think, people, one, say, feel, know.

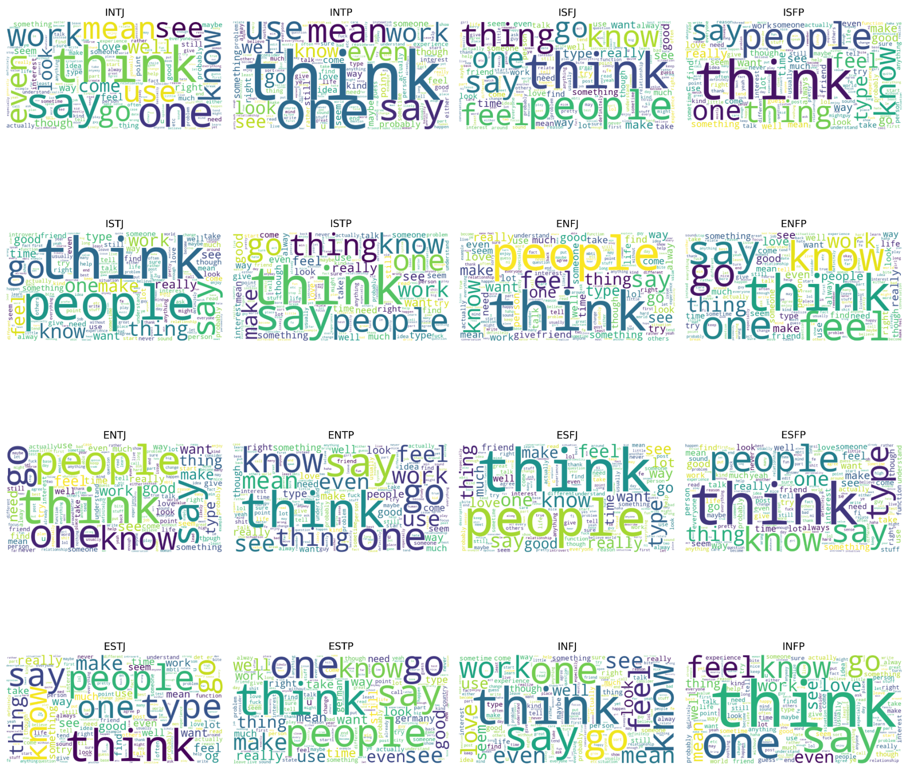


Figure 5. Word cloud

4.4 Models Implementation

In this study, we applied deep learning techniques, specifically BiLSTM and GRU layers, to predict MBTI types based on social media text. We split the dataset into 80% training, 10% validation, and 10% testing sets to ensure that the model is trained on a majority of the data while having sufficient data for model validation and evaluation. Since we need to understand the context within social media posts, we chose LSTM and GRU layers for their ability to handle sequential data and capture long-term dependencies in text sequences. Both models were implemented with bidirectional layers, allowing the model to process the input sequence in both forward and backward directions, thereby enhancing its ability to capture context from both ends of the sequence.

To optimize the learning process, we used the Adam optimizer to update the model's weights, calculated based on Equation (1). The ReLU activation function was applied in the hidden layers due to its ability to introduce non-linearity and help the model learn complex patterns. Since each MBTI dimension (I/E, S/N, F/T, J/P) was treated as a binary classification task, we employed a Sigmoid activation function in the output layer and used a binary cross-entropy loss function. Dropout layers were added to prevent overfitting, ensuring better generalization to unseen data.

The models were trained for 10 epochs with a batch size of 32. During model training, metrics such as accuracy and F1 score were used to evaluate performance. The results were reported by classification_report and visualized through loss and accuracy curves. Our methodology ensured robust training and evaluation, allowing us to effectively assess the models' performance in predicting MBTI types from social media text. This, in turn, supports our goal of enhancing recommender systems to deliver more personalized and context-aware recommendations.

The BERT model processes user text data in several stages, shown in Figure 6. First, the Input Layer uses DistilBertTokenizer to tokenize the text, producing input_ids and attention_mask with a maximum sequence length of 256. These are passed to the Embedding Layer, which converts the input into contextual embeddings with a hidden size of 768. Next, the Transformer Encoder, composed of 6 layers with 12 attention heads and 3072 hidden units, captures deep contextual relationships and outputs refined feature representations. The Pre-classifier Layer applies a fully connected layer with ReLU activation to reduce dimensionality and enhance feature extraction. Finally, the Classifier Layer maps the hidden features to the target labels using a fully connected layer. The classification task is optimized using CrossEntropyLoss and the Adam optimizer, ensuring accurate predictions for MBTI binary classification tasks across four dimensions.

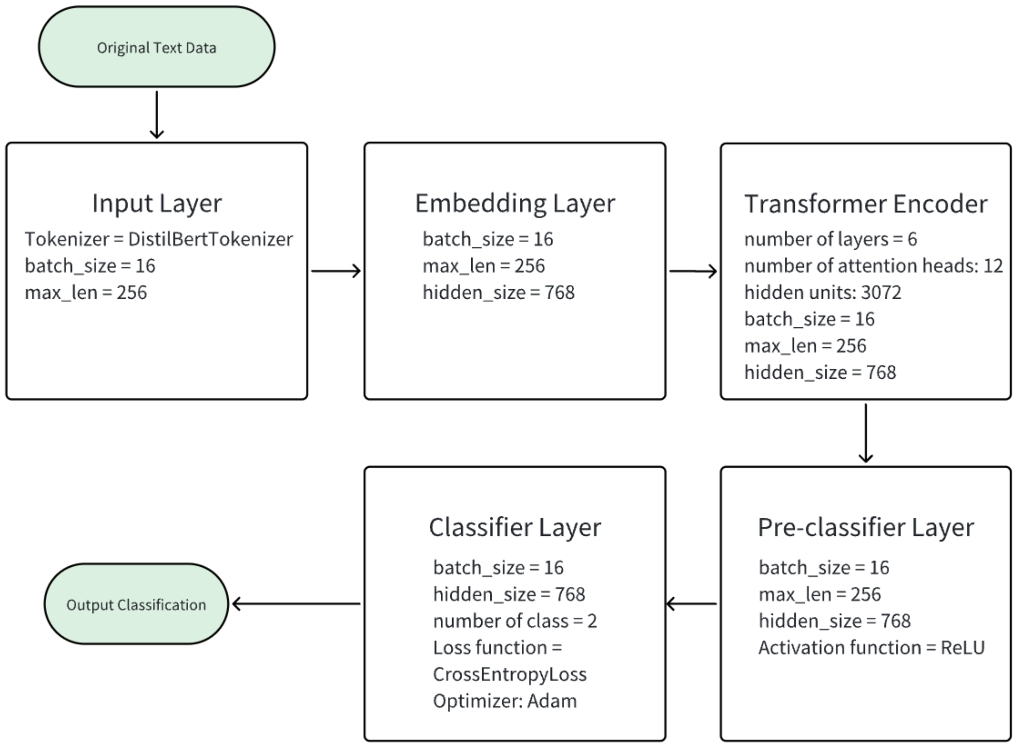


Figure 6. Distribution of MBTI eight dimensions

4.5 Results and Optimization

Model	MBTI Type			
	I/E	S/N	F/T	J/P

LSTM	72.56%	77.75%	81.44%	70.63%
GRU	71.47%	78.03%	81.00%	70.16%
BERT	77.11%	90.96%	81.45%	75.64%

Table 1. Accuracy of models

In Table 1, we observe that the BERT model achieved the best performance across all four MBTI dimension classification tasks. Specifically, the BERT model obtained accuracy rates of 77.11%, 90.96%, 81.45%, and 75.64% in the I/E, S/N, F/T, and P/J binary classification tasks, respectively.

To address the issue of data imbalance in GRU and LSTM models, we applied the SMOTE method. Although the accuracy decreased after applying SMOTE, it better reflects real-world scenarios, as imbalanced data often leads models to overfit to the majority class. Similarly, for the BERT model, we incorporated a WeightedRandomSampler to calculate the weight of each class and balance the class distribution in the training set, effectively mitigating overfitting caused by class imbalance.

Despite these optimization measures, the initial classification accuracy for the P/J task using the BERT model was below 60%. We hypothesize that this is due to the relatively weak textual feature differences between P and J. To address this, we implemented two key adjustments: first, we reduced the learning rate from 5e-5 to 1e-5; second, we increased the max_length parameter from 256 to 512. Through these iterative improvements, we gradually increased the classification accuracy from below 60% to 69.47%, and ultimately to 75.64%.

5. Conclusion

In our MBTI personality type classification task based on textual data, we achieved promising results. Among the three classification models evaluated, the BERT model demonstrated the best performance, achieving an accuracy of 90.96% in the S/N dimension classification task. Across all four dimensions, the model achieved an average accuracy of 81.29%. We believe that the proposed model and methodology could enhance the performance of personalized recommendation systems, not only saving users' time but also boosting business revenue.

5.1 Limitations

- Computational Constraints:
  Due to computational limitations, we truncated each post to the first 256 words in the BERT model, which might have reduced classification accuracy. Additionally, we only performed binary classification for each of the four MBTI dimensions instead of a comprehensive 16-type classification. If interdependencies exist among the four MBTI dimensions, a 16-type classification could yield more interpretable and meaningful results.
- Limitations of the MBTI Framework:
  Although MBTI is a widely adopted personality classification framework, individual behaviors

and personalities can also be influenced by factors such as mindset and life environment. Therefore, MBTI should not be considered the sole basis for certain decision-making processes.

5.2 Future work

With sufficient computational resources, we plan to extend the model to perform a full 16-type MBTI classification. Additionally, we aim to increase the maximum word limit for each review in the BERT model to 500, which could further improve classification performance. For the LSTM and GRU models, our goal is to conduct hyperparameter tuning using Ray Tune. Ray Tune offers significant advantages, including powerful distributed computing and various search algorithms—such as Grid Search, Random Search, Bayesian Optimization, and Hyperband—that intelligently select the best strategy for each task.

References

[1] Ryan, G., Katarina, P., & Suhartono, D. (2023). MBTI Personality Prediction Using Machine Learning and SMOTE for Balancing Data Based on Statement Sentences. Information, 14(4), 217. https://doi.org/10.3390/info14040217

[2] G. B. Mohan, R. P. Kumar, E. R and S. Gorantla, "Enhancing Personality Classification through Textual Analysis: A Deep Learning Approach Utilizing MBTI and Social Media Data," 2023 International Conference on Network, Multimedia and Information Technology (NMITCON), Bengaluru, India, 2023, pp. 01-06, doi: 10.1109/NMITCON58196.2023.10276193.

[3] Ontoum, S., & Chan, J. H. (2022). Personality type based on Myers-Briggs Type Indicator with text posting style by using traditional and deep learning. arXiv. https://arxiv.org/abs/2201.08717

[4] MBTI Personality Types 500 Dataset.
https://www.kaggle.com/datasets/zeyadkhalid/mbti-personality-types-500-dataset/data

[1]Yenting Kuang conducted data pre-processing, performed EDA and designed and trained both LSTM and GRU models

[2]Siyuan Gao conducted data pre-processing, performed EDA and designed and trained the BERT model

[3]Keerthana Balswamy participated in the project discussion