PersonaLens: MBTI Analysis with Transformer Technology

Xuetong Tang, Yicheng Lu, Ke Zhang, Emma Sun

Introduction

Our project aims to revolutionize the way Myers-Briggs Type Indicator (MBTI) assessments are conducted. Traditional MBTI assessments, with their lengthy questionnaires, can be time-consuming and daunting for users. Transformer, a popular tool for language processing tasks, captures contextual information of human language and has great potential at predicting MBTI through text inputs. In our project, we leverage advanced Transformer-based natural language processing (NLP) technology to analyze and interpret users' textual responses swiftly and accurately. Exploring this technology for MBTI classification tasks opens up possibilities for a modernized platform for personality assessment, catering to the increasing interest in various domains such as social media analysis, marketing, and personalized recommendation systems.

This project is a multi-classification problem. Here, our target variable includes 16 classes each representing one specific MBTI personality. We could also understand our target task as 4 binary classification problems where we would perform binary classification on each key dichotomy of the MBTI personality code.

Our dataset is collected from Kaggle. It was originally collected through the PersonalityCafe forum. This website provides a large selection of people and their MBTI personality type, as well as corpses of their posts. This dataset contains over 8600 rows of data. Each row consists of the person's 4-letter MBTI and the most recent 50 things they have posted (Each entry is separated by "|||"). This dataset provides a rich and diverse source of information for training and evaluating our model.

Additionally, we're drawing inspiration from related work by Tejas Pradhan et al., titled "Analysis of Personality Traits using Natural Language Processing and Deep Learning." This study explores automating personality assessments using traditional machine learning methods as well as CNN models.

Methodology

1. Preprocessing

The first step involves text cleaning to ensure that the data is free from unwanted characters, such as punctuation and special symbols, which could interfere with the subsequent analysis. Following this, we employ the BERT Tokenizer from the transformers library by Hugging Face. This tokenizer utilizes WordPiece tokenization, where common words are kept intact while less frequent words are split into subwords. The tokenizer then maps each token to an index based on a predefined vocabulary, ensuring that each token has a unique index. Additionally, special tokens such as [CLS] are added at the beginning of each example to indicate the start of a sentence, while [SEP] tokens mark the end of segments or separate sentences. To ensure uniformity in input lengths, padding is applied to shorter sequences using [PAD] tokens, while longer sequences are truncated to a maximum length. This preprocessing ensures that our textual data is properly formatted and ready for input into our model.

2. Models

In our model architecture, we first employ a Multi-layer Perceptron (MLP) as our baseline approach. The MLP consists of multiple layers of neurons, each connected to the next layer, allowing for nonlinear transformations of the input data. Our MLP model consists of a shared layer part using relu activation and 4 output dense layers using a sigmoid function to do binary classification. There are 2

dense layers in the shared layers. One with 1024 neurons and the second one with 256 neurons. All 4 output dense layers are tuned to use 128 neurons.

Next, we employ Text Convolutional Neural Networks (CNNs) with a pre-trained Word2Vec embedding model. CNNs are initially designed for tasks involving image recognition and classification due to their ability to efficiently capture spatial hierarchies in data. In our case, CNNs are applied to learn features from textual data by convolving filters over the input text and extracting meaningful patterns and structures.

Our final approach utilizes the pre-trained Bidirectional Encoder Representations from Transformers (BERT), specifically using the BertTokenizer to preprocess textual data. This tokenizer adds a special [CLS] token at the beginning of each sequence, which helps the aggregation of contextual information from the entire sequence and generates a single representative embedding for the classification task. Since case distinction is not crucial in our classification task, we enhanced the model's ability to handle text variability by using the BERT model configured with uncased settings, treating uppercase and lowercase letters as equivalent, the model's ability to handle text variability is enhanced. To enhance the model's generalization capability, we integrate a dropout layer with a rate of 0.1 immediately after the BERT output to prevent the model from overfitting to the noise in training data.

Furthermore, the model includes four separate dense layers with sigmoid activation functions to independently predict the presence of one of the four dichotomies—Introversion/Extroversion (I/E), Intuition/Sensing (N/S), Feeling/Thinking (F/T), and Perceiving/Judging (P/J). Each layer produces a binary output, indicating the likelihood that a specific personality dichotomy applies to the input text. This multi-layer structure allows for nuanced predictions that reflect complex human personality traits based on textual analysis. This method offers a robust tool for analyzing personality traits through text, allowing us to reach our aim of predicting MBTI through text inputs, instead of utilizing the traditional lengthy MBTI assessments approach.

Layer (type)	Output Shape	Param #	Connected to
attention_mask (InputLayer)	[(None, None)]	0	[]
input_ids (InputLayer)	[(None, None)]	0	[]
tf_bert_model_3 (TFBertModel)	TFBaseModelOutputWithPoolingAndCrossAttentions(last_hidden_state=(None, None, 768), pooler_output=(None, 768), past_key_values=None, hidden_states=None, attentions=None, cross_attentions=None)	109482240	['attention_mask[0][0]', 'input_ids[0][0]']
dropout_151 (Dropout)	(None, 768)	0	['tf_bert_model_3[0][1]']
output1 (Dense)	(None, 1)	769	['dropout_151[0][0]']
output2 (Dense)	(None, 1)	769	['dropout_151[0][0]']
output3 (Dense)	(None, 1)	769	['dropout_151[0][0]']
output4 (Dense)	(None, 1)	769	['dropout_151[0][0]']

Total params: 109485316 (417.65 MB)
Trainable params: 109485316 (417.65 MB)
Non-trainable params: 0 (0.00 Byte)

3. Metrics and Model Evaluation

The success of our project hinges on having a strong evaluation strategy to gauge the precision of our MBTI-type forecasting model. To accomplish this, we employ a diverse set of metrics, including accuracy, F1-score, precision, recall, and log loss. Each metric provides unique insights into the model's performance, allowing us to gain a comprehensive understanding of its predictive abilities and areas for improvement.

Additionally, to assess our progress, we'll compare our CNN and transformer models' performance against established approaches. Our baseline comparison involves using a multi-level perceptron (MLP) equipped with 4 sigmoid layer classifiers, a common dense layer choice for classification tasks. Furthermore, we implement the convolutional neural network (CNN) method. as an advanced benchmark. These comparisons will provide valuable context regarding the efficiency and effectiveness of our model in the domain of personality prediction.

Results

In our evaluation of the models, we present the outcomes from three different approaches: the transformer-based model, the multi-level perceptron (MLP) baseline, and the convolutional neural network (CNN) benchmark proposed by Pradhan et al. Firstly, the transformer-based model demonstrated promising performance across various metrics. With a mean accuracy of 84.4%, it exceeded our baseline target of 60%, showcasing its effectiveness in MBTI type prediction. Additionally, the model achieved overall F1-scores of 70.84, indicating its ability to maintain precision across different class distributions. However, the overall accuracy for correctly predicting all four categories is only 58.5%.

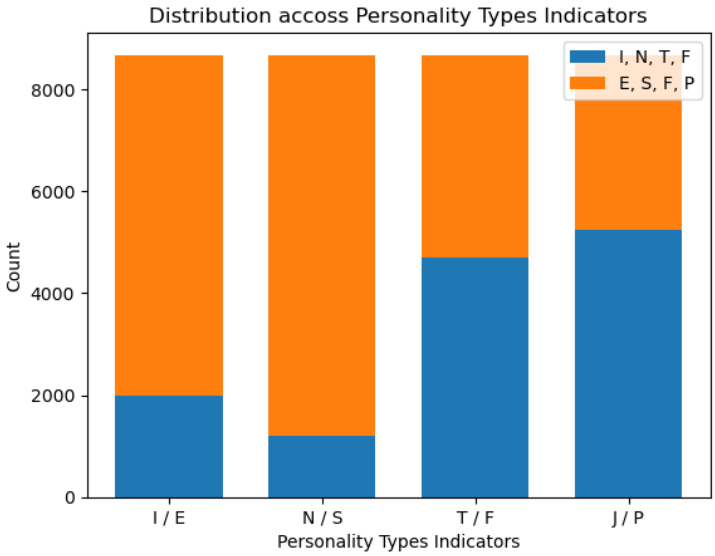
Aside from the potential impacts of the imbalanced nature of our dataset, another possible contributor to this relatively low accuracy could be the way we preprocessed the dataset. Initially, the target variable of our dataset was a sequence representing an MBTI type, for example, 'INTJ'. We divided this sequence into four individual categories, each serving as a separate binary target variable. Since the manifestation of one category could be interrelated to another category, the separation of the target variable could possibly lead to significant information loss regarding such sequence, resulting in relatively lower accuracy scores.

In comparison, the MLP baseline model attained a mean accuracy of 81.8% and an overall accuracy of around 47.8%, which is surprisingly better than expected but falling short of the transformer-based model's performance. Meanwhile, the CNN benchmark model only achieved a mean accuracy of 68.3% and an overall accuracy of 0.212, positioning it below the transformer-based model and the MLP baseline in terms of performance. Notably, the CNN model demonstrated a strong overfitting problem, where the validation score falls behind the training score, which might explain the poor performance of this model.

Overall, these results highlight the transformative potential of advanced models like transformers in MBTI-type prediction tasks. While the MLP baseline and CNN benchmark models provide valuable benchmarks, the transformer-based approach offers superior performance across multiple metrics, setting the stage for the development of a highly accurate and reliable personality assessment tool.

Challenges

One significant challenge we face with our dataset is the imbalance in the number of samples for certain personality traits, particularly between Introverted (I) and Extroverted (E), and Intuitive (N) versus Sensing (S), with ratios of approximately 3:1 and 4:1, respectively. This imbalance can hinder the model's ability to effectively learn and extract features from the less represented classes. As a result, when encountering new instances of these underrepresented classes, the model may exhibit poor generalization, potentially leading to higher rates of misclassification.



Moreover, another issue we face is overfitting, where the model becomes overly specialized in recognizing patterns specific to more frequent classes. While the accuracy of the training set continues to increase, there is minimal improvement or even a decline in performance on the validation set. This discrepancy suggests that the model is memorizing the training data rather than learning generalizable patterns, thus compromising its ability to perform well on unseen data. Addressing these challenges is essential to ensure that our model can effectively capture the nuances of all MBTI personality types and generalize well to new examples.

Reflection

Key lessons learned during preprocessing involve considerations regarding the choice between BERT Tokenizer and traditional methods like TF-IDF and Word2Vec. One crucial distinction lies in contextual understanding: BERT Tokenizer provides contextual embeddings, meaning word representations vary based on their context, while TF-IDF and Word2Vec offer fixed representations regardless of context. Additionally, the dimensionality of embeddings differs significantly: BERT embeddings are high-dimensional and dense, Word2Vec produces dense embeddings of lower dimensionality, and TF-IDF yields high-dimensional but sparse vectors. By comparing our 3 models, we find that the MLP baseline model using TF-IDF performs well and does not have overfitting issues while our advanced CNN model using Word2Vec and the transformer model using BertTokenizer both are experiencing overfitting problems. Considering our relative small data size, we conclude that advanced word embedding techniques may not always be a good fit. Choosing a suitable bag

of words embedding method could outperform complex vectorization methods depending on the size of the dataset and length and content of corpuses.

Moreover, the performance and complexity of these methods play a vital role. BERT models are more intricate and demand higher computational resources, yet they excel in tasks requiring nuanced context understanding. On the other hand, TF-IDF and Word2Vec are less resource-intensive and can be highly effective for tasks with minimal contextual dependency. Understanding these differences allows for informed decision-making during preprocessing, ensuring that the chosen approach aligns with the specific requirements and constraints of the task at hand.

We are basically satisfied with how the project ultimately turned out. By exceeding our baseline target with a mean accuracy of 84.4% and achieving an F1-score of 70.84, the transformer-based model demonstrated its robust capability in classifying MBTI types based on text inputs. This suggests that our approach to using advanced NLP technology not only meets but surpasses traditional methods in efficiency and effectiveness. The transformer model exceeded our expectations in terms of accuracy but lacks ability to handle the class imbalances inherent in our dataset. Initially, our approach relied heavily on traditional NLP methods like TF-IDF for text representation. However, as we delved deeper into the project, we pivoted towards using Transformer-based models due to their superior performance in capturing contextual nuances in language. If we could do the project over again, we might start immediately with a more focused exploration of Transformer architectures and we would consider implementing more sophisticated techniques for handling imbalanced data. With more time, we would look into more advanced strategies for dealing with data imbalance, such as synthetic data generation or more nuanced resampling techniques. One of the biggest takeaways from this project is the effectiveness of Transformer models in processing natural language and their applicability to psychological profiling. We learned the importance of data quality and balance in training machine learning models and the impact of model choice on the performance of NLP tasks.

Outlook

In tackling the challenge of imbalanced data, we can implement advanced resampling techniques to address this issue effectively. One approach involves data augmentation, particularly for the underrepresented classes in our dataset. Techniques such as Synthetic Minority Over-sampling Technique (SMOTE) or paraphrasing existing text can be employed to generate synthetic data, thereby diversifying our dataset with semantically similar yet varied entries. Additionally, experimenting with various resampling strategies is crucial. This includes oversampling the minority classes or undersampling the majority classes to achieve a more balanced distribution and mitigate the impact of class imbalance on our model's performance.

Furthermore, to enhance the richness of our feature set, we can consider incorporating additional features beyond the textual content. Psycholinguistic features, derived from

psycholinguistic databases like Linguistic Inquiry and Word Count (LIWC), offer valuable insights into the psychological and emotional states conveyed in text. By integrating these features into our model, we can capture deeper aspects of language usage related to personality traits, thereby enriching our classification process and improving the model's predictive performance. These strategies enable us to address key challenges in our data and enhance the robustness of our classification model for MBTI personality types.

Reference

[1] T. Pradhan, R. Bhansali, D. Chandnani and A. Pangaonkar, "Analysis of Personality Traits using Natural Language Processing and Deep Learning," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2020, pp. 457-461, doi: 10.1109/ICIRCA48905.2020.9183090.

[2] (MBTI) Myers-Briggs Personality Type Dataset from Kaggle. Retrieved April 10, 2024 from https://www.kaggle.com/datasets/datasnaek/mbti-type.

GitHub Repository

https://github.com/kzhangaz/Detecting_Your_MBTI_Using_NLP

Devpost

https://devpost.com/software/mbti-classification?ref_content=userportfolio&ref_feature=in_progress