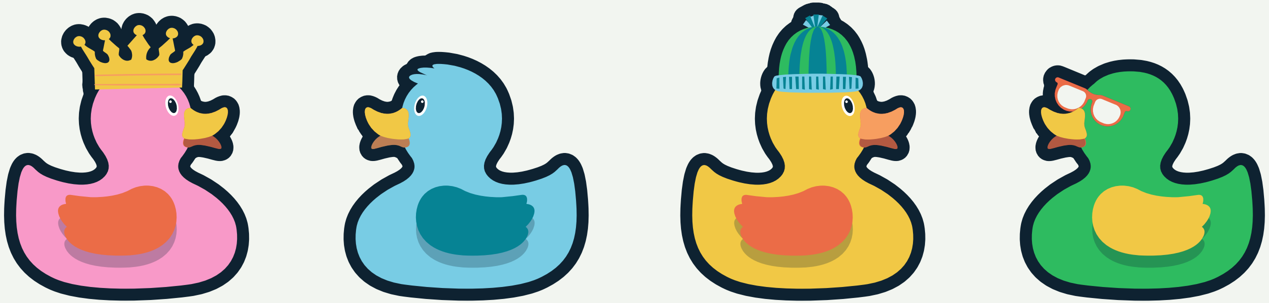
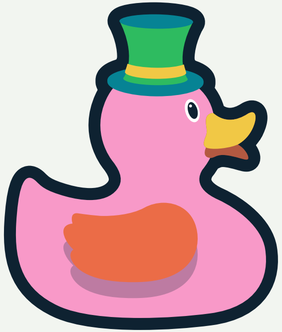


PersonaLens: MBTI Analysis with Transformer Technology

Xuetong Tang(xtang34), Yicheng Lu(ylu204), Ke Zhang(kzhan176), and Emma Sun (dsun35)

Table of contents

Introduction



What is MBTI?

Myers-Briggs Type Indicator (MBTI) is a personality assessment tool which categorizes individuals into 16 different personality types based on their preferences:

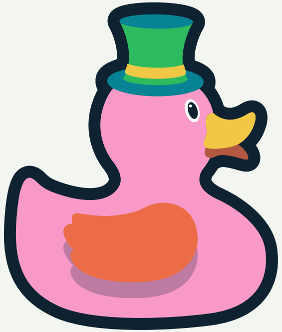
- Extraversion (E) vs. Introversion (I)
- Sensing (S) vs. Intuition (N)
- Thinking (T) vs. Feeling (F)
- Judging (J) vs. Perceiving (P)

Each type is a combination of these preferences: ISTJ (Introverted, Sensing, Thinking, Judging).

Why Transformer for MBTI classification?

- Traditional MBTI questionnaires are lengthy and time-consuming.
- Transformers captures contextual information of human language
    - great potential at predicting MBTI through text inputs!

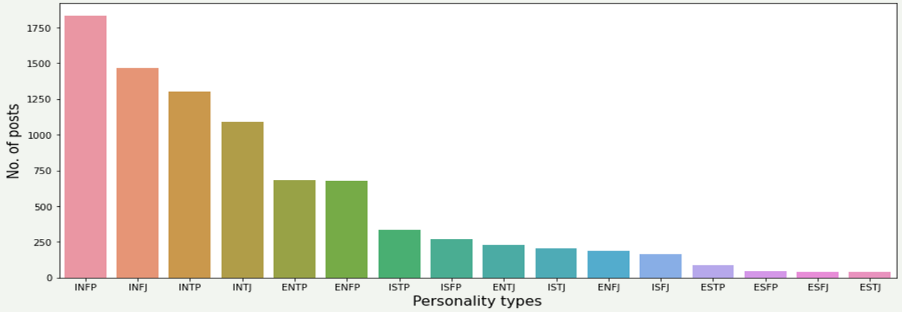
Methodology

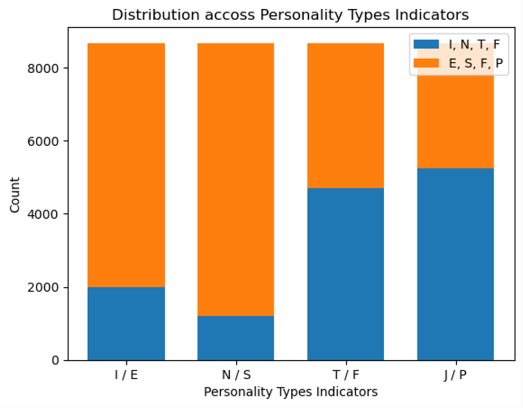


- Multi-layer Perceptron (MLP)
- Convolutional Neural Network (CNN)
- Transformer (BERT) + Extra Classification Layer

Our Dataset

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 consist of the the person's 4 letter MBTI and the most recent 50 things they have posted (Each entry separated by "|||").





ENFP	'He doesn't want to go on the trip without me, so me staying behind wouldn't be an option for him. I thi

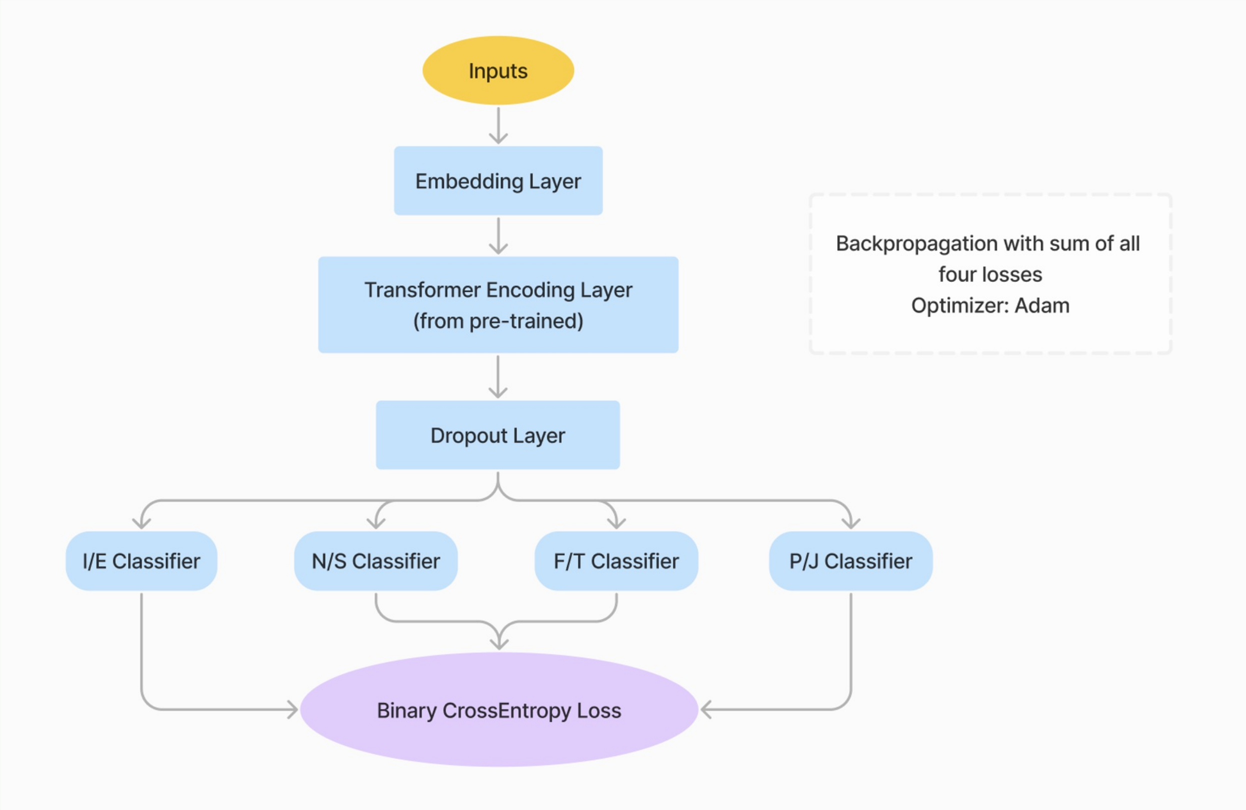
Preprocessing

1st step: Text cleaning

2st step: Bert Tokenizer (transformers library by Hugging Face)

- **Tokenization**: WordPiece tokenization where frequent words are kept whole, rare words are split into subwords.
- **Vocabulary Mapping**:
  - token is mapped to an index based on a pre-defined vocabulary
  - vocabulary contains a fixed list of tokens with unique index
- **Adding Special Tokens:**
  - [CLS] at the beginning of each sequence
  - [SEP] to separate segments/mark sentence end
- **Padding and Truncation:**
  - pads shorter sequences with [PAD]
  - truncates longer sequences to maximum length

Model Architecture



Pretrained model:

- `'bert-base-uncased'`

Parameter tune:

- Learning rate = 0.00002
- Dropout rate = 0.1
- Loss function: `BinaryCrossentropy`

Results

Transformer Results

Precision

I/E: 64.94%
N/S: 76.47%
F/T: 80.61%
P/J: 80.00%

Mean: 75.50%

Recall

I/E: 74.09%
N/S: 52.00%
F/T: 85.61%
P/J: 61.00%

Mean: 68.17%

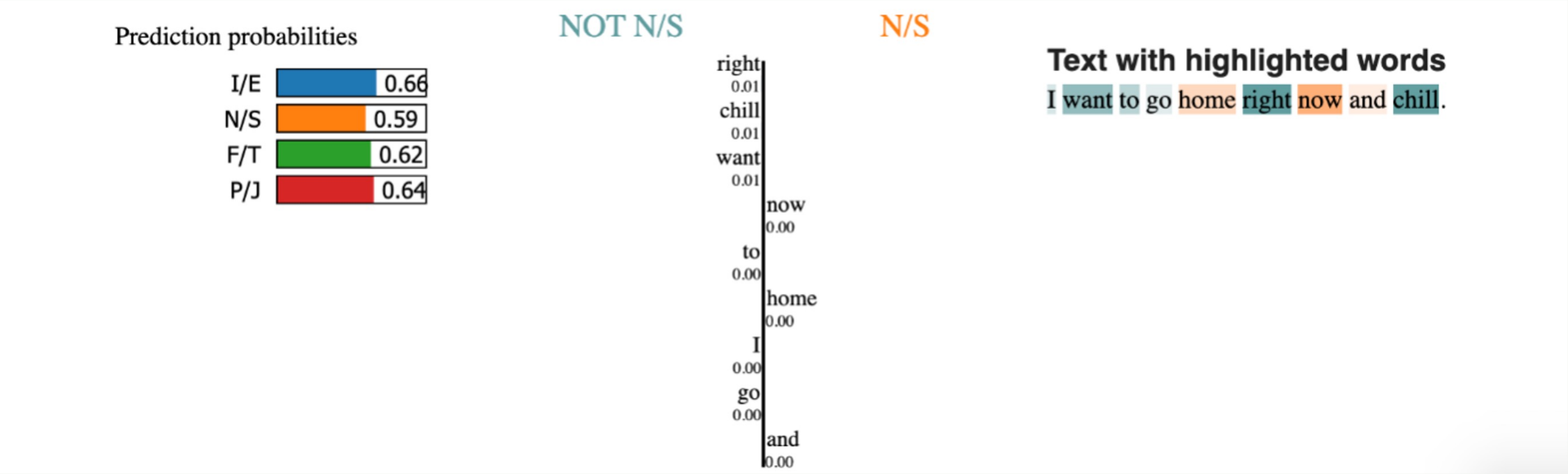
F1 Score

I/E: 69.21%
N/S: 61.90%
F/T: 83.03%
P/J: 69.22%

Mean: 70.84%

Quick Interpretation

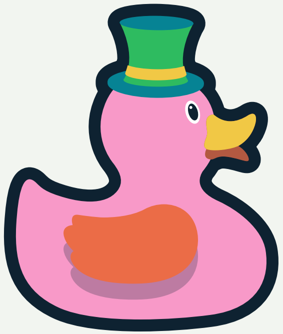
Text input: “I want to go home right now and chill.”



Comparison between models

Test scores
(round to 3 decimal points)

	I/E	N/S	F/T	P/J	Mean	Overall
MLP	0.841	0.878	0.825	0.727	0.818	0.478
CNN	0.734	0.855	0.616	0.525	0.683	0.212
Transformer	**0.866**	**0.893**	**0.831**	**0.787**	**0.844**	**0.585**



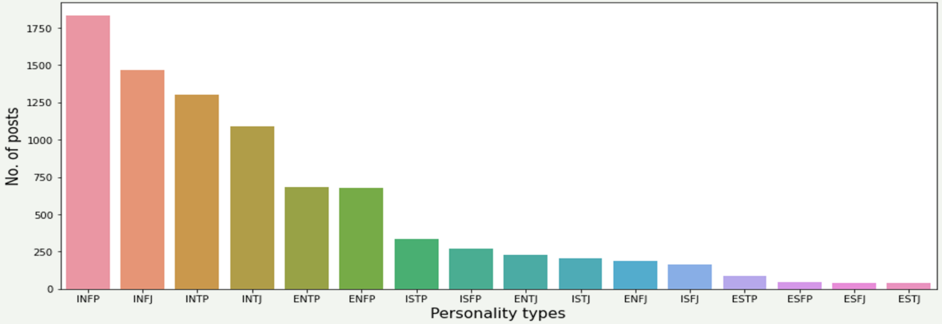
Discussion

- lessons learned
- lingering problems/limitations with your implementation
- future work (i.e. how you, or someone else, might build on what you've done)

Problems in implementation

1. **Data imbalance** - # of samples with I and N personality is significantly higher
   - Poor Generalization: it might not learn sufficient features of the less common classes. When faced with new examples of these underrepresented classes, it doesn't generalize well
   - Overfitting to Frequent Classes: The model end up being very good at recognizing patterns specific to more common classes but fail to capture broader patterns that apply to all classes

2. **Overfitting problem**
   - Accuracy in Training set keeps increasing but not much improvements on validation set after 3 epochs.



Lesson learned: Concerns in Preprocessing

BERT Tokenizer Vs. TF-IDF and Word2Vec

- **Contextual Understanding:** BERT provides contextual embeddings, meaning the representation of a word can change based on the surrounding words. TF-IDF and Word2Vec do not provide contextual embeddings; the representation of a word is the same regardless of its context.
- **Dimensionality:** BERT embeddings are very high-dimensional and dense. Word2Vec also produces dense embeddings but typically of lower dimensionality than BERT. TF-IDF produces high-dimensional but sparse vectors.
- **Performance and Complexity:** BERT models are generally more complex and require more computational resources. They tend to perform better on tasks requiring understanding of context. TF-IDF and Word2Vec are less resource-intensive and can be very effective for tasks with less contextual dependency.

Future Work

1. Advanced Handling of Imbalanced Data

- Data Augmentation: For underrepresented classes in the dataset, we could generate synthetic data using techniques like SMOTE or by paraphrasing existing text to create varied yet semantically similar entries.
- Resampling Techniques: Experiment with different resampling strategies to balance the dataset, either by oversampling the minority classes or undersampling the majority classes.

2. Incorporation of Additional Features

- Psycholinguistic Features: Include features from psycholinguistic databases like LIWC, which can provide insights into the psychological and emotional states conveyed in text.

Thanks!