# A Deep Learning model for Personality Detection with BERT and RoBERT

E4040.2020Fall.SH.report
Samrat Halder, sh3970
*Columbia University*

## Abstract

*Identifying human personality based on social media data has become an active area of research in recent years. Such machinery can help companies to better target customers with more relevant products and advertisements. Some work on this topic uses the traditional Bag-of-words approach with machine learning models. We explore several deep learning-based models, BERT and RoBERT, and also address other issues with the dataset in this project. The dataset is labeled with Myers-Briggs Type Indicators (MBTI) and contains social media comments for 9k users. We achieve 41% accuracy on an 8-class and 43% accuracy on a 4-class classification problem with MBTI type personality detection in our research.*

## 1. Introduction

Social media platforms are becoming increasingly popular among users of all ages in recent years. People not only interact with each other but also express their views about various topics. There are various restrictions on using this data and it is also challenging to obtain a consolidated dataset that anonymously tags users to their comments over a long time. This data can be used in several ways for online marketing and is of utmost importance to companies. One interesting problem is using social media data to identify human personality. Personality is defined as the individual patterns of thinking, feeling, and behaving (Corr and Matthews, 2009). There has been an increasing interest in automated personality prediction from social media from both the natural language processing and social science communities (Nguyen et al., 2016). One interesting piece of work has been done by Matej Gjurković, Jan Šnajder, 2018. The authors focused more on preparing the dataset and extracting various features which are important to personality type. We explore a very different approach where we use an dataset[1] which is available on request from the authors and implement state-of-art deep learning models, eg. BERT (Devlin et al., 2019), RoBERT (Pappagari et al., 2019) to find out if textual semantic contains similar information to that of handcrafted features in the context of human personality. We understand we miss out non-textual information, eg. how much a user is active on social media, how frequently one does comment, how long is their average comment length etc. which definitely contains important information about human personality. However, we hope that semantic features capture enough information to understand personality traits.

There are several APIs and wrapper developed for BERT by various researchers however mostly they are customized for a very particular use. Therefore we decide to use the original BERT research model developed by Google. Usually, BERT is a heavy model requiring a tremendous amount of computing resources and careful fine-tuning to make it work for a specific task. To overcome the technical resources, we used Google cloud's computing platform with high-end GPUs to train our model and build a prototype model that looks very promising. Furthermore, BERT works best at sentence level tasks. We used various techniques that are discussed in detail later to build a customized classification model for this project.

## 2. Summary of the Original Paper

We mainly referenced three papers for this project. Matej Gjurković, Jan Šnajder, 2018 discusses a Reddit dataset that contains comments by users tagged with their personality type. The paper proposes a feature-based machine learning model for automatic identification of personality types from social media comments. Devlin et al., 2019 proposes a state-of-art language modeling architecture based on attention called BERT which has been shown to improve on other NLP models for various language modeling tasks. Pappagari et al., 2019 addresses the issue that BERT may not be a recommended model when it comes to long texts. It discusses models like RoBERT and ToBERT which are built upon BERT for document classification.

### 2.1 Methodology of the Original Paper

In the original paper, the authors discuss the acquisition of the dataset. The database of Reddit posts

[1] http://takelab.fer.hr/data/mbti

and comments covers the period from 2005 till the end of 2017, totaling more than 3 billion comments and increasing at the rate of 85 million comments per month. After some rigorous data cleaning, they found there are about 9k users who had provided a specific MBTI personality type (16 dimensions). Authors also derive an array of linguistic (n-grams based tf-idf, specific psycholinguistic words lists) and non-linguistic features (user activity, topic distributions, number of comments, self comments, etc.). Also, they include the temporal aspect of one's activities by considering the time intervals between comment timestamps (the mean, median, and maximum delay), as well as daily, weekly, and monthly distributions of comments, encoded as vectors of corresponding lengths. The authors create a very rich set of features that might be relevant to an individual's personality type and used various machine learning models including support vector machine (SVM), 2-regularized logistic regression (LR), and a three-layer multilayer perceptron (MLP).

## 2.2 Key Results of the Original Paper

The most important results from the original personality detection paper for our purposes are as follows:

| | Dimensions | | | | |
|---|---|---|---|---|---|
| Model | E/I | S/N | T/F | J/P | Type |
| LR all | 81.6 | 77.0 | **67.2** | **74.8** | 40.8 |
| MLP all | **82.8** | **79.2** | 64.4 | 74.0 | **41.7** |
| SVM all | 79.6 | 75.6 | 64.8 | 72.6 | 37.0 |

While we start this as a benchmark, later because of certain caveats in the dataset we follow a different approach with the various classes. This has been discussed in detail in section 5.

# 3. Methodology

In this project, we have used a very different approach than the original paper. Instead of re-building the features that are mentioned in the original work we have completely relied on semantic features of sentence or embeddings. Considering how much success the embedding based models have seen in the last few years in NLP, our assumption is that semantic features of sentences of user comments can capture sufficient information about personality type. We use a recent deep learning model BERT published by Google research (Devlin et al., 2019) for encoding and classification tasks. Furthermore, we implement Recurrent over BERT (RoBERT, Pappagari et al., 2019) to consider the long document size of individual user comments.

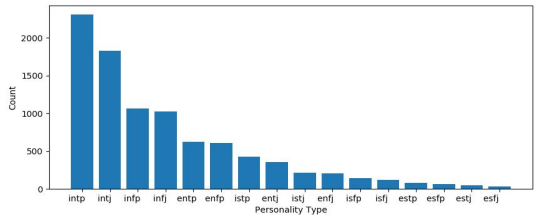
In the following sections, we discuss a detailed analysis of the dataset, various caveats, and our novel approach to reformulate the problem from a different perspective and technical details about the implementation.

## 3.1. Objectives and Technical Challenges

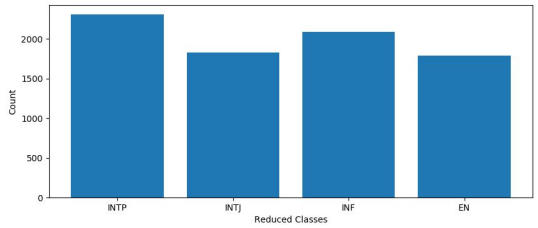
Our main objective of this project was to explore the embedding based approaches for this classification task. This problem is very different from any traditional textual classification model eg. spam detection, sentiment analysis because in most problems we deal with small sentences where both machine learning and deep learning models perform well. Also, our approach to this problem evolved very differently from the original paper as over time we came up with certain alternate dimensions to the problem. Instead of targeting to achieve better results, our focus was on understanding if embeddings can capture the abstract idea of writing style and connect it with personality type. The major technical challenge was to set up and fine-tune the BERT model considering its complexity and size. Also, another fundamental problem was that BERT works best on small sentences and can support a sequence length of up to 512. However, in our corpus, the average length of documents from each unique user was between 1800 to 2600 sentences across various personality types. In addition to this, the document length varies a lot which clearly indicates the length of the comments is definitely an important indicator. Please note that when we mention a document, it is an aggregation of all the comments from a particular user over the entire time period. However, in our research, we completely ignore any non-linguistic features and rely on embeddings.

In the original paper, the authors have used the full documents for each individual user which is certainly a very robust approach and captures more information about individual users. However, we take account of the fact that for any production level task we may not have such long documents for each user. Because of this, we assume that even a small document for any user should contain significant information about the personality type. Then we had to decide what could be a reasonable size to be defined as a paragraph. Also, as mentioned previously we take account of the fact BERT works well on relatively small size documents. After some preliminary experiments, we decided to split comments into a sequence size of 150. This serves an additional purpose of generating more training samples for any novel machine learning or deep learning model.

The next challenge is that the various classes in the data set are highly imbalanced. {**intp**: 2312, **intj**: 1831, **infp**: 1067, **infj**: 1021, **entp**: 623, **enfp**: 603, **istp**: 428, **entj**: 358, **istj**: 213, **enfj**: 206, **isfp**: 142, **isfj**: 121, **estp**: 81, **esfp**: 60, **estj**: 49, **esfj**: 34}.




Looking at the class distribution we remove the following classes for their low counts {**esfp, estp, estj, esfj, istp, istj, isfp, isfj**}**.** But still our data remains highly imbalanced. So we come up with a new class distribution by aggregating almost similar personality types with a small number of counts while keeping the original classes that have high counts. Our new distribution of data is now: {**intp**: 2312, **intj**: 1831, **inf**: 2088, **en**: 1790}. Please note that this approach is very different from the original paper where the authors had split the dataset across only one direction. Now the class distribution across these new classes is almost balanced.




The major technical challenge was to set up the BERT module and modify it for our specific fine-tuning purpose taking account of all computational constraints. The original BERT model has two versions, BERT-base and BERT-large. For computational constraints, we decide to use BERT-base which is relatively smaller in size.

## 3.2. Problem Formulation and Design

We follow two methods: **1. BERT classifier**: After the data preparation we come up with the following distribution of personality types: {**intp**: 697132, **intj**: 635008, **inf**: 456402, **en**: 441865} by splitting the documents into sequences of length 150. We use a small fraction (10%) of this data to fine-tune the pre-trained BERT-base model and use a feed-forward four-layer neural network at the end of the encoder stack to classify the sequences into various classes. **2. RoBERT**: In this approach, we try to classify the whole document by leveraging the embeddings from BERT. As discussed earlier, since BERT works best on smaller length sequences, we split each document into sequences of length 150 but with an overlapping portion of length 25 between successive sequences. We get the following distribution of each class: {**intp**: 603985, **intj**: 584098, **inf**: 399023, **en**: 394236}. We use our previously trained fine-tuned model from the previous section to get embeddings for each sequence. Now we build an LSTM model having as input the vectors created from BERT embeddings, however, we have long text sequences and most of the time these sequences are variable. So we take batch size more than one (eg. 10), pad the shorter sentences to the max length of each batch. We call this model Recurrence over BERT or RoBERT (Pappagari et al., 2019).

## 4. Implementation

Below we give a brief overview of the BERT and RoBERT architecture, and then we discuss the hardware and software setup we used on Google Cloud virtual machine.

## 4.1. Deep Learning Network

**BERT**: The following diagram explains the overall flow of the classification tasks using BERT.

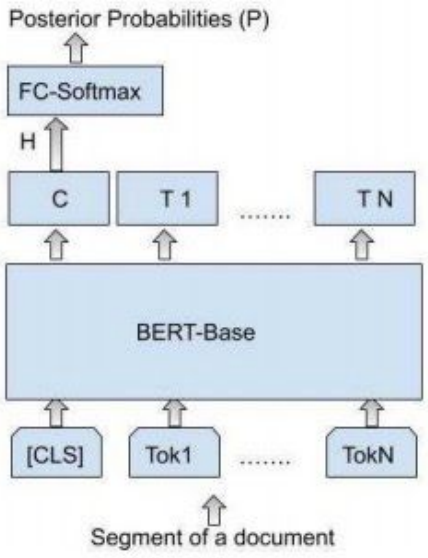



**Fig. 1**. BERT model for classification. H denotes BERT segment representations from last transformer block, P denotes segment posterior probabilities. Figure inspired from [2]

Original BERT has two versions: 1. BERT-base: L=12, H=768, A=12, Total Parameters=110M 2 BERT-large: L=24, H=1024, A=16, Total Parameters=340M. Here L represents the number of encoder layers of the transformer stack, H represents the dimension of the output, A represents the number of multi-headed attention in the encoder stack. In both versions, the number of layers in the feedforward network is set to 4. Also, both versions have two pre-trained models 1. Cased 2. Uncased. For our particular task, we do not feel the upper casing of the letters has any special significance and we use the Uncased model for BERT-base. There are two phases for using BERT: pre-training and fine-tuning. Fortunately, we already have a pre-trained model from Google research. For fine-tuning, we initialize the model with pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. We use this for the first part of our project where we have used to fine-tune the pre-trained model with our split sentence data and use the BERT classifier for the classification task.

**RoBERT**: Given that BERT is constrained by smaller input length because of O(n^2) time complexity, we split the input sequence into segments of a fixed size with overlap. For each of these segments, we obtain H ie. the last hidden state corresponding to [CLS] token of the input sequence from the BERT model. We then stack these segment-level representations into a sequence, which serves as input to a small (100-dimensional) LSTM layer. Its output serves as an embedding for the whole document. Finally, we use two fully connected layers with ReLU (30-dimensional) and softmax (same dimensionality as the number of classes) activations to obtain the final predictions.

We run two sets of experiments with both the models: 1. For all the 8-class dataset 2. For the 4-class dataset which is balanced.

## 4.2. Software Design

For all the computation we use Google Cloud virtual machine. We use the following setup: (8 core/ 30 Gb) Intel Haswell CPU, NVIDIA Tesla P100 GPU, Ubuntu 16.04. The module is compatible with Tensorflow-GPU version 1.11 Keras 1.0.7 Numpy 1.16.4 Cuda 9.0 python 3.6.10.

# 5. Results

## 5.1. Project Results

We have worked on a more difficult classification task than binary classification which was referenced in the original paper. We did not work on binary classification because if we divide the dataset across any dimension eg. **I/E** the classes become highly imbalanced. adfIt would require separate research work on how to use traditional oversampling techniques for embeddings and beyond the scope of this project. We mainly did two types of classification 1. The reduced set of personality classes (4-types) 2. All personality classes (8-types, imbalanced dataset).

With 4-class classification, we got an overall **43.43%** accuracy.

| Class | Precision | Recall | f-score |
|---|---|---|---|
| **INTP** | **0.49** | **0.42** | **0.45** |
| INF | 0.41 | 0.64 | 0.5 |
| INTJ | 0.41 | 0.41 | 0.41 |
| EN | 0.44 | 0.18 | 0.26 |

With all-class classification, the best we achieve is **41.16%** accuracy whereas the authors had reported the best accuracy of **41.7%** from a multilayer perceptron model which was applied on the whole document with all the derived linguistic and non-linguistic features.

| Class | Precision | Recall | f1-score |
|---|---|---|---|
| **INTP** | **0.47** | **0.45** | **0.46** |
| INTJ | 0.39 | 0.69 | 0.5 |
| INFP | 0.35 | 0.27 | 0.31 |
| INFJ | 0.4 | 0.24 | 0.3 |
| ENTP | 0.55 | 0.12 | 0.2 |
| ENFP | 0.37 | 0.1 | 0.15 |
| ENTJ | 0.38 | 0.05 | 0.09 |
| ENFJ | 0.28 | 0.04 | 0.07 |

We also run the 4-class classification task with RoBERT with a LSTM layer of size 50 and 5 training epochs for the full document on a dataset with around 7000 rows and achieve the best test set accuracy of **29.8%**.

**Computation time:** For 4-class classification the fine-tuning and training BERT-base model took 2 hr 25 mins with 230,428 examples (each of length 150 words). For the same classification task with RoBERT using 10 epochs, batch size of 5 took 2 hr 43 minutes with 7218 samples (whole document). Other computation times for data preparation and other experiments can be found in the log files.

## 5.2. Comparison of Results

A direct comparison of the results with Matej Gjurković, Jan Šnajder, 2018 is not necessarily

appropriate since we perform different experiments due to the class imbalance issue and model constraints. That said, we believe the model with BERT classifier did fairly well for both the classification tasks when compared to the results reported by the authors. Also, the main reason for the poor performance of the RoBERT model could be the lack of sufficient training examples. Overall, we believe the results were promising and with more data can be improved. This can definitely be an open research area.

### 5.3. Discussion of Insights Gained

In our opinion the results from this project are promising. Both the models performed similar to the machine learning models when it comes to multiclass classification with BERT showing very promising metrics. However, we were still constrained with computation time and memory and often had to use a subset of data for our training and classification tasks. What we have learned from these experiments is that embeddings do capture a wide array of characteristics from language, eg. human interaction and can save a lot of time behind custom feature engineering for traditional machine learning models. One must be careful while using the state-of-the-art BERT model. There are many versions of it and they are compatible with different versions of TensorFlow which is further linked to the Cuda version. So through this project, we learned how to carefully use these pre-trained language models for our particular task.

## 6. Conclusion

In this project, we work on a state-of-art language model BERT and further explore other models from the literature that can be used on long text documents. Although we started the project with an initial objective of beating the current machine learning model-based results, as we delved into the dataset we found alternative ways to proceed with the project. We try two methods. For "all types" classification tasks, we achieved 41.16% accuracy which is very close to the best results authors had achieved. However, one caveat is that authors used the full-text documents for their classification model while we used only documents of length 150 words. For our 4-class classification task, we achieved 43.43% accuracy. So we believe, by increasing the document length (which would require a huge amount of computational resources) much better results can be achieved. However, contrary to this claim when we use RoBERT we had slightly less accuracy for both all-class and 4-class classification models. The explanation for this is that the RoBERT model is ill-posed for this problem because we only have a 9,000 sample which is very less for training an LSTM based classification task. Therefore, we still conclude that the embedding based approach is definitely promising for this problem and worth exploring when we have data from a large number of users eg. Facebook, Twitter, etc.

## 6. Acknowledgment

I would like to thank Professor Zoran Kostic and course assistant of E6040, Zhengye Yang for giving us the opportunity to work on this project and providing us with the necessary support for high-performance computing machines on Google Cloud. I would like to thank Professor Yassine Benajiba for offering us the course on NLP in fall-2019 which inspired me to work in this area. I would also like to thank the authors for sharing this dataset with Amogh Mishra for this research. Also, I would like to thank my classmate Amogh Mishra for engaging in constructive discussion on the project idea. Lastly, I would thank my classmate Jake Stamell for reviewing the report and suggesting important edits.

## 7. References

[1] Github Repository[2][3] (samrat-halder)
[2] Gjurkovic, M., Snajder, J.: Reddit: A gold mine for personality prediction. In: PEOPLES@NAACL-HTL. pp. 87–97. Association for Computational Linguistics (2018)
[3] Pappagari, Raghavendra et al. "Hierarchical Transformers for Long Document Classification." *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (2019): 838-844.
[4] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pretraining of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[2] https://github.com/samrat-halder/personality-detection-with-BERT-RoBERT
[3] some of the source codes were imported from Google Research's BERT repository with necessary modifications