CS-1390 Final Project

Aarush Kumbhakern

December 3, 2024

Abstract

Author prrofiling infers personality and demographic traits from a sample of a document produced by an author. In this paper, we employ NLP techniques including TF-IDF vectorization, and GloVe/BERT word embedding to extract features from sample texts, and train a simple MLP to predict the Big Five personality traits from a given text sample.

1 Background

Author profiling is the process of inferring demographic traits of a text's author such as age, gender, geography, and personality traits. Author profiling finds its applications in the fields of forensics and business intelligence, eg. cyber-criminal identification, and marketing intelligence, and can provide valuable information about a population/set in a minimally invasive way. Author profiling finds an edge over author attribution by virtue of the fact that the training corpora need not contain documents by the author for inference to be made possible.

1.1 Personality Traits

Research around author profiling focuses primarily on inferring a small, fixed set of personality traits - the big five[1]:

1. **Openness to experience** (inventive/curious vs. consistent/cautious)
2. **Neuroticism** (sensitive/nervous vs. resilient/confident)
3. **Conscientiousness** (efficient/organized vs. extravagant/careless)
4. **Agreeableness** (friendly/compassionate vs. critical/judgmental)
5. **Extraversion** (outgoing/energetic vs. solitary/reserved)

[1] https://journals.sagepub.com/doi/abs/10.1177/0146167202289008

2 Goals

The aim of this project is to employ Natural Language Processing methods to extract relevant features from text samples, such that a machine learning model may be trained to infer the Big Five personality traits thereof. Existing attempts in this field largely evaluate the sample repeatedly for individual traits[2], but in this project I will attempt to replicate their accuracies with a multi-label classification approach using a simple MLP.

3 Data Collection

This project uses two widely-used existing datasets tailored for this particular problem:

1. **Stream-of-Consciousness Big Five-Labelled Essays:** a large dataset of 2400 stream-of-consciousness texts labelled with personality, produced by Pennebaker & King 1999 and used by Mairesse et al. 2007.[3]
2. **The NRC Emotion-Word Lexicon:** a mapping of large set of common words to emotional classifiers (anger, anticipation, disgust, fear, joy, negative, positive, sadness, surprise, trust, charged)[4]

4 Data Preprocessing

The essays dataset was first mildly preprocessed. The text column for each entry in the database was tokenized by sentence, following which any sentences that contained no words from the emotion-word lexicon dataset were dropped. This was done in an attempt to simplify the corpus and retain only data that was likely to contribute to an inference of personality.

5 Feature Extraction

In this section, we discuss the process of extracting features from the essay dataset. The methods employed fall all under the category of natural language processing, and attempt to both preserve context and extract meaning simultaneously.

[2] https://sentic.net/deep-learning-based-personality-detection.pdf
[3] http://web.archive.org/web/20160519045708/http://mypersonality.org/wiki/doku.php?id=wcpr13
[4] https://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm

5.1 TF-IDF Vectorization

Term Frequency-Inverse Document Frequency (TF-IDF) vectorization[5] can be split into two parts:

- **Term Frequency (TF):** measures how often a term t appears in a document d:
$$TF(t,d) = \frac{\text{No. occurrences of } t \text{ in } d}{\text{Total no. of terms in } d}$$
- **Inverse Document Frequency (IDF):** measures the importance of a term across all documents in the corpus c:
$$IDF(t,c) = \frac{\text{Total no. of documents in } c}{\text{No. of documents containing } t}$$

This ultimately gives us:

$$TF - IDF(t,d,c) = TF(t,d) \cdot IDF(t,c)$$

The benefit of TF-IDF over more straightforward methods such as Bag-of-Words (representing a document as a vector of occurrences of each word in the corpus) is its ability to capture the importance of terms specific to individual documents while diminishing the impact of ubiquitous terms.

5.2 Sentiment Vector

This feature attempts to naively quantify the overall sentiment of a text. It does so by simply tokenizing a text by words, and then computing the normalized sum of the sentiment vectors (from the Emotion-Word Lexicon dataset) of each token. This features hopes to contribute to the feature set by building on a possible mapping between Big Five traits and the expression of specific emotions.

5.3 Word Embeddings

Word embeddings are expressed as numerical representations of words in a vector space, and attempt to capture the semantic/syntactic relationship between words (words with similar meanings have vectors that are closer together). To be more precise, this definition applies to prediction-based embeddings - frequency-based embeddings operate on a different principle (an example of which is TF-IDF)[6]. for example:

$$\text{king} : [0.9, 0.7, 0.8]$$
$$\text{queen} : [0.85, 0.75, 0.8]$$
$$\text{apple} : [0.2, 0.1, 0.3]$$

[5] https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html
[6] https://www.ibm.com/topics/word-embeddings

Cosine Similarity:

$$Sim(\text{king}, \text{queen}) = \frac{0.9 \cdot 0.85 + 0.7 \cdot 0.75 + 0.8 \cdot 0.8}{\sqrt{0.9^2 + 0.7^2 + 0.8^2} \cdot \sqrt{0.85^2 + 0.75^2 + 0.8^2}} \approx 0.99$$

$$Sim(\text{king}, \text{apple}) = \frac{0.9 \cdot 0.2 + 0.7 \cdot 0.1 + 0.8 \cdot 0.3}{\sqrt{0.9^2 + 0.7^2 + 0.8^2} \cdot \sqrt{0.2^2 + 0.1^2 + 0.3^2}} \approx 0.45$$

In application, there exist multiple pre-trained embedding-focused language models, of which this paper employs two.

5.3.1 GloVe

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space[7].

GloVe provides pretrained word vectors in a choice of multiple tokens/vector dimensions - this paper employs the 6B token/200 dimension model.

5.3.2 BERT

Bert is a bidirectional transformer pretrained using a combination of masked language modeling objective and next sentence prediction on a large corpus comprising the Toronto Book Corpus and Wikipedia[8]. BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers[9].

The primary difference between BERT and GloVe lies in the fact that GloVe is context insensitive in the application training corpus (static, precomputed embeddings), whereas BERT produces dynamic embeddings that adapt to the context of a specific sentence.

Glove requires the context of a larger corpus to provide embeddings for a document in the corpus. however, due to computational limitations, this paper adopted a batch-processing approach and allowed each embedding a context of 31 other documents.

6 Principal Component Analysis

With the introduction of word embeddings into the feature set, the magnitude of the dimension of each feature vector ranges from 10^2 to 10^3, depending on

[7] https://nlp.stanford.edu/projects/glove/

[8] https://huggingface.co/docs/transformers/en/model_doc/bert

[9] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of NAACL-HLT 2019, 1-20. Association for Computational Linguistics.

how many features are selected for the final feature set. This number grows to be close to, if not exceed the number of training samples themselves, and therefore to avoid the problem of overfitting, PCA was employed to reduce the dimension of the feature vectors to 128.

7 Machine Learning Approaches

Having extracted a suitable set of features, we turn our attention to machine learning approaches for inferring author personality traits from sid features. As mentioned previously, we have chosen in this paper to view the problem as a multi-label classification problem, where each text is mapped to five labels. This suggests that the optimal strategy to employ would be a fully-connected feed-forward neural network, or multilayer perceptron (MLP). This paper employs the Tensorflow suite to implement this method.

7.1 Model and Hyperparameter Tuning

As a result of the nature of neural networks, the large majority of model and hyperparameter tuning is an exercise in empirical observation and experimentation. Having conducted this over a series of values, the optimal configuration has been found to be as follows:

1. **Selected features:** TF-IDF, Sentiment Vector, GloVe Embedding
2. **Principal component count:** 256
3. **Hidden layers:** 5
4. **Activation function:** ReLU, sigmoid at output
5. **Neuron droput fraction:** 0.41
6. **L2 regularization strength:** 0.001

8 Results

The model was evaluated on the basis of label-wise accuracy and exact match accuracy. Label-wise accuracy averaged ≈ 0.55 which is on par with the results of most papers training individual-attribute inference. Exact-match accuracy averaged ≈ 0.075, reaching a maximum of 0.098.

We found, surprisingly, that a feature set with GloVe embeddings showed better results than one with BERT. This is likely a result of the constrained context size.

9 Conclusion

We conclude that the process followed here is not sufficient to achieve a satisfactory exact-match accuracy. This is not to say that the method showed no promise, albeit minimal - the model outperforms a blind guess by a factor of 3 at its best. The fault of the unsatisfactory results may also lie in the shortcomings of the test machine, and a larger context size may allow a feature set containing BERT embeddings to perform better. A larger dataset with shorter texts may allow for a more contemporary application of the model, and finally, alternative machine learning approaches such as one vs. many SVM and tree-based models (eg. XGBoost) would likely show significant improvement.