Essays are a Fickle Thing

Lucija Arambašić, Miroslav Bićanić, Frano Rajič

University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
{Lucija.Arambasic,Miroslav.Bicanic,Frano.Rajic}@fer.hr

Abstract

Automatic classification of a person's personality based on a piece of text written by that person is an inherently difficult task, but its difficulty could increase depending on the dataset used. In this work, we explore the classification performance of many different machine learning models with various feature combinations when the dataset consists of stream-of-consciousness essays written by students. Despite achieving very good performance, we argue that such a dataset may not be ideal for personality trait classification.

1. Introduction

Personality is defined as the pattern of thoughts, feelings and behaviour specific to each individual. Personality is often characterized by using different personality traits. One of the most famous ways of describing personality is the Big Five model, which defines five fundamental personality traits: extroversion (EXT), neuroticism (NEU), agreeableness (AGR), conscientiousness (CON) and openness to experience (OPN).

Automatic personality assessment from text has numerous applications, ranging from job interviews to author profiling. Because of this, there was a need for an appropriate dataset. Social network platforms are a popular source for such data since posts on them often contain people's opinions and feelings. Park et al. (2014) built such a dataset using Facebook posts, while Gjurković et al. (2020) built their dataset using posts from Reddit. One popular dataset not obtained from social media is the *essays* dataset, in which stream-of-consciousness essays are matched with Big Five gold labeling of their authors.

In this paper, we analyze our attempt to perform personality trait classification of essay authors on the *essays* dataset. We employ several machine learning techniques, including both static and sequence-based models, utilizing various standard and hand-crafted features. Additionally, we experiment with data augmentation. Finally, we bring into question the quality and applicability of the *essays* dataset for personality trait classification.

2. Related Work

Some of the previous work regarding personality trait classification from text was done by the authors of the *essays* dataset. They extracted Linguistic Inquiry and Word Count (LIWC) features and used them to determine the link between the written essay and the authors Big Five personality traits (Pennebaker and King, 1999).

Similarly to us, Pizzolli and Strapparava (2019) trained their models on the *essays* dataset, but then used the model to classify the personality traits of characters in Shakespearean plays. It is important to note that this task does not have gold labels, so the performance was evaluated manually and subjectively.

More recent work on personality trait classification using the *essays* dataset is done in (Majumder et al., 2017). Their method consisted of data preprocessing and filtering, feature extraction, and finally classification. Similarly to us, they used *word2vec* embeddings to get vector representations of words and essays. Unlike us, they used a deep CNN for classification. To our knowledge, the results they achieved (displayed in Table 3) represent state-of-the-art performance on this dataset.

3. Essays Dataset

As we mentioned earlier, we used the *essays* dataset, which is the result of research by Pennebaker and King (1999). It consists of 2467 *stream-of-consciousness* essays written by 34 psychology students between 1993 and 1996 (Tighe et al., 2016). Each essay is accompanied with five binary labels, one for each of the Big5 personality traits. Specifically, each entry in the dataset is in the format `author_id, essay, ext, neu, agr, con, opn`, where the binary labels for traits are represented with `y` or `n`.

Trait distribution in the dataset is shown in Table 1. The values in the first row are the absolute numbers of essays with a positive label for the trait in that column, while the ratio of such essays in the dataset is given in the second row. A more detailed statistical analysis of the traits can be found in (Pennebaker and King, 1999) and (Mairesse et al., 2007).

The essays themselves come in a lot of shapes and sizes: the minimal number of words and sentences in an essay is 39 and 1, while the maximal numbers are 324 and 2964, respectively. The average number of words and sentences is 742.4 and 48.6. While a diverse dataset is generally desirable, such drastic differences in length can pose a problem, especially for models such as LSTMs. Furthermore, many examples are incomplete in some way: the essay with 39 words actually abruptly ends mid-word, while the essay with a single sentence doesn't contain any interpunction.

4. Our Approach

Our goal was to design a system that would facilitate experimentation so that we could easily try different models using different features. The models we implemented can roughly be categorized into three groups: (1) true baselines,

Table 1: Trait distribution in the *essays* dataset.

EXT	NEU	AGR	CON	OPN
1276	1233	1310	1253	1271
51.7%	49.9%	53.1%	50.79%	51.52%

(2) static models, (3) and sequence-based models. Each group has a different set of features at its disposal. All models simplify the multilabel classification task by separating it into five independent binary classification problems.

4.1. Data Preprocessing

As part of the preprocessing step, we discard the `author_id` field and convert the `y/n` labels into numerical `1/0` labels. Then we perform sentence- and word-level tokenization on lowercased essays. At this point, every example contains three views of an essay: (1) a raw essay, (2) a list of sentences in the essay, (3) a list of words in the essay. Tokenization is performed using the *punkt* tokenizer from the NLTK framework. Finally, the dataset is split into standard train/valid/test subsets with a 60/20/20 ratio, respectively.

4.2. Feature Extractors

A feature extractor is in charge of converting a dataset of essays into a dataset of fixed-size vector representations, to be used by static models. Any and all extractor parameters are initialized based on the train split of the dataset to avoid data leakage. Initialized extractors are then used to extract features from all three splits of the dataset.

To make extraction flexible and robust, the extraction method receives all three views of the essay. For example, our custom capitalization extractor requires raw essays in order to detect capitalized letters, while a *word2vec* extractor requires a list of words in the essay. All the implemented extractors are shown in Table 2.

4.3. True Baselines

True baselines consist of two rudimentary models which don't rely on any of the features, nor the essays themselves. The first baseline is a dataset-agnostic random classifier (RC), and the second one is a most common class classifier (MCC) which classifies all examples with the majority label for each of the traits, based on the distribution in the train split.

4.4. Static Models

Static models refer to classifiers whose input is a fixed-size vector representation of an essay. This group consists of three classifiers: (1) a fully connected neural network (FC), (2) a support vector machine (SVM), (3) and a naive Bayes SVM (NBSVM).

We implemented the FC model using the PyTorch framework, and the SVM implementation is taken from scikit-learn. Both of these models utilize the features generated by feature extractors. On the other hand, we used a pre-built NBSVM implementation[1] (Wang and Manning, 2012) designed to work exclusively with bag-of-words features.

4.5. Sequence-Based Models

Sequence-based models refer to models which take into account the sequential nature of essays - each essay is a sequence of words or sentences. One of the most popular models for sequential data is an LSTM cell, which we implemented using PyTorch.

We tried training LSTMs with sequences of words as well as sentences. In both cases, the elements of the sequence first had to be converted to their corresponding vector representations. Since the inputs to sequence-based models do not have a fixed dimension, we couldn't use the feature extractors as described in Section 4.2..

Instead, when working with word sequences, we used Google's 300-dimensional embeddings obtained on the Google News corpus. The embeddings were loaded and processed using the *gensim*[2] library. When working with sentence sequences, we used two different pre-trained word embeddings with a larger dimensionality (Pagliardini et al., 2018): 600-dimensional *sent2vec-wiki-unigrams* obtained on English Wikipedia and 700-dimensional *sent2vec-toronto-books* obtained on BookCorpus. The word embeddings were combined into sentence embeddings using the *epfml/sent2vec*[3] library.

4.6. Data Augmentation

It is known that LSTMs (and recurrent neural networks in general) struggle with sequences longer than a few dozen words. As stated in Section 3., an average essay contains over 700 words and around 50 sentences. This means that even the average essay is far too long to be adequately processed by an LSTM.

An additional problem for LSTMs is the great discrepancy in essay lengths. Namely, when the dataset is being batched, every instance in the batch is zero-padded to match the length of the longest instance. A big difference in length can result in some examples having more nil-vectors than actual useful information.

To address these issues, as well as the relatively small size of the dataset, we split each essay into several chunks, with each chunk having a minimum of C words (C is a hyperparameter). Splitting was implemented to only occur on the position of an interpunction symbol (`.,!?`). Each chunk was assigned the same labels as the essay from which it was taken. This resulted in a dataset of more than 40 thousand examples, the vast majority of which are similar in length.

Because the essays are already scarce with emotion, many of the examples generated by chunking were completely void of emotionally charged words. Furthermore, the fewer words there are in a chunk, the greater the chance that the dataset already contains a very similar chunk. This can lead to contradictory examples if the two chunks come from essays with different trait labels, thus making the training process even more difficult.

[1] `https://github.com/mesnilgr/nbsvm/`
[2] `https://radimrehurek.com/gensim/`
[3] `https://github.com/epfml/sent2vec`

Table 2: Feature extractors used for static models.

Extractor	Semantics	Vector dimension
Capitalization	Number of uppercase letters; normalized by sentence count	1
WordCount	Number of words; normalized using mean and SD of word counts in the train split	1
Interpunction	Number of periods, exclamation and question marks; normalized by sentence count	3
RepeatingLetters	Number of letters that were repeated 3 or more times	1
TF-IDF	TF-IDF vectors; vocabulary V built only on train set	$\|V\|$
W2V	Averaged vector representations of words in the essay	300
S2V	Averaged vector representations of sentences in the essay	600 - 700

A possible solution for the described problem was found in (Majumder et al., 2017): removing every emotionally void sentence from every essay. A sentence is considered emotionally void if it has no emotionally charged words. The emotional charge of a word is determined by comparing the word against a known set of emotionally charged words - in this case the NRC Emotion Lexicon[4] (Mohammad and Kiritchenko, 2015). We expanded on this idea and implemented two different variants of filtering: (1) removing sentences from raw essays, and then chunking the essays; (2) chunking the essays, and then removing emotionally void chunks. The second approach is motivated by the desire to remove useless and problematic examples from the generated dataset. Emotional dropping improved the performance of LSTM cells, with the second variant bringing greater benefits.

5. Results

As previously stated, we performed all model training and evaluation on the *essays* dataset. The evaluation results are shown in Table 3 and Table 4. The tables show accuracies and F1 measures of the state-of-the-art model from Majumder et al. (2017), the MCC baseline, and most of the models with which we experimented.

We ran various combinations of features and models, but displayed only the best ones. Each of the models was independently trained 10 times. All the metrics from those 10 runs were averaged and their standard deviation was calculated. It is important to note that accuracy is an acceptable performance metric on this dataset because the traits are balanced, as we have shown in Table 1. Nonetheless, we also show the achieved F1 scores and their standard deviations. To enable reproducibility we split the dataset once before all the runs, and we set up the same random number generator seed for all our experiments.

It can be seen that our NBSVM models achieve higher accuracy than the state-of-the-art on openness and neuroticism, but it should be noted that Majumder et al. (2017) evaluated their results using cross-validation, and we only split the dataset once, creating train/valid/test subsets.

6. Dataset Commentary

Human personality is very complex in its nature and determining the Big Five traits solely from text is a very challenging task. We feel that the *essays* dataset makes the problem even harder, primarily due to the stream-of-consciousness nature of the essays. Such essays exhibit the thoughts of the person writing them, but such thoughts may not reveal enough to determine the author's personality traits, as they often lack emotional expression. This is backed by the fact that dropping emotionally neutral sentences improved performance.

Furthermore, the author's thoughts are often influenced by their surroundings. If the author is not trained in controlling and structuring their thoughts for the essay, which is likely the case with most students that wrote them, there is a lot of noise in the text. For example, some essays are just describing the room the author was in at the time of writing. Because of this, noise some models have difficulty grasping the essence of the author's traits. Another downside of the noise is that the essays become unnecessarily long, which is a huge problem for models like the LSTM cell. We addressed the problem of lengthy essays by using essay chunking, previously described in Section 4.6.

In contrast to essays, posts from social media like Twitter or Facebook offer a deeper insight into people's perspectives and attitudes. The nature of social media posts is such that authors often express their opinions in an emotional manner, with emotions ranging from anger and frustration all the way to joy and excitement. Moreover, such posts are often shorter, and in the case of Twitter they even have an upper bound. This means the information present in them could be acquired faster and easier. For these reasons, we believe that social media datasets are better suited for the task of personality trait classification.

7. Conclusion

The *essays* dataset proved to be a challenging dataset for the task of personality trait classification. We managed to obtain very good results with our models, with some coming close to state-of-the-art models. In spite of that, we still believe that this dataset has shortcomings and makes the task more difficult than it could have been.

In future work, we would use cross-validation to better evaluate our models, and to be able to directly compare them to the state-of-the-art. Furthermore, we would explore the effects of emotionally charged words in essays to a greater extent. Finally, it could be informative to compare the performance of one model on several different datasets.

[4] http://saifmohammad.com/WebPages/

Table 3: Accuracies of models on each of the traits. † NBSVM used uni+bi+tri+quadgrams. ‡ NBSVM used uni+bigrams. * NBSVM used uni+bi+trigrams.

Model	OPN [$\% \pm \sigma$]	CON [$\% \pm \sigma$]	EXT [$\% \pm \sigma$]	AGR [$\% \pm \sigma$]	NEU [$\% \pm \sigma$]	AVG [$\% \pm \sigma$]
(Majumder et al., 2017)	57.30	**62.68**	58.09	**56.71**	59.38	**58.83**
NBSVM	**63.08**†	57.61†	58.01†	52.94‡	**60.45***	58.42
MCC	52.13	51.12	54.36	52.13	49.90	51.93
SVM-CUSTOM	51.32	54.77	50.10	53.55	52.54	52.45
SVM-BOW	52.13	51.12	54.36	52.13	49.90	51.93
SVM-W2V	52.13	51.12	54.36	52.13	49.90	51.93
SVM-S2V	52.13	51.12	54.36	52.13	50.51	52.05
SVM-CUSTOM,BOW,W2V	60.45	57.20	**58.42**	53.14	58.62	57.57
LSTM	51.54 ± 1.49	49.43 ± 0.45	53.08 ± 1.42	52.41 ± 0.94	51.32 ± 0.98	51.56 ± 1.06
BiLSTM	51.46 ± 1.52	49.13 ± 0.91	52.27 ± 1.66	52.03 ± 0.45	50.20 ± 1.16	51.02 ± 1.14
LSTM-CHUNK	57.93 ± 1.64	52.37 ± 2.55	51.40 ± 3.27	52.11 ± 0.06	50.20 ± 0.10	52.80 ± 1.52
LSTM-CHUNK+EMOv1	58.48 ± 2.26	52.84 ± 1.57	51.99 ± 1.88	52.09 ± 0.12	51.78 ± 3.28	53.44 ± 1.82
LSTM-CHUNK+EMOv2	59.59 ± 1.59	51.83 ± 0.85	50.97 ± 2.25	52.27 ± 0.37	59.43 ± 2.72	54.82 ± 1.56
BiLSTM-CHUNK+EMOv2	58.48 ± 2.28	51.54 ± 0.96	51.30 ± 2.50	52.11 ± 0.06	52.52 ± 1.98	53.19 ± 1.56
FC-CUSTOM	51.72 ± 1.37	51.87 ± 1.35	50.63 ± 0.93	52.37 ± 0.70	52.80 ± 1.05	51.88 ± 1.08
FC-BOW	60.93 ± 0.39	58.42 ± 0.84	54.24 ± 0.44	49.68 ± 0.45	60.00 ± 0.39	56.65 ± 0.50
FC-W2V	62.23 ± 0.56	58.26 ± 1.01	52.86 ± 1.38	51.60 ± 0.23	57.79 ± 0.33	56.55 ± 0.70
FC-S2V	60.89 ± 0.72	58.44 ± 0.60	55.46 ± 0.86	52.31 ± 0.41	57.28 ± 0.59	56.88 ± 0.64
FC-CUSTOM,BOW,W2V	62.66 ± 0.74	58.62 ± 0.40	54.58 ± 0.39	52.27 ± 0.31	59.45 ± 0.40	57.52 ± 0.45

Table 4: F1 scores of models on each of the traits. † NBSVM used uni+bi+tri+quadgrams. ‡ NBSVM used uni+bigrams. * NBSVM used uni+bi+trigrams.

Model	OPN [$\% \pm \sigma$]	CON [$\% \pm \sigma$]	EXT [$\% \pm \sigma$]	AGR [$\% \pm \sigma$]	NEU [$\% \pm \sigma$]	AVG [$\% \pm \sigma$]
(Majumder et al., 2017)	n/a	n/a	n/a	n/a	n/a	n/a
NBSVM	68.07	61.65	64.74	60.81	62.57	63.57
MCC	**68.53**	67.65	**70.43**	**68.53**	nan	**68.79**
SVM-CUSTOM	58.76	66.17	50.40	67.39	44.29	57.40
SVM-BOW	**68.53**	67.65	**70.43**	**68.53**	nan	**68.79**
SVM-W2V	**68.53**	67.65	**70.43**	**68.53**	nan	**68.79**
SVM-S2V	**68.53**	67.65	**70.43**	**68.53**	7.58	56.55
SVM-CUSTOM+BOW+W2V	62.14	58.22	61.54	57.46	59.20	59.71
LSTM	58.89 ± 11.62	62.49 ± 1.91	58.95 ± 10.51	64.23 ± 1.34	52.72 ± 7.39	59.46 ± 6.55
BiLSTM	59.14 ± 9.93	61.09 ± 1.75	55.48 ± 13.24	64.90 ± 3.62	45.98 ± 14.81	57.31 ± 8.67
LSTM+CHUNK	59.57 ± 7.11	$nan \pm nan$	53.60 ± 11.75	68.52 ± 0.05	$\mathbf{66.80 \pm 0.05}$	62.12 ± 4.74
LSTM+CHUNK+EMOv1	62.06 ± 5.86	65.80 ± 3.01	56.02 ± 6.94	68.48 ± 0.09	64.57 ± 5.37	63.39 ± 4.25
LSTM+CHUNK+EMOv2	61.91 ± 6.89	$\mathbf{67.66 \pm 0.17}$	54.87 ± 9.07	68.38 ± 0.19	59.24 ± 9.54	62.41 ± 5.17
BiLSTM-CHUNK+EMOv2	58.86 ± 12.25	66.29 ± 2.83	54.98 ± 13.55	68.52 ± 0.05	55.41 ± 21.54	60.81 ± 10.04
FC-CUSTOM	58.34 ± 1.92	60.48 ± 3.93	56.30 ± 1.96	68.01 ± 0.75	47.82 ± 7.19	58.19 ± 3.15
FC-BOW	62.40 ± 0.77	60.61 ± 1.53	58.09 ± 0.85	57.08 ± 0.61	60.38 ± 1.24	59.71 ± 1.00
FC-W2V	63.44 ± 2.27	61.56 ± 1.23	53.70 ± 8.01	57.98 ± 0.55	53.30 ± 1.63	58.00 ± 2.74
FC-S2V	62.33 ± 1.06	60.64 ± 2.13	59.67 ± 2.87	57.20 ± 1.47	57.68 ± 2.08	59.50 ± 1.92
FC-CUSTOM+BOW+W2V	63.62 ± 0.71	59.81 ± 0.50	60.15 ± 0.88	66.77 ± 0.47	57.73 ± 0.80	61.61 ± 0.67

References

Matej Gjurković, Mladen Karan, Iva Vukojević, Mihaela Bošnjak, and Jan Šnajder. 2020. Pandora talks: Personality and demographics on reddit.

Francois Mairesse, Marilyn Walker, Matthias Mehl, and Roger Moore. 2007. Using linguistic cues for the automatic recognition of personality in conversation and text. *J. Artif. Intell. Res. (JAIR)*, 30:457–500, 09.

Navonil Majumder, Soujanya Poria, Alexander Gelbukh, and Erik Cambria. 2017. Deep learning-based document modeling for personality detection from text. *IEEE Intelligent Systems*, 32:74–79, 03.

Saif M Mohammad and Svetlana Kiritchenko. 2015. Using hashtags to capture fine emotion categories from tweets. *Computational Intelligence*, 31(2):301–326.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, Louisiana, June. Association for Computational Linguistics.

Gregory Park, H. Schwartz, Johannes Eichstaedt, Margaret Kern, Michal Kosinski, David Stillwell, Lyle Ungar, and Martin Seligman. 2014. Automatic personality assessment through social media language. *Journal of personality and social psychology*, 108, 11.

James Pennebaker and Laura King. 1999. Linguistic styles: Language use as an individual difference. *Journal of personality and social psychology*, 77:1296–312, 01.

Daniele Pizzolli and Carlo Strapparava. 2019. Personality traits recognition in literary texts. In *Proceedings of the Second Workshop on Storytelling*, pages 107–111, Florence, Italy, August. Association for Computational Linguistics.

Edward P. Tighe, Jennifer C. Ureta, Bernard Andrei L. Pollo, Charibeth K. Cheng, and R. D. Bulos. 2016. Personality trait classification of essays with the application of feature reduction. In *SAAIP@IJCAI*.

Sida Wang and Christopher Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, Jeju Island, Korea, July. Association for Computational Linguistics.