Advanced Natural Language Processing: Style Analysis by Sentence Embedding

Abel De Wit, Rik Dijkstra, Max Knappe

Abstract—**Writing style has a huge influence on people. In our modern world where we receive hundreds of messages in any form each day, style is a key element to get recognized. This paper focuses on finding measures to compare the style of twitter post by different celebrities of our time based on the concept of sentence embedding.**

I. Introduction

Like many other Natural Language Processing (NLP) topics, writing style is not trivial for a computer to dissect. In recent years, techniques for NLP have numerously increased; the recent applications would be unthinkable some decades ago. That said, it still lags behind fields like computer vision because it cannot be represented in the same way pictures are. Furthermore, language is not absolute. It changes over time and differs from culture to culture and region to region.

The fact that NLP is less straightforward to work with does not entail that the uses are any less practical or exciting. In some sense, what makes it so challenging to dissect also makes the applications so valuable. Communicating with humans through their language brings along a broad range of everyday applications. From generation tasks to automate content creation to summarisation tasks to make vast amounts of data accessible.

Style Transfer is amongst those tasks as well. Much style-related research is done on sentiment, where positive and negative writing is conceptualised into two styles. However, the concept of author style appealed to us in particular. The idea of capturing the personal touch of any author can have inspiring or even humorous applications. The idea to build a style-transfer model quickly emerged. The user could enter a target text and convert it to the writing style of a particular author. However, after conducting some more in-depth research, we had to revise this plan. Style-transfer, with non-parallel data in particular, is not a trivial task. We spoke to Ms Liu about the project and its progress, and keeping in mind the work-load permitted for this project, we decided in our conversation to shift our focus to the analysis of style rather than its transfer.

The analysis of sentence embeddings can be achieved with less complicated models, as identification tasks are often more straightforward than generation tasks, especially in the field of NLP. Still, it is a task that is somewhat aligned with our interest in style-related topics. It is not a given that style can be represented in sentence embeddings, therefore we will adopt a broader focus. We will also be researching the generation of pseudo-parallel data, from non-parallel datasets. Related to this problem is topic clustering, which we could use to more efficiently find pseudo-parallel [1] entries.

II. Research Questions

There are a number of research questions that we intend to answer.

1) **Is it possible to generate pseudo-parallel data based on the clustering of sentence embeddings of different authors?**
   Pseudo-parallel data is data which has a high similarity, but is not definitively parallel. It could be used for training models for which no parallel data is available.
2) **Is there a relation between the difference in pseudo-parallel sentences and the difference in style?** This could be achieved by analysing the difference between two pseudo-parallel data entries.

III. Related Work

A. *Stlye and content extraction*

The paper "Disentangled Representation Learning for Non-Parallel Text Style Transfer" proposes a neural network structure for extracting both content and style from text. They provide two approaches, using an *autoencoder* to encode text into the latent space representation and a *recurrent neural network* to decode the text word by word. Alternatively, they propose the usage of a *variational autoencoder*. They describe loss functions each for style and content to train their model. After validating that the approaches both satisfy the content and style extraction to a sufficient level, they demonstrate their work by applying different styles to the extracted content. [2]

B. *Clustering*

Reimers and Gurevych propose multiple clustering algorithms to be used for natural language processing, including *k-Means*, *agglomerative clustering* and *fast clustering*.While *k-Means* is a basic clustering algorithm, based on the k nearest neighbours, *agglomerative Clustering* extends this method with a threshold to merge small clusters together. Therefore, *agglomerative clustering* is a useful method if the number of clusters is unknown. However, *agglomerative clusterin* performs slow on large datasets. To solve this, *fast clustering* is designed to cluster large datasets in a short period of time. [3]

C. SBERT

The paper also provides extentions of *BERT (Bidirectional Encoder Representations from Transformers)* and its optimized version *RoBERTa* called *SBERT* and *SRoBERTa*, each for sentences. Both use multiple *BERT / RoBERTa* networks connected in either a so called *siamese* or *triplet network structure*. The respective outputs of all networks run through a pooling stage and get connected with a objective function. Experiments demonstrate that *SBERT* outperforms other state-of-the-art sentance embedding methods such as *InferSent* and *Universal Sentence Encoder* and extends the application of *BERT* to sentence level. [3]

D. Style Embedding

At the Thirty-Second AAAI Conference on Artificial Intelligence a paper was published describing a style-embedding model, where text style is embedded into a vector representation via a *autoencoder seq2seq*-model. This method allows networks to export and exchange styles between sessions. [4]

A possible application field for style embedding is plagiarism detection. While simply copy pasting information without referencing them can easily be recognized, changes in writing style may also indicate plagiarism, which may not be obvious to the reader. Several papers address this topic, providing models to analyze documents and recognize different styles which may indicate plagiarism. [5]

IV. Methods

A. Definitions

Before we elaborate on our approach, it is essential to define a number of concepts. First of all, *"style"* is a very ambiguous concept. In many papers, it is related to the sentiment of a sentence, but any author could write in both positive and negative sentiment. Therefore, we define ***"style"*** as a particular use of language and vocabulary to convey a message. That said, the reader should be able to differentiate between two identical messages with different styles.

We also talk about *"content"* and *"topic"*, which imply something very similar. Therefore, we define ***"content"*** as the message conveyed and ***"topic"*** quite literally as the subject matter: e.g. natural disaster or cars.

B. Sentence embedding

Our approach is based on sentence embedding. There are several ways to do this, but the state of the art algorithm at this moment is the *Bidirectional Encoder Representations from Transformers*, or BERT. There are multiple variations to BERT, of which we chose a RoBERTa based pre-trained model from the *sentence-transformers* library [3]. The RoBERTa model is a modification of BERT and modifies its key hyper-parameters. Besides some changes to the model, it is trained on a much larger amount of data, for a longer amount of time. Therefore, it is said to generalise better to downstream tasks. [6]

C. Dimensionality reduction

To be able to analyze and cluster our data entries, it is required to reduce the dimensionality of their sentence embeddings. There are multiple methods suitable for this task, but we chose to implement *t-Distributed Stochastic Neighbor Embedding* (t-SNE) [7]. t-SNE is a non-linear dimensionality reduction technique that works particularly well on high dimensional data, like sentence embeddings. It uses two distributions, one of which measures pairwise similarities in the input dimension and the other in the output dimension. It minimizes the divergence between these two distributions to map the higher-dimensional data to a lower-dimensional space. In our case, we found that two-dimensional output was preferable over three-dimensional output because of better clarity in plots.

D. Topic Clustering

The topic clustering approach [8] used for our topic clusters can be subdivided into two parts.

Clustering of Sentence Embedding

The sentences are clustered based on their euclidean distance. First, several cluster centres are chosen, after which the algorithm clusters all sentence embedding within a certain euclidean distance. The algorithm defines these centres and the distance threshold automatically, based on k-medoids. K-medoids is a clustering algorithm much like k-means. It uses medoids to find clusters, which are entries of a dataset whose average dissimilarity is minimized to all other elements in a cluster, as illustrated in Figure 1.

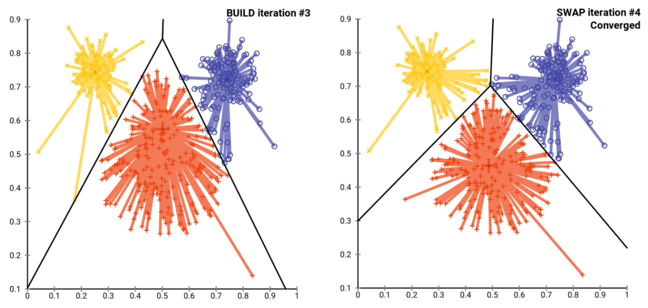

Fig. 1. K-medoids clustering a dataset

Topic Modeling

The topic modeling approach we follow is based on a simple *Bag Of Words* (BOW) model. We count the most frequently appearing words and order them as such. We remove any tags we added, explained in section V-A, and use *Part Of Speech tagging* (POS tagging) to exclude all non-nouns. The five remaining most used words are our topic. It is worthy to note there are much more advanced topic modelling techniques, however, this is a solid and easy alternative.

Another method to retrieve topics from tweets is the usage of *Latent Dirichlet Allocation* (LDA) model by Andrew Ng.

at al. The basic idea that a text is based on a random mixture of topics. By taking every word into account, a probabilistic model of the whole text is generated by using the dirichlet distribution, giving the LDA its name. The model outputs a k-dimensional vector of probabilities representing the distribution of topics in the text. [9]

The model achieves this by generating a user defined number k of topics. For each so called *Document*, which in our case is a tweet, a topic probability for each topic should be generated.

1) Assuming there are exactly k topics in the document, each word is assigned a topic by using the dirichlet distribution.
2) For each word in the document suppose its topic is wrong, but everything else is assigned correctly.
3) Re-assign the word to a new topic based on the topic distribution of the current document and the topics the word is already assigned to in all documents.
4) After repeating the last step for a number of times, the model is set up.

In contrast to BOW, the LDA model provides probability values for each word, while BOW is only capable of providing the most used words without any measure. Because of this, we dropped the BOW approach during the project, as our evaluation metric uses the probability values supplied by LDA.

E. Style embedding

The possibility of embedding style in a vector leads to many possibilities to compare styles in their vector form, although a specific style may not be recognized based on the sheer numbers. One interesting measure is the similarity of two styles retrieved via style embedding. A common measure for this is the cosine similarity, resulting in similarity values between minus one and one, in which minus one indicates exactly opposite, zero indicates decorrelation and one exactly the same.

$$cos(\theta) = \frac{A * B}{||A|| * ||B||} \tag{1}$$

While investigating similarities between style vectors, we thought about subtracting one style vector from another to get a remainder vector. This vector could be interpreted as a style difference. We discussed on how to investigate the meaning of this, as it could be interpreted as a style itself or just an element of a specific style. We want to try to address this issue in our experiments, but we're unsure if our experiments will verify this or even give evidence because of unfit data.

V. Data

A. Preprocessing

To get optimal results from our twitter datasets, the tweets have to be preprocessed to fit the models. To do that, we use the *Natural Language Toolkit* library (nltk). In our case, twitter username links, URLs and numbers have to be removed as they aren't a part of sentences. In addition to that, we remove stopwords and punctuation to focus more on style and content, as those elements do not contribute to neither to style nor to content. *nltk* itself defines a list of stopwords, which facilitates the process of preprocessing in our case.

B. Data sets

For our tests, we use six twitter data sets of 5000 randomly selected tweets for each twitter user we now want to introduce.

- Donald Trump, 45th president of the United States for the republican party, commonly seen as a person with a very unique style
- Barack Obama, 44th president of the United States for the democratic party
- Hillary Clinton, former first lady and secretary of defense of the United States, member of the democratic party
- Neil deGrasse Tyson, an astrophysicist known for keeping pluto from being referred as a planet
- Adam Savage, a special effects designer and producer known from movies like *Star Wars* and *The Matrix*
- Kim Kardashian, a social media personality known from the series *Keeping Up with the Kardashians*

These well known celebrities have more or less commonalities, making them an optimal mixture for experiments regarding topic clustering and style analysis.

VI. Experiments

A. Cluster and Topic size

To determine the optimal cluster size and amount of topics, an experiment was set up to test different values for both. By using the reduced dimensionality embeddings of our tweets to cluster the tweets using the K-Medoids technique, an attempt is made to cluster topics together. In order to measure how well a topic covers a cluster, the LDA model is used to predict a topic for both the collection of tweets in a cluster, and a prediction on each single tweet in a cluster. After the LDA model is trained on the tweets in its entirety, it can give a prediction about sentences or collections of those. The output of this prediction is a list of possible topics, and their corresponding certainties. This allows us to measure the correlation between the 'main' topic of a cluster, and the topics of the sentences that are part of it.

By adding the certainty of the correct predictions of a sentence, and normalizing that with respect to the chance of having the topic right, we are able to model a score for each cluster.

We sum the model's certainty on a sentence topic prediction and either add or subtract this to the overall cluster score. This way, a prediction with a high certainty will influence the score more, for better or for worse. The certainty is also multiplied with a fraction (1/num_topics), this is to normalize the score as predicting a topic for each sentence in a cluster gets a lot easier when there are very few topics, and it gets harder with more topics. In order to not punish the higher number of topics, this fraction will make sure that the score is balanced over the tested variables.

With this metric, experiments were run with both varying cluster sizes and a varying number of topics. Ideally a sufficiently large number of clusters is chosen, such that the embeddings are divided into small clusters that have more in common than embeddings in large clusters. However, a number of clusters that is too high, will result in clusters with a very small number of sentences, or as observed in our research, might even result in a cluster consisting of just one embedding. Something similar is true for the amount of topics: if there is just one topic, the metric will perform very well as there is just one topic to choose from, while the desired outcome will have a large amount of topics that are not too specific to only apply to margin cases of our corpus.

B. *Pseudo-parallel data*

To generate pseudo-parallel data, each tweet assigned to a cluster has to be compared to each tweet in the same cluster, using cosine similarity 1. We chose to save all tweets with a cosine similarity of 0.6 or higher are worth to be saved as pairs. Similarity values close to one are only found for tweets with almost exactly the same content and wording, tweets near those chosen border of 0.6 aren't that similar anymore, but share a broad topic. Similarity scores of 0.5 and less are not taken into account for pseudo-parallel data, as the difference to high and would impact the outcomes of further research.

C. *Style embedding*

To further analyze the style of the tweets, we take a look at their style embeddings described as before in section IV-E. All pseudo-parallel data we found earlier is compared via subtracting the sentence embeddings of the respective vector from its counterpart, which should only leave the style in the remaining vector, as the data used can be seen as data with the same content. For each cluster and all combinations of pseudo-parallel data in this cluster, we calculate the style vector as described before. To put the style similarity in numbers, we calculate the average cosine similarity 1 for each pseudo-parallel data pair in the cluster.

VII. Results

A. *Cluster and Topic size*

As we can see in Figure 2 and Figure 3 the score of correctly identified topics per cluster and topic size increases with larger numbers. In Figure 2 we can see that until one hundred clusters, the score does not improve much with an increased number of topics. This result is due to the fact that a small amount of clusters will have a small amount of general topics, while their underlying sentences have the ability to be classified as many more topics. Only when the number of clusters is sufficient to capture the general topic of the sentences that are part of each cluster, an increase in the score can be seen.

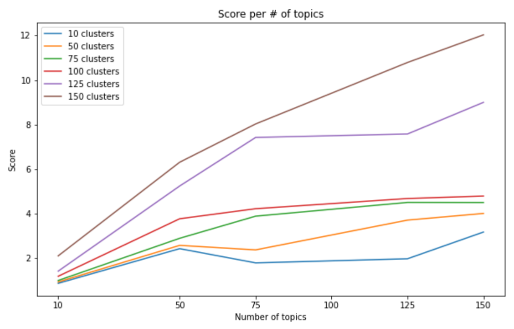

Fig. 2. Score for each cluster size with varying topic size

We see the same occurrence in Figure 3, where the lines with different topic sizes all perform almost the same until after the cluster size gets larger than 100.

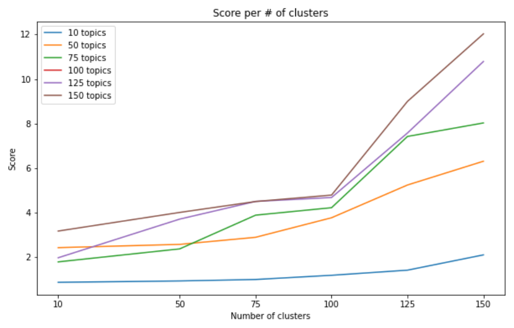

Fig. 3. Score for each topic size with varying cluster size

These results are as expected when the experiment was set up. Too little clusters or too little topics will neither generalise enough, nor create clusters specific enough to create topics that apply well to the whole cluster.

B. *Pseudo-parallel data*

Analyzing the pseudo-parallel data pairs, several interesting observations can be made. When having a look at figure 4, the amount of tweets that our experiment defines as pseudo-parallel data is shown for each pair of authors and color-coded based on their cosine-similarity.

The second highest amount of high level pseudo-parallel data (0.9 -1) can be found for Barack Obama and Hillary Clinton, both politicians for the democratic party. The first place in this category however goes to Donald Trump and Kim Kardashian with a total of 25 out of 5000. Comparing former presidents Obama and Trump, they seem to have a lot of similar tweets, which may sound wrong in the first place. On a second look however, both are politicians probably talking about the same topics, so this can be comprehended. Least similar twitter users seem to be Neil deGrasse Tyson and Kim Kardashian, only sharing 53 pseudo-parallel tweets only with similarity scores less than 0.7. While all of these observations seem reasonable, also unexpected results can be found, e.g.

the high amount of pseudo-parallel data for Donald Trump and Neil deGrasse Tyson.

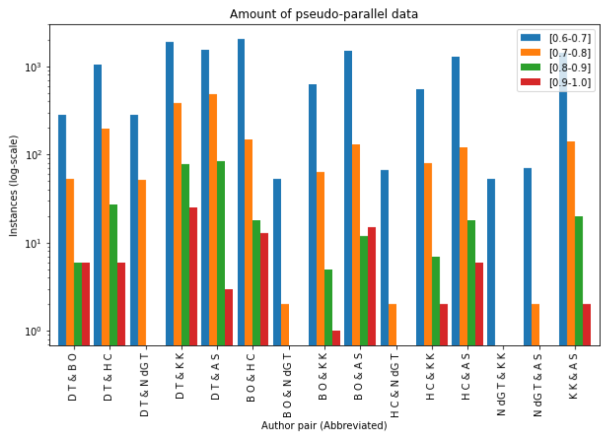

Fig. 4. Amount of pseudo-parallel data found categorized by author pair and similarity score

C. Style embeddings

Talking about style embeddings, our results display the average similarity of the remaining vectors of each author. A high number means that most differences of those sentence embeddings are equal, so we suggest that these remaining vectors represent style in some form. However, not only style may be taken into account; the authors opinion could also be represented by this remaining vectors. Both aspects would explain the high values for Trump and Obama, as they each use a highly different style of writing, but also represent opinions at different ends of the politic spectrum. In addition to that, Obama and Clinton seem pretty equal, which could be reasoned by both following their party's political line for opinion and being members of the highly educated American upper class.

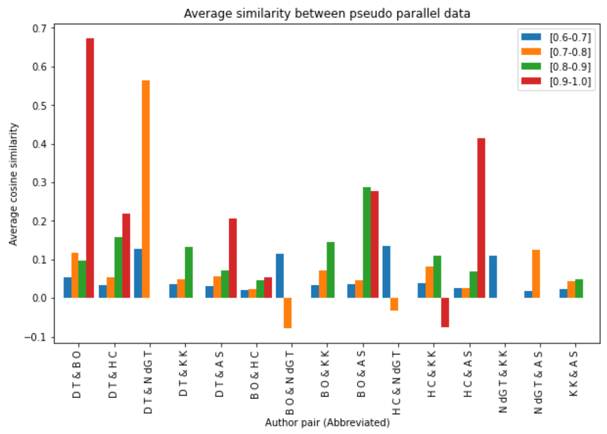

Fig. 5. Similarity of styles in clusters with respect to pseudo-parallel data similarity

VIII. Conclusion

Clustering and topic modelling produce decent results, but choosing the parameter for number of clusters is elementary to prevent the model from overfitting on the one hand, where too little clusters would be generated, and underfitting on the other hand, where topics are too general. Both topic clustering and sentence embedding seem to be able to produce pseudo-parallel data with high cosine similarities of between roughly 0.7 and 0.9. Values outside that range are either not similar enough or too identical, as wishing somebody a happy birthday doesn't contribute to writing style at all.

To conclude, style embedding seems to be a promising concept of calculating style differences for author pairs, but results are quite mixed, as some pairs produce unexpected and inexplicable results. Based on our tests, it is hard to draw any real conclusions without implementing an auto encoder or other style-transfer models.

References

[1] Y. Liao, L. Bing, P. Li, S. Shi, W. Lam, and T. Zhang, "Quase: Sequence editing under quantifiable guidance," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3855–3864, 2018.

[2] V. John, L. Mou, H. Bahuleyan, and O. Vechtomova, "Disentangled representation learning for non-parallel text style transfer," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 424–434, Association for Computational Linguistics, July 2019.

[3] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 11 2019.

[4] Z. Fu, X. Tan, N. Peng, D. Zhao, and R. Yan, "Style transfer in text: Exploration and evaluation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, Apr. 2018.

[5] O. Hourrane and E. H. Benlahmer, "Rich style embedding for intrinsic plagiarism detection," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 11, 2019.

[6] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[7] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[8] N. Reimers, "fast_clustering.py." UKPLab Sentence-transformers GitHub repository, 2021.

[9] D. M. Blei, A. Y. N. Ng, and M. I. Jordan, "Latent dirichlet allocation," in *Journal of Machine Learing Research 3*, 2003.