Natural Language Processing

Authors:
Janusz Jakub Wilczek, jawi@itu.dk
Mariusz Oskar Kurek, maku@itu.dk
Lini Zhang, linz@itu.dk
Julia Trznadel, jtrz@itu.dk

Introduction

The recent century has brought to humanity many innovations, which expanded the way we interact within society and with the universe. Many of these innovations are doubly edged swords, like the discovery of modern explosives by Albert Nobel, which allowed quicker extraction of priceless resources below the surface of Earth. On the other hand, the same technology was used to effectively bring death to thousands of people. In this paper, we investigate a two-faced aspect of information posted on Twitter. Similar to the aforementioned discovery, Twitter enables two fundamentally different results, one positive and one harmful. We aim to investigate a way how we can effectively categorise tweets.

Natural language processing (NLP) is a branch of computer science and linguistics that focuses on the analysis and processing of human language. It deals with natural language, a human-generated data highly distinguished from other kinds of data like images. The challenging part of NLP is the ambiguity and complexity of human language.

With the popularisation of social media, the amount of content grows respectively to their popularity. Unfortunately, some users post inappropriate, harmful, and harassing messages. Given that every second on Twitter, 10,000 tweets are published (internetlivestats.com, 2022), and assuming one person can evaluate a tweet every second, it would require 10,000 people working non-stop to remove all the harassing content, which is neither cost-effective nor possible. Therefore, an automatic detection method is desirable.

Furthermore, we investigate the classification of sentiment as it caught our attention. The field of sentiment analysis has grown in popularity recently (Salinca, 2015) and some implementations of it managed to find their way into the most popular business intelligence tools, such as Microsoft Power BI (Forum Users, 2022). We reckon it a challenging task for humans, so we want to observe how well can a computer execute this task. In addition, the importance of sentiment analysis was well expressed by (Chaturvedi, Mishra, & Mishra, 2017) “Due to tremendous expanse of opinions of users, their reviews, feedbacks and suggestions available over the web resources, it is so much indispensable to discover, analyse and consolidate their views for enhanced decision making.”

We investigate how machine learning (ML) algorithms can be applied to automatically detect the category of tweets.

Data and pre-processing

In this project, we used a total of 5 datasets, namely offensive, sentiment, Gatsby, profanities, and hate speech. Offensive and sentiment consist of posts from Twitter. Each line of these datasets is a single tweet and all users are anonymised as “@user”. Dataset Gatsby consists of the context of a book, as the name suggests. The last two datasets were used in data augmentation and are discussed in more detail later.

We assumed that all our datasets are written in English and contained correct labels. We started with a simple binary detection task, offensive language detection. We aimed to predict senders' intentions with ML, in other words, we trained an algorithm on the data with manually annotated labels to later predict labels for unknown inputs.

Before we could create any ML model to annotate data, first we had to analyse our datasets and extract features from them. Tokenisation was the first step. It has become customary in NLP to name words as tokens, which nomenclature we comply with in this paper.

We discussed what ideal tokenisation looks like and concluded that it removes stop words, considers emojis as tokens as they can be offensive (Malkin, 2016), and does not separate hashtags from a hash sign. Our tokeniser is based on the demo code provided by the lecturer. However, we have made significant changes to it to suit our presumptions.

We compare our tokeniser implementation to the one available in the NLTK package. Figures 2 and 3 show plots in a logarithmic scale, which try to verify if tokenised data is compliant with Zipf's law. According to the Britannica encyclopaedia (Hosch, 2009), Zipf's law defines a property in the English language that the frequency of any word is inversely proportional to its rank in the frequency table. It means that the most common word appears about 10% of the time in any text, whereas the second most common word appears about 5%, the third around 3.3% and so on. Given that property, we should observe Zipf's law in our data.

Ideally, in a logarithmic scale, points derived from the rank and frequency of words should create a line. Figures 1 and 2 contain such red lines. The data points from tokenisers are connected with a blue curve.

Figure 1 shows Zipf's law plots for the offensive and sentiment dataset, created with our tokeniser. Figure 2 shows respective plots but created with the other tokeniser. As we look at the graphs, we do not see major discrepancies between the two tokenisers, but we note that our tokeniser seems to produce results slightly more consistent with Zipf's law, yet it is not perfectly compliant with Zipf's law.

When we looked at tokens, which occur between 1 and 3 times, we found that our tokeniser created tokens, such as "u002c", which is a Unicode representation of a comma.

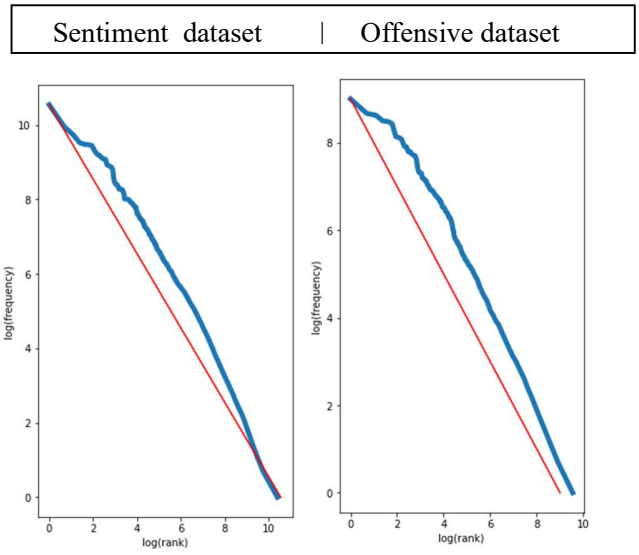

Figure 1 Zipf's law plot from our tokeniser

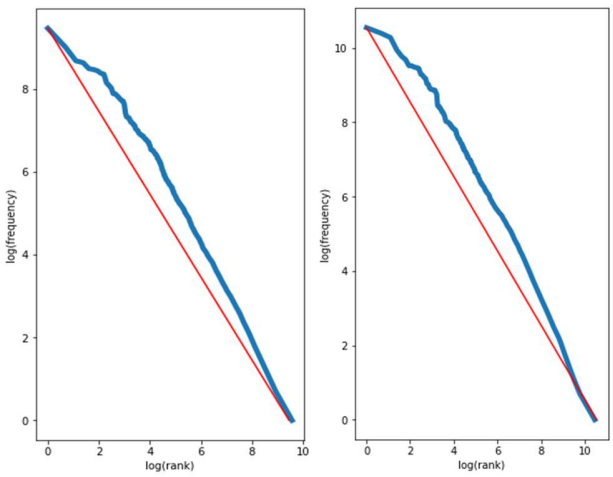

Figure 2 Zipf's law from NLTK tokeniser

If we look into our data, we can find such data entries. Thus, it was not an error of our tokeniser, but the issue of the data itself. Ideally, we should remove such errors, but we decided not to do that, as we think it should not substantially influence our final results.

Annotations

The ML algorithm requires that the training data have labels. The quality of the ML's predictions heavily depends on the quality of training labels. For example, if the labels are incorrect, i.e., offensive tweets are labelled as not offensive, then naturally the algorithm will not be accurate. Thus, we performed the manual annotation in the following manner:

1. Annotate the first 100 tweets individually.
2. Discuss the results, especially tweets on which we did not agree on how to classify.
3. Refine the guidelines.
4. Revaluate the first 100 tweets.

The 2nd and 3rd steps aim at minimisation of the discrepancies in labelling, as we might have perceived offence differently, which turned out to be true. We decided to use the following requirements on what is an offence:

- Cambridge Dictionary's definition of an offence (Cambridge Dictionary, 2022)
  - "Upset and hurt or annoyed feelings, often because someone has been rude or shown no respect"
- Vulgarisms are offensive
- The accusation is not offensive. For instance, if one accuses somebody of being a liar or calls out a fake, it is not an offence.

In addition, as we are not native English speakers, we would often look at the definitions of words in the Cambridge dictionary to determine if they are offensive.

The last point stems from the belief that people commit crimes. Thus, everybody should be permitted to freely express concerns regarding one's morality, as long as it does not include threats, vulgarisms, or other inappropriate language. Despite the guidelines, there were still instances on which we could not agree. For example:

"@user You are becoming the court jester of the senate"

This tweet divided us into halves, 2 of us considered it offensive, other 2 did not. We could see where each group was coming from regarding their decision. One group considered it as an invective, derogatory language, whilst the others treated it as an acceptable critique. Such disagreements show the limitations of any ML labelling. Since there exists a tweet, which cannot be uniformly classified by all people, how can a computer do it any better? This means the real maximum accuracy score for ML is below 100%. Nonetheless, automatic labelling still is feasible, but it will never be perfect.

We calculated inter-annotator agreement using 2 methods:

- Scott 1955
- Cohen 1960

The choice was motivated by our lack of a thorough understanding of the differences between methods to calculate inter-annotator agreement. Thus, we decided to empirically observe if there are significant disparity. The results were identical

if rounded up to the third significant digit. They were 0.613. Given the Rules of thumb (Landis and Koch, Biometrics 1977), our inter-annotator agreement was substantial.

Classification

Our data pipeline (Figure 3) begins with tokenisation tweets. We use CountVectorizer to convert tokens to numbers and to create n-grams. The ML classifier of choice cannot handle string datatype. However, one can express strings, which is the purpose of CountVectorizer. Next, we resample the data using SMOTE (Synthetic Minority Oversampling Technique), in case the training data is imbalanced. We have chosen this resampling technique because we have used it previously and obtained a satisfactory result. In addition, SMOTE can synthetically create new tweets, which can yield better results than other more rudimentary techniques. In addition, it is a form of data augmentation. We use TfidfTransformer instead of the raw frequencies of occurrences to scale down the impact of tokens that occur very frequently in a corpus and that are less informative than features that occur in a small fraction of the training corpus. Having designed the pipeline, we can use the ML classifier. In agreement with the project slides, we used SGDClassifier. To find the best scoring parameters, we used 126 unique parameter combinations. The parameters were:

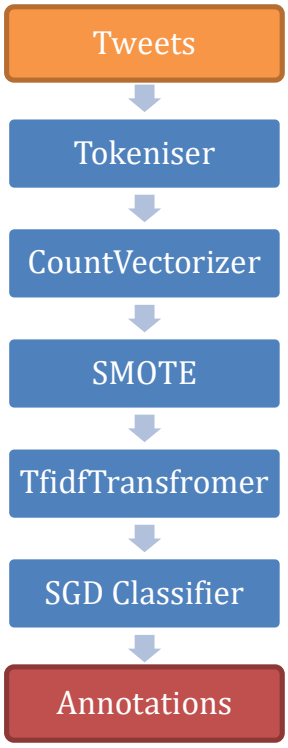

Figure 3 Pipeline flowchart

- 'clf__loss': 'log', 'hinge', 'squared_hinge', for SGD classifier
- 'vect__ngram_range': (1, 1), (1, 2), (2, 2), which are an order of n-grams
- 'tfidf__norm': 'l1', 'l2', which are different calculation settings for
- 'ovs__k_neighbors': range (2, 9), which are the number of neighbours to consider in SMOTE

We achieved the best F1-score of 68.0% for offence detection with a following combination:

- 'clf__loss': 'hinge',
- 'ovs__k_neighbors': 3,
- 'tfidf__norm': 'l2',
- 'vect__ngram_range': (1, 1)

We achieved the best results with unigrams, which can be explained by the intuition we got in the manual annotation. Often what determines if a text is offensive are singular words, like profanities.

We achieved the best F-score of 59.5% for sentiment detection with a following combination:

- 'clf__loss': 'hinge',
- 'ovs__k_neighbors': 7,
- 'tfidf__norm': 'l2',
- 'vect__ngram_range': (1, 1)

Once again, we unigrams were the best scoring parameter. However, we do not have as strong intuition as before. Similarly, singular words are good predictors of one's opinion, yet they can be misleading, as one can express a negative opinion with the use of positive words, like irony.

Data augmentation

Data augmentation (DA) is a term referencing a wide category of techniques, which aim at improving models and tasks later performed with these models. In our case, we used DA to improve the performance of classifiers in an NLP scenario. However, DA is not limited to NLP (Zou & Wei, 2019). DA allows boosting performance in cases where an extremely limited amount of data is available (Liu, et al., 2020).

We chose the offensive dataset for data augmentation since it was considerably smaller than the sentiment dataset. However, we should note that essentially, we performed data augmentation in each dataset via SMOTE, which creates new data points for evaluation. We decided on a few experiments with various data augmentation methods.

We chose the hate speech dataset from SemEval 2019 (International Workshop on Semantic Evaluation, 2019), as we think that hate speech and offensive language are strongly correlated. We have decided not to choose our data set under perplexity score, as ML models yielded the highest results with the use of unigrams, which suggests the irrelevance of a given word's neighbours. In addition, Gao et. Al. (2002) argue for such application in statistical language modelling (SLM), which is beyond our scope.

We followed the guidelines established in the paper investigating the validity of self-training models, which was written by Vincent Van Asch and Walter Daelemans (2016). The obtained final F-score was 54.0%, highly below the score we got without augmentation.

We experimented with an insight gained from manual annotation, which was the consideration of profanities as offensive. Hence, we obtained a dataset of numerous profanities, labelled them as offensive, and then train our ML model with this additional data. We presumed that some profanities might have not been present in the training dataset, yet they were the best predictors to categorise if a tweet is offensive. The additional data came from GitHub (Anger, 2022) and contained around 2850 instances of profanities. We trained the model solely on these profanities, then we proceeded to train the model on the offensive dataset. With such an approach we got an F-score of 52.7%, which was still lower than without augmentation, and comparable to the self-training approach.

Table 1 presents results from all ML cases we experimented with.

Type of ML experiment	Offensive, without data augmentation	Sentiment, without data augmentation	Offensive, with self-training	Offensive, with profanity data augmentation
F-Score (%)	68.0	59.5	54.0	52.7

Table 1 Summary of ML experiments

Conclusions and future work

The potential for further investigation is high considering the number of comments not only on Twitter but also on other social media platforms. Offensive comments are spreading across social media which can have a negative impact on society. *"The role that the Internet, particularly social media, might have in suicide-related behaviour is a topic of growing interest and debate."* (Luxton, June, & Fairall, 2012). Through ML there is a possibility to detect offensive comments, delete them and hence minimalize undesired messages. Regarding sentiment analysis, further research is especially desirable in the evaluation of various services and products. "*The task usually involves detecting whether a piece of text expresses a POSITIVE, a NEGATIVE, or a NEUTRAL sentiment; the sentiment can be general or about a specific topic, e.g., a person, a product, or an event.*" (Rosenthal, Farra, & Nakov, 2017). Sentiment analysis has a vast application in business, which is proved by the implementation of sentiment analysis in Power BI. For instance, Data augmentation techniques are valid and effective in increasing the results of ML. However, we were unable to replicate the reported improvements. Furthermore, the experiments ran with augmented data had lower scores than those without. It can be attributed to the fact that we did not run ML algorithms with all 126 parameter combinations, but only used parameters, which were found to be the best scoring before. Data augmentation might create a different scenario, which should be processed differently. Thus, due to the lower number of different settings we got lower results. Nonetheless, our findings do not diminish the value of data augmentation.

Furthermore, the imperfect inter-annotator score did not hinder our results, as we never used manually annotated data. However, we rely solely on the quality of the provided labels. Nonetheless, manual annotation showed us the limitation of automatic categorisation. Thus, it is hard to imagine it would be possible to create such a piece of software which would reliably annotate data with 100% accuracy.

Disclaimer

The majority of code, especially all elements regarding ML, were made only by Mariusz Oskar Kurek and Janusz Jakub Wilczek.

References

Anger, Z. (2022). *zacanger/profane-words: A very long list of English profanity.* Retrieved from GitHub: https://github.com/zacanger/profane-words

Cambridge Dictionary. (2022). *offence*. Retrieved from Cambridge Dictionary: https://dictionary.cambridge.org/dictionary/english/offence

Chaturvedi, S., Mishra, V., & Mishra, N. (2017). Sentiment analysis using machine learning for business intelligence. *IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI).* Chennai, India: IEEE.

Forum Users. (2022). *How to: Use Sentiment analysis and Opinion Mining*. Retrieved from docs.microsoft.com: https://docs.microsoft.com/en-us/azure/cognitive-services/language-service/sentiment-opinion-mining/how-to/call-api

Gao, J., Goodman, J., Li, M., & Lee, K.-F. (2002). Toward a unified approach to statistical language modeling for Chinese.

Hosch, W. L. (2009). *Zipf's law*. Retrieved from Britannica: https://www.britannica.com/topic/Zipfs-law

International Workshop on Semantic Evaluation. (2019). *SemEval-2019*. Retrieved from International Workshop on Semantic Evaluation: https://alt.qcri.org/semeval2019/index.php

internetlivestats.com. (2022). *Twitter Usage Statistics*. Retrieved from Internet Live Stats: https://www.internetlivestats.com/twitter-statistics/

Liu, R., Xu, G., Jia, C., Ma, W., Wang, L., & Vosoughi, S. (2020). Data Boost: Text Data Augmentation Through Reinforcement Learning Guided Conditional Generation.

Luxton, D. D., June, J. D., & Fairall, J. M. (2012). Social media and suicide: a public health perspective.

Malkin, B. (2016). *News*. Retrieved from The Guardian: https://www.theguardian.com/technology/2016/aug/02/apple-replaces-gun-emoji-water-pistol-revolver-violence-debate

Rosenthal, S., Farra, N., & Nakov, P. (2017). SemEval-2017 task 4: Sentiment analysis in Twitter.

Salinca, A. (2015). Business Reviews Classification Using Sentiment Analysis. *17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC).* imisoara, Romania: IEEE.

Van Asch, V., & Daelemans, W. (2016). Predicting the Effectiveness of Self-Training: Application to Sentiment Classification.

Zou, K., & Wei, J. (2019). EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks.