Semantic Analysis and Classification of Twitter Posts

George Darkens, *Member, IEEE, gd17659*

Abstract—Natural language processing has seen increasingly rapid advances in recent years stemming from the explosion of readily accessible raw language data from the World Wide Web. With the huge corpus that the Web provides, researchers have been quick to utilise this source of language data to train and test their language models. Despite this, some areas cause issues for these models as they don't conform to traditional formal language rules, and in some cases create and evolve separate dialects entirely.
Social media presents one of these problems, notably Twitter, which promotes short, conversational, and idiosyncratic messages that sometimes are almost devoid of meaning when removed from their time and place on the Web. Language models aren't trained on such a corpus, nor do they have the capabilities to interpret and classify it by extrapolating from their existing training. This paper tests multiple approaches to detecting sentiment and hate, as well as attempts to predict which emoji was used in a tweet, using multiple custom modified classifiers.

Index Terms—Natural Language Processing, Social Media, Semantic Analysis, Twitter, Language Classification, Language Model

1 Introduction

Natural language processors (NLPs) are the culmination of years of research in both computer science and linguistics, marrying the two together to create systems capable of processing, analysing, and comprehending human language. In it's original application, it was a simple translator from Russian to English over 60 years ago [1], but with rapid increases in computational power and access to the largest conceived corpus ever, modern processing models are evolving at an incredible place. Some systems, such as the Generative Pre-trained Transformer's (GPTs) created by T. Brown et al. at OpenAI, are consistently on the cutting edge of NLP technology. The latest version, GPT-3, can perform tasks far beyond regular language processing; being able to code in many programming languages, write music for various instruments, and even generate language of it's own that is practically indistinguishable from that which is created by humans. [2], [3]

A model such as GPT can only be good as the sum of it's parts, and GPT is trained on enormous amounts of data from the World Wide Web. With the amount of data created by the Web increasing exponentially each decade, yet remaining freely accessible, using Web sources as a corpus has been commonplace for researchers in the NLP field for some years. GPT is trained on billions of tokens of data - of those billions of tokens, only around 15% are from non-internet sources, most of them books. [3, Fig. 2.2]

NLPs of today are well suited to some tasks, but struggle in areas in which they have had little effective training. One area notoriously hard for NLPs to understand and process accurately is social media text, specifically the short-form platform Twitter. The way that users use their given language on social media can be very far removed from traditional, formal speech that many models train on. Twitter's raison d'être is presenting short form, singular focused ideas - forced to be under 240 characters in length.[1] Factors like the use of hashtags and emojis are atypical for any traditionally trained NLP, thus presenting a unique opportunity to develop and train models capable of interpreting and classifying the incredibly noisy corpus of social media.

TweetEval [4] is a standardised benchmark created by F. Barbieri et al. designed to evaluate the performance of NLP models over several heterogeneous tasks on a curated Twitter corpus. Of these tasks, this paper will discuss three; emoji, the prediction of the most likely emoji in a tweet; hate, predicting if a tweet targets hate towards women or immigrants; and sentiment, predicting if a tweet is positive, negative, or neutral in tone.

2 Literature Review

The work of F.Barbieri et al. in the *TweetEval* paper is the foundation for this projects work as well as other researchers in the same area. Providing a testing methodology and curated datasets for specific tasks alleviates many issues by allowing models to be directly comparable to each other. [2]

TweetEval's paper discusses the complexities in understanding the language used on Twitter. Users tweets are "...high-paced, conversational and idiosyncratic..." [4, Ch. 1] in nature, with their short text length providing little information from which to glean an understanding for tasks such as emotion and sentiment recognition. In pointing out the complexity of the domain, they also suggest that much like in other areas, a "...lack of a unified evaluation framework" holds researchers back from more rapid progress - thus the creation of their benchmarking system.

[1] In 2017, Twitter doubled their allowed length from 140 to 240, however the datasets being used in this paper have no tweets over 140 characters.

[2] *TweetEval*'s GitHub has a leaderboard for the highest scoring models: https://github.com/cardiffnlp/tweeteval

TweetEval's paper also tested various models on their system which they divided into various categories to determine which worked best. One category was models that were trained on traditional corpora, a second trained only on Twitter language data, and a third that was pre-trained on traditional corpora before further Twitter-specific training. The results of these tests led the authors to suggest "...that using a pre-trained [language model] may be sufficient, but can improve if topped with extra-training on in-domain data" [4, Ch.5, p.5]

The authors suggest the reasoning behind these results can be found in another study by Hu et al. In their paper, the authors complete an analysis of the spectrum of language on Twitter compared to other mediums such as SMS, E-Mail, blogs, and broadsheet newspapers. They came to the conclusion that "...Twitter is markedly more standard and formal than SMS and online chat...", making the text found on there "...closer to email and blogs, and less so than newspapers." [5, p.245, Ch.1] On the other hand, they describe Twitter users to be developing their own "linguistically unique" styles compared to the other mediums they compared to, even those that were also online. The example they provide is the use of both first-person and third-person pronouns, whereas other mediums typically stick to one.

A paper attempting some of the same goals is seen in Alexander Pak and Patrick Paroubek's paper entitled *'Twitter as a Corpus for Sentiment Analysis and Opinion Mining'* [6]. The authors discuss the importance of the Web as a "rich [source] of data for opinion mining and sentiment analysis" that has attracted little research attention. The authors developed a classifier capable of determining sentiment much like the one this project proposes. Their approach utilised Part of Speech Tagging (POS-tagging) alongside other techniques to find linguistic indicators of emotional text. POS-tagging involves grammatically tagging words and sentences to identify grammatical syntax like nouns, adjectives, and verbs to greater understand meaning conveyed in text.

To train their model, they used a corpus of 300,000 tweets that contained either happy or sad emoticons.[3] They assumed that in such a tightly constrained medium of speech where the word count was only 140 characters (prior to Twitter doubling the limit on characters), that a emoticon would accurately represent the emotion of the tweet. Other such papers have used emojis and emoticons to derive sentiment, or to interpret hate speech against groups to great success, reducing the time required to determine sentiment when combined with existing techniques. [7]

3 Methodology

The dataset used is that provided by the *TweetEval* paper for their benchmark. Of the seven heterogeneous topics, the author chose sentiment [8], hate [9], and emoji [10] prediction. Each of these tasks has a mapping for the possible values that the tweet can have for it's classification. The 'hate' mappings are binary - either the message is a hate message or it is not. For sentiment, it is a ternary position of positive, negative, or neutral - and for emojis it is an assortment of the 20 most popular emojis.

Each of these tasks has it's own dataset curated for it, with a corpus of testing and training files consisting of pre-screened and pre-tagged tweets. The training files are used, as the name suggests, for training the model prior to testing it on the real test tweets to evaluate the success of the model. The tweets provided by *TweetEval* are preprocessed somewhat before they are distributed. Their paper discusses an equal pre-processing pipeline for all tasks where "user mentions are obfuscated and line breaks and website links are removed". Indeed, the author found user names to all be replaced with "@User", however some links remain such as Twitter's URL shortening domain https://t.co/ which seems to escape the techniques used to clean other links.

This project has it's own pre-processing pipeline for it's model that goes significantly further than the almost raw corpus that *TweetEval* provides. When the tweets are first processed, they are combined into an array that stores the tweet's raw text alongside it's mapping tag for future reference. All pre-processing steps that occur to the tweet are stored separately in a new column. This new column is progressively processed by different steps until it is 'fully processed'.

Firstly, all tweets are case folded into lowercase using a simple .str.lower() method on each row. Then a process for removing 'handles' is used, removing the '@User' strings that have been left behind by *TweetEval*'s pre-processing. Removing as much data as possible that carries no meaning is important to raise efficiency of the model as well as to improve accuracy. In this scenario, the '@User' string conveys practically nothing, and is used in many tweets regardless of message or meaning.

Next, the tweets are stripped of all punctuation markers using Regular Expressions (regex) through a tokenizer (although not tokenizing the tweets yet). Keeping punctuation was a considered feature to the model, as some punctuation can be good for determining sentiment, and some are used to form emoticons (note that emojis emoticons). However the author found that the use of punctuation in online mediums such as Twitter to be too idiosyncratic, with punctuation being used sarcastically or to convey meanings that escape easy definition. Simple features such as ellipsis' "..." can convey shock, apathy, or a simple trailing sentence. Whilst other features of language also possess this lack of exact meaning, punctuation was a feature that was not trained into the model for this paper. However, the author believes that with additional work this could add value to a model should one be trained for punctuation on a specifically online medium.

The next processing step is the removal of 'stop words', a process common to almost all language processors. Typically natural language processors have their own list of stop words, removing items such as function words like pronouns, auxiliary verbs, and conjunctions for example. In this project, the author used the Natural Language Toolkit (NLTK) stop word list comprising of around 130 entries.

Now the tweets are quite far removed from their original text, with no punctuation and minimal filler words, all that remains are the remaining contextual clues from which the model must determine a classification. These contextually distilled tweets are then stemmed using a snowball stemmer from the NLTK. There are various stemmers to choose from,

[3] Note that emoticons are not emojis. Emojis are pictographic representations, whilst emoticons are ones such as :) and :(

each following a set of rules to reduce words down to their word stem. For each tweet, every word is individually parsed through the stemmer and the resulting stem takes it's place.

Finally, the processed tweets are stripped of preceding and trailing white-spaces before being tokenized, sequenced, and padded. Each word from the tweet is taken and sequenced, turning the tweet into an array of integers. Then all strings are padded or truncated to meet a fixed length, regardless of original size.

At the conclusion of this pre-processing stage, tweets are no longer directly readable unless converted back to plain text. What remains of the tweets is a distilled form of what is determined to be important - stemmed versions of all words that convey significant meaning from which the model can derive enough to classify the tweet.

4 Results

Results are split into their respective sections, although the underlying processes behind the garnering of results for each of them is similar. For most of these models, the author set predefined hyper-parameters to be iterated through by GridSearchCV to best fit the estimator model being used. The best parameters were picked based on their overall F1-macro performance.

4.1 Emoji Prediction

Emoji prediction turned out to be the poorest performer of any model, no matter the approach taken. Results from the *TweetEval* leaderboards suggest a similar problem faced by other models, with a highest macro F1 score of 0.33 from the *BERTweet* model.

As expected, initial tests with various modified dummy classifiers produced incredibly poor macro F1-scores, around 0.2-0.7 typically as the model struggled to achieve any classification of value. Those emojis that had the greatest F1-scores were the top three emojis in the dataset, the heart, heart-eyes, and crying laughing emojis. Of these, the dummy classifiers attained F1-scores of 0.21, 0.11 and 0.10 respectively.

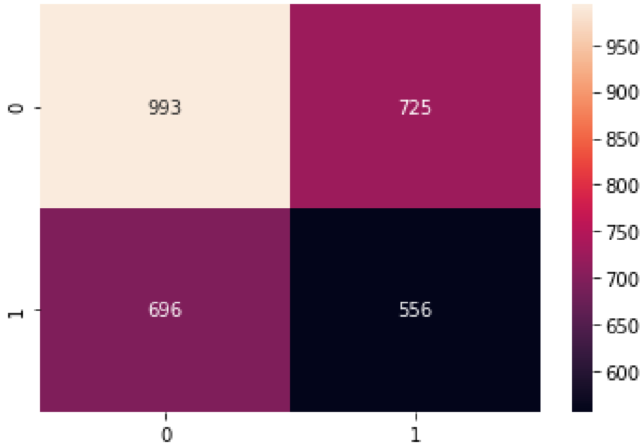
Results from a modified decision tree produced better results with a macro F1-score of 0.14. The performance among the top three emojis was once again significantly better than the others, with respective scores of 0.37, 0.15, and 0.24. One particular emoji also saw above average F1-score - the Christmas tree emoji with an F1-score of 0.5, likely due to the ease of learning the applicability of the emoji to contextual clues of the holiday season. The results of the modified decision tree closely match those of the C-Support vector classifier which had an overall F1-score of 0.15, but also attained better results for the top three emojis, at 0.4, 0.16, and 0.33 - with the Christmas tree at 0.62. Some other emojis also saw better scores than typical, the sun emoji at 0.32, the USA flag at 0.37, and the fire emoji at 0.34.

Finally, the linear support vector classification (LinearSVC) results improve upon the overall F1-score again reaching 0.2, however performance across emojis is erratic even for those that are typically easier to predict. The usual top three have scores of 0.26, 0.16, and 0.33, with the Christmas tree at 0.59 and the USA at 0.46.

4.2 Hate Prediction

The results for the hate dataset improve upon those in the emoji dataset due to the decreased complexity for classification with only two values - 'hate' or 'not-hate'.

With the dummy classifiers, an F1-score of 0.51 was achieved, already surpassing all results in the emoji dataset. However this result surprisingly turns out to be the strongest overall F1-score of all hate classifiers trained and tested, if only by a small margin in some cases. The individual scores for the detection of hate and non-hate were 0.58 and 0.44.



The modified decision tree achieved a macro F1-score of 0.45, but improved upon detecting hate by 32%, achieving a score of 0.58 for hate detection. This improvement in hate detection persists in the C-Support vector classifier which detected hate with an F1-score of 0.6, but an overall F1 of 0.45 - the same as the modified decision tree.

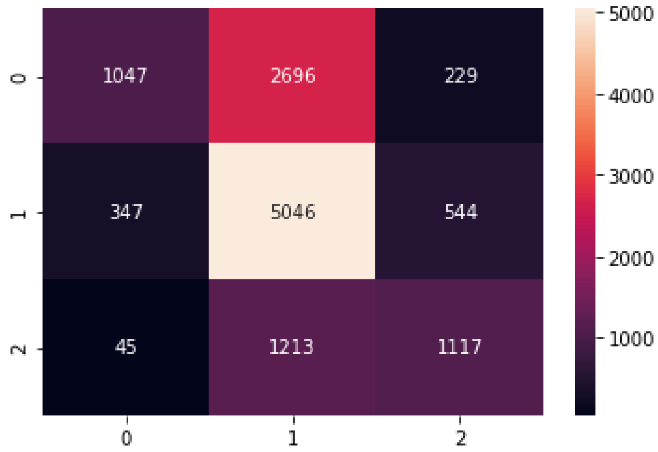
The results for the last two models tested were largely the same. Both the forest and the naïve Bayers classifier achieved overall F1-scores of 0.42, with hate and non-hate detection at 0.27 and 0.6.

4.3 Sentiment Prediction

Results for sentiment were also improvements upon the emoji dataset performance, with the increase in possible classifications two three (positive, neutral, negative) not depressing results significantly.

Dummy classifier results were not great, achieving 0.31, with the easiest sentiment to detect being neutrality at 0.46. As seen previously, the decision tree classifier results improved upon this in all areas. An overall F1-score of 0.49 and neutrality detection of 0.61, with detection of negativity at 0.38 and positivity at 0.48.

C-Support vector classifier achieved an F1-score of 0.53 overall, whilst also hitting an F1 of 0.69 in neutrality detection. Negativity was good at 0.52, but positivity was poorer performance at only 0.39, dragging the overall F1 down.



Random forest classifier is another example of excelling at detecting neutrality with a score of 0.66 in that classification, but struggled to perform in positive sentiment recognition at only 0.25. With an overall F1-score of 0.56, the random forest classifier was the strongest model for sentiment.

Finally, the results for the linear SVC were just short of the strongest with a macro-F1 of 0.55, surprisingly being better at detecting both positive (0.62) and negative (0.55) sentiment, but lacking in the area that other models do best, achieving 0.48 for neutrality.

5 Discussion

The poor performance in the emoji prediction was not unexpected, with prior results from other models posting significantly worse scores in emoji compared to every other task. The complexity of this task is discussed by the *TweetEval* paper when they describe the task, "...due to their skewed distribution, this task proved to be highly difficult, with low overall numbers." [4, Ch. 2.1] They go on to state that 42% of tweets use one of three emojis, the 'red heart', the 'smiling face with heart-eyes', and the 'face with tears of joy'. Emoji prediction seems to require a greater deal of training data compared to other tasks, and the *TweetEval* paper discusses how poor results in emoji prediction stem from 'the downscaling of the training data' due to Twitter placing limits on their API usage. [4, Ch. 4.2]

For most of the models, the emojis most accurately predicted were those top three that appeared in 42% of tweets, possibly because the models had more data to train off. Scores for emojis that are more likely to have surrounding context, such the Christmas tree or USA emoji, also fared better than the other emojis.

Notable for the hate dataset is that all the models detected hate easier than non-hate. The simple explanation is that hate tweets have more context clues to train off, whilst non-hate is harder to predict since it is an absence of something that is already not well detected.

Results for sentiment prediction were the best of all three chosen tasks, but only by a slim margin. The author had expected performance to be worse due to the increase in categories, but all models seemed adept at detecting neutrality, the easiest category to detect since it is that which is devoid of all markers for hate and non-hate. The result of the C-Support vector classifier was interesting, with good performance all around but the best in detecting positive and negative sentiment - a contrast to all other models.

6 Conclusion

Various features were considered for this project that were too logistically difficult to complete in the given time frame, but that the author considers to be interesting avenues of further research. The author stripped punctuation from tweets before processing, however they believe punctuation could be a useful denotation of possible emotion. Not only is punctuation used to form emoticons in text, it also helps convey meaning to sentences and phrases. Whilst basic rules could assist in picking out emoticons and simple punctuation markers for emotion, a model specifically trained to understand the use of punctuation online could greatly assist in deriving features such as emotion.

Another feature that the author considered was Part of Speech Tagging (POS-tagging). POS-tagging was partially implemented in the original non-Jupyter code base that the author used, however he never found a way to adequately introduce POS-tagged data into the system. Noun-phrase chunking is one such possible example of a technique that could greatly assist in determining the context in speech, however the ability to train this into a model seems beyond the scope of this project.

The results of this project are not of a quality to broach the *TweetEval* leaderboard against their trained models. However, the author believes it shows the varying approaches to classification in model selection that can result in unexpectedly good quality results given a relatively small training data set.

Acknowledgments

The author would like to thank those on the *TweetEval* team for their work in curating and publishing a substantial corpus for research and educational purposes. The time and effort spent in acquiring, tagging, and processing for public use is greatly appreciated.

References

[1] J. Hutchins, "The first public demonstration of machine translation: the georgetown-ibm system, 7th january 1954," 01 2004.

[2] A. Sabeti, "Gpt-3: An ai that's eerily good at writing almost anything," Jul 2020. [Online]. Available: https://arr.am/2020/07/09/gpt-3-an-ai-thats-eerily-good-at-writing-almost-anything/

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[4] F. Barbieri, J. Camacho-Collados, L. Espinosa-Anke, and L. Neves, "TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification," in *Proceedings of Findings of EMNLP*, 2020.

[5] Y. Hu, K. Talamadupula, and S. Kambhampati, "Dude, srsly?: The surprisingly formal nature of twitter's language," *Proceedings of the 7th International Conference on Weblogs and Social Media, ICWSM 2013*, pp. 244–253, 01 2013.

[6] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining," in *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. Valletta, Malta: European Language Resources Association (ELRA), May 2010. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2010/pdf/385$_Paper.pdf$

[7] J. Read, "Using emoticons to reduce dependency in machine learning techniques for sentiment classification," in *Proceedings of the ACL Student Research Workshop*, ser. ACLstudent '05. USA: Association for Computational Linguistics, 2005, p. 43–48.

[8] S. Rosenthal, N. Farra, and P. Nakov, "Semeval-2017 task 4: Sentiment analysis in twitter," in *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, 2017, pp. 502–518.

[9] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. M. Rangel Pardo, P. Rosso, and M. Sanguinetti, "SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter," in *Proceedings of the 13th International Workshop on Semantic Evaluation*. Minneapolis, Minnesota, USA: Association for Computational Linguistics, 2019, pp. 54–63. [Online]. Available: https://www.aclweb.org/anthology/S19-2007

[10] F. Barbieri, J. Camacho-Collados, F. Ronzano, L. Espinosa-Anke, M. Ballesteros, V. Basile, V. Patti, and H. Saggion, "Semeval 2018 task 2: Multilingual emoji prediction," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 24–33.