Assessing the Effects of Lemmatisation and Spell Checking on Sentiment Analysis of Online Reviews

James Kavanagh
School of Computing
University of Kent, UK
ORCID: 0000-0001-7822-4969

Keith Greenhow
School of Computing
University of Kent, UK
ORCID: 0000-0001-9263-5086

Anna Jordanous
School of Computing
University of Kent, UK
ORCID: 0000-0003-2076-8642

***Abstract*—With many options for text preprocessing techniques, choosing the most efficient methodology is important for both accuracy and computational expense. Online text often contains non-standard English, spelling errors, colloquialisms, emojis, slang and many other variations that affect current natural language processing tools, with no clear guidelines for preprocessing this type of text. In this work we analyse text preprocessing techniques using a data set of online reviews scraped from iTunes and Google Play store. The objective is to measure the efficacy of different combinations of these techniques to maximise the amount of detected sentiment in a dataset of 438,157 reviews. Sentiment detection was performed by two state-of-the-art sentiment analysers (RoBERTa and VADER). Statistical analysis of the results suggest preprocessing strategies for maximising sentiment detected within mental health app reviews and similar text formats.**

***Index Terms*—NLP, Language parsing and understanding, Web text analysis, Sentiment analysis**

I. Introduction

People across the world give online reviews on products/services. Such data holds value to various stakeholders, creating need for efficient ways to scrape and process this data [1]. In using natural language processing (NLP) techniques, effective preprocessing allows improved detection / interpretation, with significant effects on the results [2].

Mediatisation impacts language use in social media corpora [3]. Most NLP tools are designed for error free text, whilst the majority of natural text in large corpora is not; typically containing non-standard English, spelling mistakes, colloquialisms, emojis, slang etc. As Chai (2022) states: "[T]ext preprocessing is not a one-size-fits-all process. [...] [T]here is not a set of comprehensive guidelines that can advise researchers on which text preprocessing practices to apply to a brand-new text corpus." [2]

To address this gap, we investigate preprocessing strategies for online reviews. We focus on detecting sentiment in a specific set of app store reviews to test the efficacy of lemmatisation and spell checking. Hence we propose guidelines for lemmatisation and spell checking use based on priorities for accuracy and time/resource costs. [1]

[1]Full dataset and code available on GitHub: `github.com/Jak-Kav/ICSC_2023_Kavanagh_Greenhow_Jordanous`

A. Related Work

We adopt sentiment analysis (SA), an important task in NLP, as our proof-of-concept use case. Two approaches to SA are (1) lexicon based and (2) machine learning based methods [4]. We note that appropriate methods for preprocessing depend on the approach used [2].

Work studying effects of text preprocessing on SA of Brazilian Portuguese tweets [1] uses heuristics in machine learning algorithms to obtain an accuracy and polarity bias, resulting in increased classification accuracy.

Work exploring preprocessing in opinion mining from Google Play reviews (in Portuguese) [5] uses different stages *(1. Without any preprocessing, 2. Remove accentuation, punctuation, special characters, numbers and all to lower case, 3. Stage 2 with a spell checker, 4. Stage 3 with stemming)*, training their own ML model to handle data pre-labelled with sentiment using the 1-5 star rating. While they find no significant improvement in results with their ML models, they show star ratings are a valid way of labelling training data with sentiment.

II. Research Goals

The goal of this research is to investigate the effect of preprocessing and spell checking on sentiment detection on reviews. This paper is not comparing the efficacy of the different types of SA methods, so uses 'out of the box' SA models. The results are passed through both a lexicon based SA process, and a pre-trained machine learning SA process, to investigate differences between preprocessing requirements for either of the two main approaches to SA.

Part-Of-Speech (POS) tagging, lemmatisation and spell checking are the most computationally expensive tools. This research aims to establish if performing POS tagging and lemmatisation produce a significantly different result than not performing it, and also to establish if any of the six variations in spell checking produces a significantly different result than not performing it.

A. Data Set

We focus on improving the detection of available sentiment in online text. The choice of using online reviews was preferred as this will contain positive, negative and neutral sentiment. The choice to work with reviews of mental

health apps was made because the app itself is linked to the emotional state of the reviewer, and is likely to contain a large volume of sentiment-rich, authentic speech.

Our dataset was obtained by scraping the iTunes and Google Play stores for reviews of seven mental health apps, for a data set of **438,157** unique reviews. The seven mental health apps are `Replika, Wysa, Woebot, Daylio, MoodMeter, iMood Journal and eMood Tracker`.

B. Tools Used

We used 'Natural Language Toolkit' (NLTK) for many of the required tools, including POS tagging, lemmatisation, and **'TweetTokenizer'** tool, which has been shown to perform well for this type of corpora [6]. The set of 'stopwords' used was the union of the default `nltk.corpus.stopword` set and the set of singular punctuation characters, {!,?,&,⋯}. The sentiment analysers (used with default parameters) are the **VADER** [7] model (a lexicon based SA provided in NLTK), and the **RoBERTa** [8] model (a pre-trained ML model).

C. Spell Checkers

Online text often requires a more nuanced approach to spellchecking for handling language such as slang, acronyms, punctuation errors and deliberate misspelling [9]. We used three different types of spell checkers, `TextBlob`, `Symspell`, and `Symspell Compound`. These three were chosen as they are highly cited [10], well documented, and allow creation of custom dictionaries. Symspell Compound is applied prior to tokenisation as it works on the entire texts. Symspell and TextBlob both function on individual words so occur after tokenisation.

III. Control Data Set

This study focuses on the amount of sentiment (not whether it is positive or negative). We work on the assumption that evidence for maximising sentiment detection is if *more* sentiment is detected.

To test our assumption that the experiment was *detecting* sentiment, and not *generating* false sentiment, a control data set was put through the method and tested. The control data set was a list of 'Amazon' reviews with their star rating [11], as a review's star rating has been shown to be a valid indicator of its sentiment [5]. An inverted-bell curve graph of the mean score for each star rating would show that sentiment is only being detected, not generated. After passing the control data set through, it did indeed produce a parabolic (U-shaped) graph.

IV. Method

A. Spelling Correction

Standard spelling correction techniques are likely to have a negative impact [9, 5]. So, creating a custom dictionary could allow spelling correction to have a positive impact on the corpus. The hypothesis is that with the additional terms added into a custom dictionary allowing for urban words and spellings, this will reduce false corrections that may alter the sentiment of a review.

A dataset consisting of 1,048,576 terms from 'Urban Dictionary' was obtained [12]. Each term had a 'thumbs up' and a 'thumbs down' score, generated by users voting on the definition given. We gave each term a total score by subtracting the 'thumbs down' score from the 'thumbs up' score, and removed any terms with total score less than 1. Duplicate terms were removed leaving only the term with the highest score. Lastly, the app names were added to avoid correction. The resulting list of 230,659 *unique* unigrams was appended to copies of the original TextBlob and SymSpell dictionaries.

B. Tokenising

Each review is tokenised using TweetTokenizer. After each review has gone through the preprocessing pipeline, the result is a list of tokens. However, it is not possible to pass a list of tokens to either VADER or RoBERTa, so each list of tokens is 'detokenised' back into a single string using `TreebankWordDetokenizer` from NLTK.

C. Order of Processes

As each review enters the pipeline, any formatting marks from the source are removed. Next, if the pipeline includes spell checking before tokenisation, the review is spell checked as a single (compound) string. The review is then tokenised using TweetTokenizer, and if not already spell checked, each token is then spell checked individually.

The review has **not** been converted to all lower case yet, and the punctuation has not been removed or adjusted either. Caution is taken when removing punctuation from the corpus [2], as a reviewer's use of punctuation in an emotional context, e.g. repeated exclamation marks "!!!", can change the level of sentiment in that review [13]. Punctuation and capitalisation are useful information for the POS tagger [14], and the lemmatiser [15].

It is at this point that 'stopwords' are removed. Like punctuation and capitalisation, stopwords are important for increased accuracy of POS tagging and lemmatisation, and so they are only removed after this process is complete. Not all punctuation is removed, only erroneous punctuation. The remaining tokens are then all converted to lower case to reduce the number of distinct tokens. See fig. 1.

D. Scoring Variations on Sentiment Analysis

Table I shows the preprocessing pipelines each review is passed through. So for *each* review in the corpus, there are 14 variants, each then passed to RoBERTa and VADER.

The RoBERTa sentiment analyser returns a individual scores for negative, neutral, and positive sentiment, normalised to sum to 1. Positive and negative values measure sentiment and neutral signifies no sentiment, so $negative{+}positive$ is used as a score for the magnitude of detected sentiment by RoBERTa, with a domain of [0,1].

The VADER model's `polarity_scores()` function returns a value in the domain [-1,1] as a "normalized,

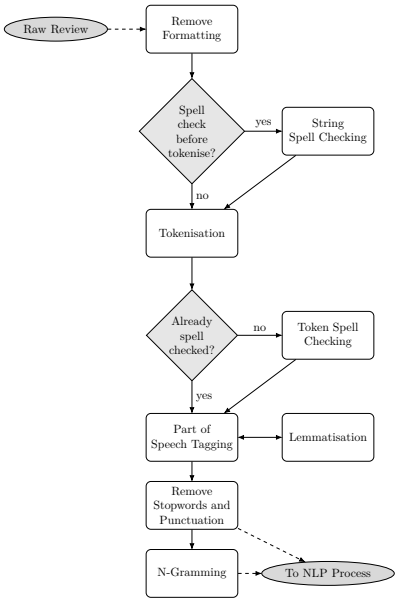

Fig. 1: New order of preprocessing tools

Review	Lemmatised	Spell Checker	RoBERTa Result	VADER Result
Review 1	True	None	*x*	*y*
	True	TextBlob	*x*	*y*
	True	TextBlob_Custom	*x*	*y*
	True	SymSpell	*x*	*y*
	True	SymSpell_Custom	*x*	*y*
	True	SymSpell_Compound	*x*	*y*
	True	SymSpell_Compound_Custom	*x*	*y*
	False	None	*x*	*y*
	False	TextBlob	*x*	*y*
	False	TextBlob_Custom	*x*	*y*
	False	SymSpell	*x*	*y*
	False	SymSpell_Custom	*x*	*y*
	False	SymSpell_Compound	*x*	*y*
	False	SymSpell_Compound_Custom	*x*	*y*
...	...	...	...	...
Review *n*	...	...	*x*	*y*

TABLE I: Preprocessing score outputs

weighted composite score" (recommended option) from the negative, neutral and positive scores [7]. As we only need the magnitude of detected sentiment (not polarity), the absolute value of the score is used for a domain of [0,1].

E. Choosing A Statistical Test

A 'Two Way ANOVA *without* replication' is used for comparison of means. The 'Tukey' post hoc test was chosen as the sizes for each spell checker are the same.

a) Null Hypothesis ($H_0 : Sig. \geq 0.05$): That preprocessing yields equal sentiment to no preprocessing, and that all spell checkers yield equal sentiment.

b) Alternate Hypothesis ($H_1 : Sig. < 0.05$): That preprocessing does not yield equal sentiment to no preprocessing, and all spell checkers do not yield equal sentiment.

V. Results

A. Analysis of Variance (ANOVA) for RoBERTa

If discounting spell checker choices, preprocessing (or not) significantly affected sentiment detection scores. When we ignore whether the review was preprocessed or not, the type of spell checker used influenced the sentiment detection score. The effect of spell checking on reviews differed for preprocessing versus not preprocessing.

Model	Source	F	Sig.
RoBERTa	Preprocessed	15486.666	.000
	Spell Checker	1456.351	.000
	Preprocessed * Spell Checker	47.219	<.001
	Total	3583068.969	6134198
VADER	Preprocessed	456.437	<.001
	Spell Checker	402.653	.000
	Preprocessed * Spell Checker	.133	.992
	Total		

TABLE II: Between-Subjects Effects.

B. Analysis of Variance (ANOVA) for VADER

If discounting spell checker choices, preprocessing (or not) significantly affected sentiment detection scores. When we ignore whether the review was preprocessed or not, the type of spell checker used influenced the sentiment detection score. The effect of spell checking on reviews did not differ for preprocessing versus not preprocessing.

VI. Discussion & Evaluation of Results

A. RoBERTa

Looking at the results of the statistical analysis in table II, it is clear that RoBERTa performs better (detects more sentiment) with **no** preprocessing. It is also clear that RoBERTa is better with **no** spell checking.

Spell checking has a significant effect on detecting sentiment of reviews for mental health apps, whether the review is preprocessed or not. But, when the reviews are *not* preprocessed, more sentiment is detected. This pattern of results is evident from the graph in fig. 2. This shows that when using 'RoBERTa' to detect sentiment, there is no advantage to performing lemmatisation or spell checking, which is computationally expensive. Doing so, will in fact *reduce* the amount of sentiment detected.

B. VADER

The VADER model performs better **with** preprocessing. VADER is also better **with** spell checking.

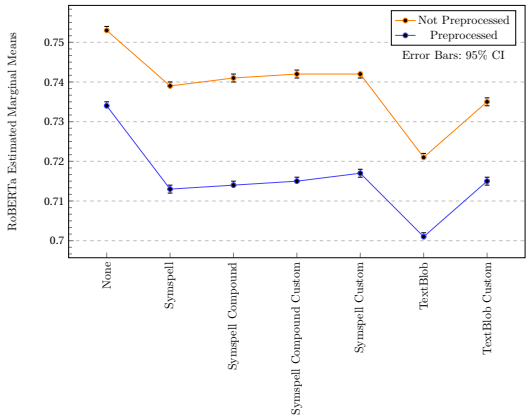

Fig. 2: RoBERTa Estimated Marginal Means Graph

In summary, spell checking has an effect on detecting sentiment of reviews for mental health apps, whether the review is preprocessed or not. But, when the reviews *are* preprocessed, more sentiment is detected, as shown in fig. 3. Hence when using 'VADER' to detect sentiment, there is a significant difference in the amount of sentiment detected when the reviews are lemmatised and spell corrected using a compound spell checker. Although the result is significant, it is marginal. So, if accuracy is important, performing these steps will help. If speed is prioritised, consider if these extra operations are beneficial.

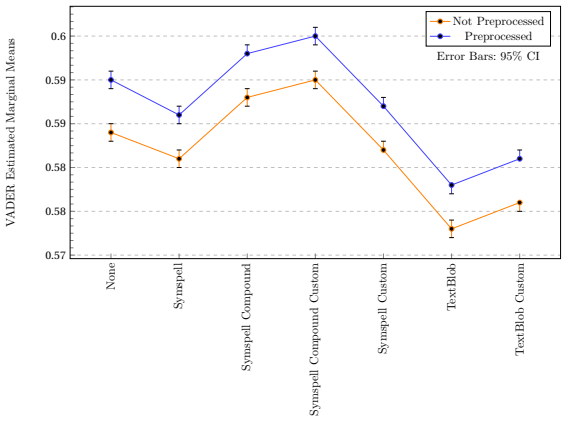

Fig. 3: VADER Estimated Marginal Means

VII. Future Work

We acknowledge that the criteria in this research being tested is narrow. As the results for each analyser show opposing recommendations, future work on experimenting with each stage of the preprocessing tools flowchart (Fig. 1) and whether the computational time required is worth the increase (or possible decrease) of detected sentiment.

VIII. Conclusion

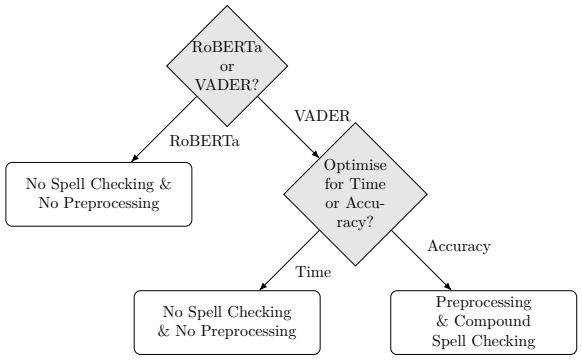

Fig. 4: Recommended Processes Flowchart

Current preprocessing guidelines for common NLP tasks such as Sentiment Analysis (SA) do not fully account for challenges posed by the evolution of English language usage online. This work studies the effects of various preprocessing and spell checking techniques applied to sentiment detection on a dataset of app reviews. The amount of sentiment detected by each combination was scored and statistically analysed.

Our results show a difference in how to approach using lexicon based approaches to SA, versus ML approaches.

Our results contribute new guidelines for preprocessing online reviews, based on computational time and resource costs to extract the maximum amount of sentiment. We provide a recommended workflow in fig. 4.

Statistical analysis showed when using a leading ML model for SA (RoBERTa), performing preprocessing or spell checking in fact *reduces* the amount of sentiment detected. In contrast, when using a leading lexicon based model for SA (VADER), performing preprocessing and a compound spell checker would *increase* the amount of sentiment detected, by a small but statistically significant amount. Hence, different strategies are needed, based on SA approach and whether accuracy or time is prioritised.

References

[1] F. B. Gomes, J. M. Adán-Coello, and F. E. Kintschner, "Studying the effects of text preprocessing and ensemble methods on sentiment analysis of brazilian portuguese tweets," in *International Conference on Statistical Language and Speech Processing*. Springer, 2018, pp. 167–177.

[2] C. P. Chai, "Comparison of text preprocessing methods," *Natural Language Engineering*, vol. First View, p. 1–45, 2022.

[3] J. Androutsopoulos, Ed., *Mediatization and Sociolinguistic Change*. Berlin, Boston: De Gruyter, 2014.

[4] Shahnawaz and P. Astya, "Sentiment analysis: Approaches and open issues," in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, 2017, pp. 154–158.

[5] F. L. Dos Santos and M. Ladeira, "The role of text preprocessing in opinion mining on a social media language dataset," in *2014 Brazilian Conference on Intelligent Systems*, 2014, pp. 50–54.

[6] A. A. M. Shoeb and G. de Melo, "Assessing emoji use in modern text processing tools," 2021. [Online]. Available: https://arxiv.org/abs/2101.00430

[7] C. J. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text." in *ICWSM*. AAAI, 2014.

[8] 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.

[9] E. Clark and K. Araki, "Text normalization in social media: Progress, problems and applications for a pre-processing system of casual english," *Procedia - Social and Behavioral Sciences*, vol. 27, pp. 2–11, 2011.

[10] A. Tolegenova, "Automatic error correction," *Suleyman Demirel University Bulletin: Natural and Technical Sciences*, vol. 58, no. 1, pp. 15–21, 2022.

[11] A. Bittlingmayer. (2019) Amazon reviews for sentiment analysis. [Online]. Available: https://www.kaggle.com/datasets/bittlingmayer/amazonreviews

[12] R. Kulkarni. (2019) Urban dictionary words and definitions. [Online]. Available: https://www.kaggle.com/datasets/therohk/urban-dictionary-words-dataset

[13] B. Liu, *Sentiment analysis: Mining opinions, sentiments, and emotions*, 2nd ed. Cambridge university press, 2022.

[14] J. Foster, O. Cetinoglu, J. Wagner, J. L. Roux, S. Hogan, J. Nivre, D. Hogan, and J. van Genabith, "#hardtoparse: Pos tagging and parsing the twitterverse," *AAAI Workshop*, 2011.

[15] T. Bergmanis and S. Goldwater, "Context sensitive neural lemmatization with Lematus," in *Proceedings of 2018 ACL*. New Orleans, Louisiana: ACL, 2018, pp. 1391–1400.