Studying the Effect of Feature Extraction, Annotation and Preprocessing Methods for Supervised Twitter Sentiment Classification

Anonymous ACL submission

Abstract

Sentiment classification has become a popular automated method for extracting sentiment from online text. In this paper we explore this topic, and evaluate how a range of techniques employed prior to classification can enhance performance. Using well-known machine learning methods, notably Naïve Bayes, Logistic Regression and Linear Support Vector Machines, we conduct a thorough experimental analysis of the combinations of different preprocessing, annotation and feature extraction methods using these models, including a comparison of the effect each combination has on sentiment classification performance. We conclude by identifying the models and their hyperparameters which yield high validation accuracies following cross-validation and evaluate model performance on test datasets. We find the use of lemmatisation during the annotation phase generally weakens performance compared to when it is not used, and we find including stopwords results in higher performance compared to when they are removed.

1 Introduction

Sentiment analysis (opinion mining) is the computational study of opinions, sentiments and emotions represented in text (Indurkhya and Damerau, 2010). This arose from the need to automate the extraction of opinions hidden in huge volumes of online text. Sentiment analysis can be treated as a supervised classification problem, in which we attempt to classify opinionated documents (e.g. product reviews) as expressing positive or negative opinions. In contradistinction to topic-based text classification, where topic-related words aid in the identification of topics, sentiment classification aims to classify text based on the sentimental polarities of the opinions that the text contains. Multi-class sentiment classification is also possible, in which we attempt to extract sentiment from text and classify into ≥ 3 output classes.

The performance of sentiment classification is heavily dependent on the choice of preprocessing methods carried out on the text. The inherent nature of online text can present many challenges, and the choice of preprocessing methods is highly problem dependent. When concerning textual 'features', determining what is considered a feature is of paramount importance in classification problems, as this information is what is input to machine learning models. Similarly to preprocessing, choosing the feature extraction methods which result in high performance is problem dependent and should be experimented with accordingly. In addition to feature extraction, annotation methods can include additional information to text which may or may not be useful in determining sentiment.

Our main research question is the following: How do different preprocessing, annotation and feature extraction methods affect sentiment classification results? To answer this question we implement and evaluate a set of standard machine learning models which are: Multinomial and Bernoulli Naïve Bayes, Logistic Regression and Linear Support Vector Machines. Following hyperparameter tuning, an extensive evaluation of cross-validation and individual model performance is given.

One source of opinionated text which is abundant and easily available is Twitter [1]. User tweets can be queried by positive or negative sentiment through the Twitter API and downloaded to produce a dataset for classification. Analysing online text from Twitter presents various challenges for sentiment classification due to the inherent nature of the text. Misspellings, slang words and informal language provide complications for automated methods, alongside further issues such as mentions, hashtags and punctuation which must be dealt with appropriately during the preprocessing phase. In this paper we utilise a dataset of pre-labelled tweets with a dichotomous sentiment variable, that is, a variable which takes one of 2 values, positive or negative.

For each preprocessing and feature extraction method studied, we provide a comprehensive analysis of the impact each one has on classification performance in terms of standard evaluation metrics, as well as grid search for hyperparameter tuning.

[1] https://www.twitter.com

2 Related Work

Fouad et al. (2018) considered the Sentiment140 studied in this paper on a sub-corpus of 3000 tweets. They considered Bag of Words, Part of Speech (POS), emoticon and lexicon-based feature sets, and evaluated performance using the SVM, Naïve Bayes and Logistic Regression classifiers. They performed unified sampling due to computational limitations, and constructed two smaller corpora of tweets containing 1000 and 3000 tweets respectively. They discovered that the inclusion of all feature sets resulted in higher accuracy scores for the set of 1000 tweets, with a marginal increase in accuracy observed when emoticon features are removed for the set of 3000 tweets. They state that lexicon-based features and POS enhance classification performance when combined with the Bag of Words feature set with emoticons not proving significant. They also discover that using the information gain (IG) feature selection technique enhanced classification accuracy across the classifiers by roughly 15% on average, as well as reducing dimensionality of the feature vector. Interestingly, a Majority Voting Ensemble method was found to slightly outperform the individual classifiers, although on the set with 3000 tweets, the SVM outperformed the ensemble.

The original paper detailing the procedure that was used to obtain the data used in this paper also conducted sentiment analysis on the full dataset (Go et al., 2009). They yielded accuracies of roughly 80% for Maximum Entropy, Naïve Bayes and SVM classifiers. This gives a rough indication of what metrics we should expect to achieve in our analysis, however our dataset is downsampled (see section 3 for further details). They also discovered using part-of-speech tags were not useful in predicting sentiment.

Bao et al. (2014) explored the use of preprocessing methods for Twitter sentiment classification, and found both stemming and lemmatisation reduce classification accuracy, and accuracy increases when URL features, negation transformation and repeated letters normalization are employed. Asghar et al. (2014) provides a comprehensive overview of existing preprocessing and feature extraction methods used in sentiment analysis. Part of speech tagging, stemming, lemmatisation and removal of stop words can be carried out during the preprocessing phase, and facilitates feature extraction and eliminating noise. Whether some or all of these techniques should be performed is highly problem dependent, and should be considered appropriately.

Gautam and Yadav (2014) conducted an interesting analysis on a dataset of $\sim$ 20,000 tweets for customers' review classification. Their methodology involved extracting adjectives to determine sentiment and using WordNet to find synonyms with the unigram model, as well as using the naïve bayes, support vector machines and maximum entropy classifiers. An accuracy of 89.9% was achieved using WordNet for semantic analysis. Pang et al. (2002) also conducted a similar experiment using unigram and bigram models, with the inclusion/exclusion of part-of-speech tags. A notable outcome of their research was when accounting only for feature presence using SVM's, they observed an accuracy of 82.9% compared to feature frequencies (72.8%) under the unigram model. They also found the inclusion of part-of-speech tags did not affect the results significantly.

Lilleberg et al. (2015) conducted research using word2vec to convert words into vector representations, with the idea of introducing additional semantic features which facilitate text classification. They find using word2vec with TF-IDF outperforms both methods individually due to the complementary features provided by word2vec. Acosta et al. (2017) also explored the use of word2vec for sentiment analysis, and obtained an accuracy of 72% using an SVM trained on 10000 tweets and evaluated on a test set of 4000 tweets.

Research carried out by Bindal and Chatterjee (2016) involved a lexicon-based sentiment scoring method. They used the Point-wise Mutual Information (PMI) metric to score texts with positive or negative polarity, which were then fed to an SVM classifier. Glorot et al. (2011) propose an unsupervised approach using deep learning to extract sentiment from a set of Amazon product reviews, and show the classifiers trained on this high-level representation of features outperform state-of-the-art methods and scale to large datasets for the domain adaptation problem.

3 Data

For our experiments we chose to work with a pre-labelled corpus of Twitter data known as the Sentiment140 corpus [2]. Tweets were obtained by keyword search using the Twitter Search API. Using this automated approach, a labelled dataset of tweets was constructed automatically without the need for human annotation (Go et al., 2009). The automated procedure that was used to obtain the data proves to be a particularly advantageous one; manual human labelling of large datasets is a very time consuming and tedious task, albeit a necessary one when concerning supervised machine learning methods as they rely on labelled data. This data is suitable to conduct experiments on due to its size and wide language variety.

The data consists of 1.6 million instances (tweets) collected between April 6th 2009 to June 25th 2009, with a balanced distribution of both positive and negative sentiment labels (800k tweets for each sentiment class). A tweet is characterised by the text it contains alongside its polarity (i.e. its sentiment). Meta information is also included, such as the user that posted the tweet and the date it was posted. Tweets are labelled based on the emoticon they contain. Tweets containing any of the emoticons :), :), :-), :), :D and =) are classified as positive sentiment, with emoticons :(, :-(and

[2]The corpus is available in CSV format at http://help.sentiment140.com/for-students

: (classified as negative sentiment. These emoticons were then removed, as their presence can result in lower classification metrics for certain classifiers such as the Support Vector Machine, a commonly used algorithm used in NLP, studied later in this report.

Due to the size of the dataset, it will be necessary to scale down the size to a more reasonably sized sub-corpus, thus reducing computational time when evaluating different methods and preventing any potential memory issues. This provides a potential limitation, given that we are reducing the amount of tweets and the vocabulary set which represents it. The sampling approach used is discussed further in section 4. Another limitation is that the data is based on English tweets only and does not account for tweets with different languages. Hence the evaluation metrics we provide in our analysis are only evaluated using English tweets and will not necessarily perform similarly with other languages. A further limitation may be that tweets labelled as having positive or negative sentiment may actually express neutral sentiment, hence it may be necessary to include a neutral sentiment class to represent these tweets, although one would have to re-label a significant number of tweets to obtain a corpus suitable for supervised learning.

4 Methodology

We used the Python programming language and Jupyter notebooks to conduct our experiments. The machine learning library of choice for this research was scikit-learn [3], which provides a comprehensive set of tools to facilitate development of machine learning pipelines. We use the default tokeniser provided by the scipy library [4], as well as their lemmatisation and part-of-speech functionality, as well as incorporating their default list of English stopwords.

Our experimental workflow is summarised in the schematic diagram given in Figure 1. We began by downloading the Sentiment140 dataset into memory programmatically, extracting the text and polarity attributes of the data excluding all other attributes. We then randomly sub-sample the data to a total of 100k tweets, with a balanced class distribution (i.e. half of tweets have negative sentiment, with the other half having positive sentiment).

4.1 Data Preprocessing

We then proceeded to the data preprocessing phase, in which we performed the following:

- Replacing hashtags (e.g. text), user mentions (e.g. @user) and URLs with the tokens HASHTAG, MENTION and URL respectively
- Replacing contractions with their corresponding expanded forms (e.g. can't is replaced by can not)

[3] scikit-learn library: https://scikit-learn.org/stable/

[4] spacy library: https://spacy.io/

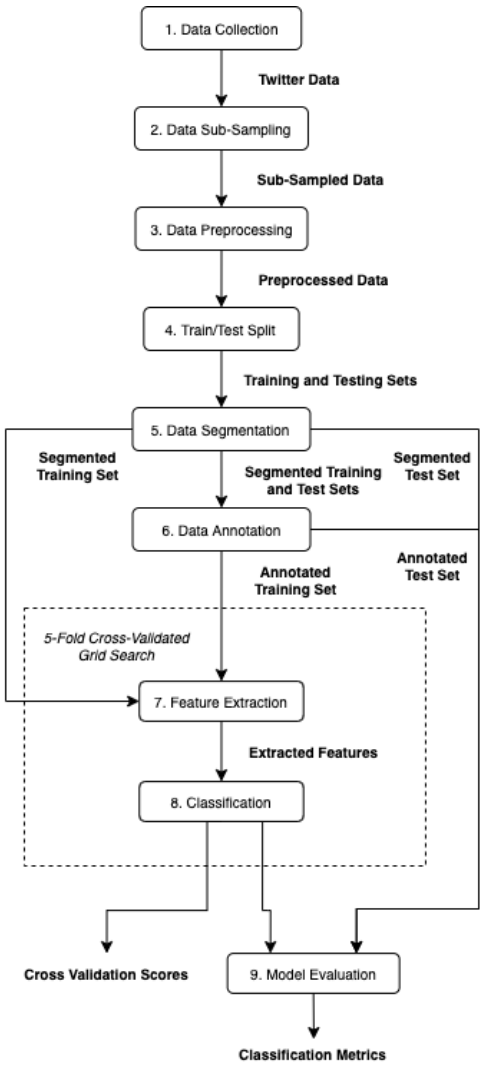

Figure 1: Experimental workflow.

- Removing any emojis or additional emoticons which have not already been removed
- Fixing any Unicode encoding mistakes and removing special characters such as 'smart quotes'
- Removing any punctuation
- Lower-casing the text

Notice here we perform all previously mentioned data preprocessing techniques before fitting a classification pipeline - this prevents any unnecessary preprocessing computation on each iteration of cross-validated grid search, hence improving experimental efficiency. The inclusion/exclusion of stopwords are also considered during preprocessing, however we analyse the effect of their presence using grid search later in the pipeline. Stopwords are words such as "and", "it", "at"; depending on the context these words may or may not be informative in representing content in text, hence we study the effect of their inclusion and exclusion on classification performance. We then partitioned the preprocessed

data into training and testing sets, with a 90%:10% split respectively.

4.2 Data Segmentation, Annotation and Feature Extraction

We then performed data segmentation, in which we performed tokenisation using spacy to tokenise tweets into words. Data annotation was then performed - in our analysis we studied the inclusion/exclusion of the following annotation methods:

- **POS tags**: labels assigned to words in text which indicate part of speech (e.g. nouns, adjectives, adverbs etc). We use the POS tagger provided by spacy to facilitate this.
- **Lemmatisation**: the process of grouping different inflected forms of a word so they can be analyzed as a single item (e.g. singing becomes sing after lemmatising). Again we use spacy to facilitate this.

Each of these methods will be subjected to 5-fold cross-validated grid search, a form of hyperparameter tuning, to obtain near-optimal hyperparameter values for each of the 4 models studied (see section 4.3). 5 validation scores will be returned and their means will be calculated. This will enable a thorough comparison between all models, fitted with near-optimal hyperparameters, in terms of their mean validation accuracies. Using grid search, we can then evaluate the inclusion/exclusion of the following feature extraction methods:

- **Bag-of-words**: a simple collection of words along with their counts, disregarding the order in which they appear.
- **Bag-of-n-grams**: a model similar to bag-of-words, however text is now represented as an unordered collection of n-grams.
- **TF-IDF**: a numerical statistic intended to reflect how important a word is to a piece of text in a collection of texts.
- **Adjectives and Adverbs**: a custom feature set consisting of only adjectives and adverbs present in each piece of text for a collection of texts.

When we study TF-IDF, this method removes words with no semantic value, hence stopwords will be removed anyway. In summary, the following experiments will be carried out:

1. Bag of unigrams (with/without lemmatisation)
2. Bag of unigrams (with POS tags, with/without lemmatisation)
3. Bag of bigrams (with/without lemmatisation)
4. Bag of trigrams (with/without lemmatisation)
5. Bag of unigrams and bigrams (with/without lemmatisation)
6. Bag of unigrams, bigrams and trigrams (with/without lemmatisation)
7. TF-IDF model (with/without lemmatisation for different n-gram ranges)
8. Adjectives and Adverbs (without lemmatisation)

For each of the experiments, we consider how including and removing stopwords affects performance, as well as considering feature presence and frequencies. This will enable the different preprocessing, feature extraction and annotation methods to be compared and analysed in terms of mean validation accuracies, and evaluated on their respective test sets using accuracy, precision, recall and F1 scores. Studying the effect each of the preprocessing techniques has on classification performance using grid search enables techniques to be identified which yield high validation set performance.

4.3 Classification Models

We experimented with the following supervised machine learning methods: Multinomial and Bernoulli Naïve Bayes, Linear Support Vector Machine and Logistic Regression. These classifiers are known to work well with natural language-based tasks so will be suitable for our analysis. The relevant annotated and preprocessed data for a given experiment is then used as training data to each of these classifiers. Upon completion of cross-validated grid search, we obtain cross validation accuracy scores for the set of hyperparameters and the model fitted on these hyperparameters. Since our dataset has no class imbalance, we choose accuracy as our evaluation metric. The former allows for comparison between preprocessing and feature extraction methods following cross-validation, with the latter facilitating understanding of classification performance using these methods on an unseen dataset.

5 Results

5.1 Grid Search Evaluation

Firstly, we present the results of the previously described experiments for the different feature sets, annotation strategies and preprocessing methods for all classifiers using 5-fold cross-validated grid search. For brevity, only the results yielding the highest validation accuracies following grid search are presented. Accuracies in Table 1 with bold font indicate this classifier achieved the highest validation accuracy across all classifiers for a given set of hyperparameters. Accuracies marked with asterisks indicate this score, when presence of features was considered, was marginally worse compared to feature frequencies (in the order of 0.01%). The inclusion/exclusion of stopwords are given by SW (stopwords) and NSW (no stopwords) respectively. In the

Features	Lemma?	LR		MNB		BNB		SVM	
		SW	NSW	SW	NSW	SW	NSW	SW	NSW
Bo/Unigrams	Y	**77.23**	74.36	76.03	73.95	75.65	73.80	75.36	72.74
Bo/Unigrams	N	**77.83**	74.88*	76.53	74.32	76.10	74.25	75.86	73.29
Bo/Unigrams (POS)	Y	**76.99**	74.16*	75.89	73.90	75.47	73.78	75.07	72.54
Bo/Unigrams (POS)	N	**77.55**	74.59	76.28	74.11	75.79	74.12	75.53	72.86
Bo/Bigrams	Y	**75.30***	67.76	74.83	68.36	74.94	68.36	73.78*	66.58*
Bo/Bigrams	N	**75.33***	65.79*	74.93	66.60	75.10	66.59	74.05	65.20*
Bo/Trigrams	Y	**70.37**	58.85*	69.19	59.05	70.20	59.03	69.59	58.43*
Bo/Trigrams	N	**69.64**	55.23	68.43*	55.14	69.50	55.14	69.06	55.16
Bo/Uni+Bi	Y	**78.73**	74.75	77.35	73.82	76.84	73.69	76.84	72.97
Bo/Uni+Bi	N	**79.14**	75.27	77.74	74.26	77.30	74.15	77.60	73.94
Bo/Uni+Bi+Tri	Y	**79.13**	74.96*	77.54	73.62	76.47	73.45	77.74	73.46
Bo/Uni+Bi+Tri	N	**79.39**	75.35*	77.77	74.11	76.67	73.98	78.24*	74.02*
TF-IDF (Uni+Bi+Tri)	Y	**79.13**	74.96	77.54	73.62*	76.47	73.45	77.74	74.46
TF-IDF (Uni+Bi+Tri)	N	**79.39**	75.35	77.77*	74.11*	76.67	73.98	78.24*	74.02
Adjs+Adv (Bo/Uni)	N	64.24*	61.42*	62.06	60.89	**64.28**	61.35*	64.03	61.22*
Adjs+Adv (TF-IDF)	N	**64.28***	61.35	62.06	60.89	**64.28**	61.35	64.03	61.22

Table 1: 5-fold cross-validation mean accuracies for different feature sets, annotation strategies and preprocessing methods for different classifiers.

table we present accuracies when the presence of features was considered in the classification pipeline. We only report results for TF-IDF when unigrams, bigrams and trigrams are considered as they achieve the best results compared to different n-grams. Often we observe higher accuracies with only presence of features compared to feature frequencies. We generally notice across all feature extraction, annotation and preprocessing methods, Logistic Regression is the highest performing model. Interestingly, lemmatisation has a negative impact on validation accuracy when it is included. A simple bag-of-words model considering counts of unigrams, bigrams and trigrams without lemmatisation yielded the best result (79.39%) when stopwords were included for the Logistic Regression classifier. When TF-IDF is considered for unigrams, bigrams and trigrams, Logistic Regression also achieved the same result. Across the classifiers, the effect stopwords have on validation accuracy is significant - in most cases, validation accuracies tend to be a few percentage points higher when stopwords are maintained in the text. We find that using POS tags for the unigram model performs worse when lemmatisation is considered or not. Interestingly, it was found extracting adjectives and adverbs from each tweet using both the bag-of-words and TF-IDF methods achieved poor results across the classifiers, with peak accuracy reaching only 64.28% for both Logistic Regression and Bernoulli Naïve Bayes respectively.

5.2 Model Evaluation

Following this experiment, we conducted model evaluation for the highest performing models and hyperparameters in terms of cross-validated accuracies:

1. Logistic Regression: **79.39%** (without lemmatisation, including stopwords, for bag of unigrams, bigrams and trigrams, and presence of features)
2. Support Vector Machine: **78.24*%** (without lemmatisation, including stopwords, using TF-IDF with bag of unigrams, bigrams and trigrams, presence of features)
3. Bernoulli Naïve Bayes: **77.30%** (without lemmatisation, including stopwords, for bag of unigrams and bigrams, and presence of features)
4. Multinomial Naïve Bayes: **77.77*%** (without lemmatisation, including stopwords, using TF-IDF with bag of unigrams, bigrams and trigrams)

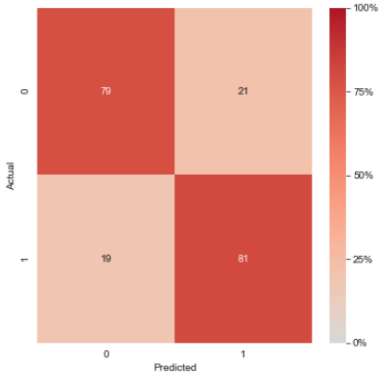

Figure 2: Logistic regression confusion matrix.

The results are shown in Table 2. Logistic Regression yielded the largest F1 score compared to other models for both negative and positive sentiment tweets, hence is the best overall model. We notice that the Linear Support Vector Machine performs surprisingly better in terms of accuracy when compared to other models at

Model	Accuracy		Precision		Recall		F1		Support
	Neg.	Pos.	Neg.	Pos.	Neg.	Pos.	Neg.	Pos.	
Logistic Regression	78.64	**80.72**	**80.00**	79.00	79.00	**81.00**	**79.00**	**80.00**	10000
Bernoulli Naïve Bayes	75.36	79.52	79.00	76.00	75.00	80.00	77.00	78.00	10000
Multinomial Naïve Bayes	83.38	72.12	75.00	81.00	83.00	72.00	**79.00**	76.00	10000
Support Vector Machine	**84.44**	71.92	75.00	**82.00**	**84.00**	72.00	**79.00**	77.00	10000

Table 2: Model evaluation for highest performing models following 5-fold cross-validated grid search for positive and negative sentiment tweets with 5000 tweets in each class (values for precision, recall and F1 have been rounded automatically using scikit-learn).

predicting negative sentiment tweets. Figure 2 provides a heatmap visualisation of the final confusion matrix for Logistic Regression using percentage values. We notice when tweets are predicted incorrectly, they are often predicted as positive sentiment (21%).

6 Findings

We can summarise our findings as follows:

- In general, lemmatisation has a negative effect on validation accuracies for bag of words, TF-IDF and custom feature sets. This may seem somewhat contradictory, given that reducing a set of words to their inflected forms would reduce dimensionality of the vocabulary used and summarise similar words to a common representation which would aid in predicting sentiment, however this was found not to be the case.
- Including stopwords tends to result in higher validation accuracies when different annotation and feature extraction methods are considered for a given classifier.
- The bag of unigrams model performs better when compared to bag of bigrams and bag of trigrams models in terms of validation accuracies.
- Adjectives and adverbs alone perform significantly worse when compared to bag of words, bag of n-grams and TF-IDF models containing all words. Interestingly, this does not align with the research conducted by Gautam and Yadav (2014) given they observe significantly higher accuracies when only the presence of adjectives are considered.
- In general, validation accuracies observed for both presence and feature frequencies yield similar results, with feature presence performing slightly better in the majority of cases, although difference in accuracies obtained is only marginal.
- Annotating tokens with POS tags results in marginally worse validation accuracies.
- Logistic Regression yields the greatest validation accuracies across different feature extraction, annotation and preprocessing methods when compared to the other classifiers.

7 Conclusions and Future Work

In this paper we studied how various combinations of preprocessing, feature extraction and annotation methods influence sentiment classification performance. We discovered that for the case of Twitter data when using a corpus of 100k tweets, removing stopwords tends to result in higher validation accuracies for common machine learning methods. This is an interesting result, given that stopwords tend to be uninformative for classification purposes, however we find it is not always necessary to remove them. We also found the presence of features performed marginally better than considering feature frequencies, although the difference was not significant. Lemmatising the tweets was found to reduce validation accuracies across all models. Considering a feature set of only adjectives and adverbs performed worse when individual tokens and n-grams were considered.

We envisage future work could involve the exploration of word embeddings, which would aid in the representation of similar words. Given the set of tweets were posted in 2009, classification accuracies we observed from our analysis will not necessarily be consistent with current language present in tweets today, hence a more up-to-date set of data would enable a more representative set of evaluation metrics to be obtained. The use of univariate feature selection mechanisms to further reduce the number of features used in the classification pipeline could also be explored. The value of k, the number of top features to retain according to an appropriate statistical measure such as the χ^2 test, can then be tuned using grid search. Given we only studied a single stopwords list, one could also investigate using different stopword lists to determine whether the removal of different stopwords is significant. The Twitter data analysed in this paper was restricted to the English language only, hence one could investigate classifying sentiment in tweets for different languages. Finally, one could explore more sophisticated machine learning models, such as deep neural networks or transformers, compared to the standard methods studied in this paper that are known to work well for NLP tasks.

References

Joshua Acosta, Norissa Lamaute, Mingxiao Luo, Ezra Finkelstein, and C Andreea. 2017. Sentiment analysis of twitter messages using word2vec. *Proceedings of Student-Faculty Research Day, CSIS, Pace University*, 7.

Muhammad Zubair Asghar, Aurangzeb Khan, Shakeel Ahmad, and Fazal Masud Kundi. 2014. A review of feature extraction in sentiment analysis. *Journal of Basic and Applied Scientific Research*, 4(3):181–186.

Yanwei Bao, Changqin Quan, Lijuan Wang, and Fuji Ren. 2014. The role of pre-processing in twitter sentiment analysis. In *International conference on intelligent computing*, pages 615–624. Springer.

Nimit Bindal and Niladri Chatterjee. 2016. A two-step method for sentiment analysis of tweets. pages 218–224.

Mohammed M. Fouad, Tarek F. Gharib, and Abdulfattah S. Mashat. 2018. Efficient twitter sentiment analysis system with feature selection and classifier ensemble. In *The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018)*, pages 516–527, Cham. Springer International Publishing.

G. Gautam and D. Yadav. 2014. Sentiment analysis of twitter data using machine learning approaches and semantic analysis. In *2014 Seventh International Conference on Contemporary Computing (IC3)*, pages 437–442.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing*, 150.

Nitin Indurkhya and Fred J. Damerau. 2010. *Handbook of Natural Language Processing*, 2nd edition. Chapman Hall/CRC.

Joseph Lilleberg, Yun Zhu, and Yanqing Zhang. 2015. Support vector machines and word2vec for text classification with semantic features. In *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 136–140. IEEE.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, page 79–86, USA. Association for Computational Linguistics.