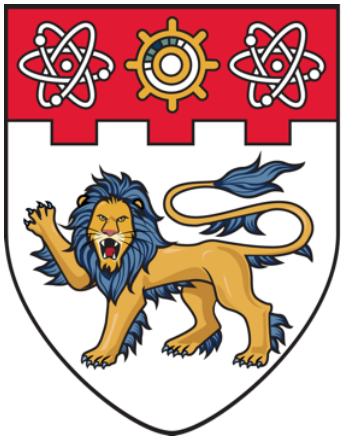

NANYANG TECHNOLOGICAL UNIVERSITY

SINGAPORE

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

Final Year Project

SCE17-0595

Final Year Project Report

# Developing an Affect Intensity Tool by Combining Lexicon and Learning Based Approaches


Author:

Derrick Peh Jia Hao

Student ID:

U1621219F

Supervisor: Dr Althea Liang Qianhui



Submission Date: 21 / 10 / 2018

# Abstract

The implementation of sentiment analytical tools is largely centred between the use of lexicons resources and machine learning algorithms that are commonly used for general classification tasks. The results are commonly classified based on its polarities and are being engaged by a wide range of applications in the recent years. This report provides an insight of a comparison between the two approaches, and is compared with the hybrid approach which is developed in the report. The results are demonstrated on a competition task hosted by the International Workshop on Semantic Evaluation where the intensity of a different spectrum of emotions is evaluated. When run on the given datasets provided by the organisers, the results indicate that using lexicons resources alone performs poorly, especially when a general lexicon resource is selected. Traditional machine learning algorithms have been assessed as better candidates to produce better results however at the cost of expending more time to annotate the data. The hybrid approach which is the combination of both approaches has proven to yield better results with lesser costs involved than the implementation of individual approaches.

# Table of Contents

## Chapter 1

Introduction ..... 1

1.1 Problem Definition ..... 1
1.2 Research ..... 2
1.3 Hypothesis ..... 3
1.4 Motivation ..... 3
1.5 Limitations ..... 4
1.5.1 Scope ..... 4
1.5.2 Neural Networks vs Traditional Algorithms ..... 5
1.6 Structure of Report ..... 6

## Chapter 2

Related Works ..... 7

2.1 Lexicon-based Approaches ..... 7
2.2 Learning-based Approaches ..... 8
2.3 Hybrid Approach ..... 9

## Chapter 3

Implementation ..... 10

3.1 Programming Language and Frameworks ..... 10
3.2 Data Pre-processing ..... 11
3.2.1 Cleaning and arranging the data ..... 11
3.2.2 Pre-processing of data ..... 13
3.2.3 Feature Generation and Selection ..... 15
3.3 Lexicon-based method ..... 22
3.4 Learning-based method ..... 23
3.5 Hybrid method ..... 25

## Chapter 4

Results ..... 27

4.1 Lexicon-Based Approach ..... 27
4.1.1 Table Comparison of lexicon resources ..... 28
4.2 Learning-based Approach ..... 29
4.2.1 Table Comparison of learning-based classifiers ..... 29
4.3 Hybrid Approach ..... 30
4.3.1 Table Comparison on different combinations of pre-processors ..... 30
4.3.2 Table Comparison on different combinations of various learning classifiers ..... 31
4.4 Scoreboard ..... 31
4.4.1 Baseline Scores by Organisers ..... 32
4.4.2 Submission Score ..... 32
4.4.3 Top Scorers ..... 33
4.4.3 Evaluation of top scorers ..... 33

## Chapter 5

Discussion ..... 34

5.1 Lexicon Resources ..... 34
5.2 Learning-based Algorithms ..... 36
5.3 Hybrid Approach ..... 38
5.4 Advantages of different models ..... 39
5.5 Possible Improvements ..... 40
5.6 Conclusion ..... 41

## Chapter 6

Appendix ..... 42

Complied Results ..... 42
Setup ..... 45
Prerequisites ..... 45
Instructions ..... 46
Source Codes ..... 46
Bibliography ..... 47

# Chapter 1

# Introduction

Classification is a supervised learning approach where the computer program automates the process of predicting a discrete class label based on the properties or features of the data given. Regression follows a similar process as classification, but predicts a continuous quantity as an output instead of a discrete class. In the field of sentiment analysis, these kinds of predictive modelling problems are branched into two main methodologies - the lexicon-based approach and learning-based approach. They are the building blocks for creating a sentiment model to predict the different emotions in texts.

The lexicon-based approach is conducted by analysing the words from a text that are tokenised and then comparing against with a set of predefined or manually-created lexicon. The polarity of the word that is extracted from the comparison will describe how positive or negative a word is.

The learning-based approach is conducted by training a machine learning algorithm on a training data set. Every text in the training data set is pre-labelled with a defined set of parameters before it is run against a classifier. The classifier analyses the training data and evaluates similar patterns to produce a prediction.

## 1.1 Problem Definition

The learning-based approach generally outperforms the lexicon-based approach [1] at the cost of being more time and resource intensive. In specific, using learning-based approach requires data to be manually classified before it is handled by the classifier model. According to various studies[2] [3], results produced from lexicon-based approach gave a high accuracy but produced a low recall. This means that the lexicon-based approach can capture and classify

datasets with minimal errors – lower rate of false positive and higher rate of true positive, however being constrained to specific domains – where a high rate of false negatives is not being captured.

In the case of learning-based approach, neural networks and traditional algorithms are currently used. Though neural network has proven to outperform nearly every other traditional machine learning algorithm in various scenarios, there are some disadvantages in which traditional algorithms may deliver a more satisfying result. In the case of neural networks, the best-known disadvantage is their "black box" nature where the individual implementing finds it hard to understand what caused the model to come up with the prediction.

This report presents the hybrid model of combining both lexicon-based and traditional learning-based approach. The lexicon-based classifier will be used to annotate the training data for the learning-based classifier to perform on. This leverages the performance of setting up a learning-based approach as well as taking advantage of the accuracy given by the lexicon-based approach.

## 1.2 Research

The report serves to investigate the possibility of creating a hybrid sentiment analysis tool by combining the lexicon-based and learning-based approaches. The purpose is to determine the performance and convenience of the hybrid model as compared to lexicon-based or learning-based approach when analysing the sentiments of texts.

The secondary aim of this report is to engage in a Semantic Evaluation competition task hosted in the Internal Workshop on Semantic Evaluation 2018 without implementation of any neural network techniques. Given the rise of popularity of deep learning usage in the recent years, the author wants to evaluate if traditional machine learning methodologies can be on par or outperform neural network techniques.

As natural language processing plays an important role in the evaluation of sentiments, the author will also evaluate different pre-processing procedures that may aid in improving the evaluation score of the sentiment tool.

The following questions regarding the implementation of a hybrid model are answered as part of the investigation criteria:

- What pre-processing procedure aided in the improvement of sentiment regression result?
- Which lexicon-based resource yielded the best result for sentiment regression?
- Which traditional learning-based algorithm yielded the best result for sentiment regression?
- What combination of pre-processing techniques, together with the hybrid model yielded the best result within the author's defined parameter constraints?

## 1.3 Hypothesis

The proposed hybrid model is expected to outperform the lexicon-based model while being on par or with the possibility of performing better than traditional learning-based model.

## 1.4 Motivation

Social network services such as Twitter, Instagram and Facebook have become an integral part of our daily lives. Unlike traditional media outlets, these online services do not have any restrictions to what is being said and have become a popular avenue for expressing opinions on the Internet[4]. Companies and government agencies have also resorted utilising these social network services to understand how people react to certain products or political news. During the Unites States Presidential Election in 2016, such tools were used to predict the next United States President. These tools analyse the emotive language used by

users in social network services. EMOTIVE, a sentiment analysis tool for example, was able to successfully predict Donald Trump's win over the election[5] based on the tweets it had processed prior to the results. Subjective analysis of such user-generated content has gained traction over the years for political, marketing and advertising purposes.

The increasing number of empirical analysis of sentiment and mood based on textual collections of data generated social network services has been evident over the recent years and has been used to categorise such textual information[6]. However, we often use language to communicate not only the emotion or sentiment we are feeling but also the intensity of it. Existing sentiment analysis tools have features that only annotate the emotive states of the datasets categorically but without an indication of intensity. These tools have almost always been framed as classification tasks – to identify one emotion among n-affect categories for a phrase or sentence. On the other hand, coming out with regression-based categorisation for such classification tasks is often useful for natural language applications to understand the intensity to which affect is expressed in text[7].

In view of the popularity of sentiment analysis and the growing demand of building artificial models to aid with such tasks, the author concludes that the findings presented in this report may be of use in the development of a more optimal sentiment analysis tool in the future.

## 1.5 Limitations

### 1.5.1 Scope

The datasets used targets a corpus of English tweets collected by polling the Twitter API for tweets that included emotion-related words. The datasets are provided by the organisers of the competition, International Workshop on Semantic Evaluation. As an additional initiative in conjunction with to the author's main research objectives, the author has participated in a competition hosted by

the organisers - SemEval-2018 Task 1: Affect in Tweets[8] as part of a research effort. The emotion intensity regression task is attempted:

- EI-reg (an emotion intensity regression task): Given a tweet and an emotion E, determine the intensity of E that best represents the mental state of the tweeter—a real-valued score between 0 (least E) and 1 (most E).

### 1.5.2 Neural Networks vs Traditional Algorithms

Neural Networks with its implementation of multi-layer processing have proven to be a topic of interest and studies in the recent years. The advances and breakthroughs in the development of algorithms have been a contributing factor to the increased usage to perform various predictions. This is made possible with the availability of the computational resources that are readily available. However though with advantages over traditional algorithms, it is still coupled with several disadvantages, and will likely not see it as a replacement of traditional algorithms but as a supplement for the advancement of artificial intelligence.

First, the computation of neural networks comes in a "black box" settings where we do not know how the model comes out with a certain output. In comparison, algorithms such as Decision trees and Support Vector Machines are interpretable which makes use of rules and distance vectors to compute its outputs. This is important in several domains in the industry, such as banks where they do not use deep learning techniques to predict whether a person is suitable to apply a loan as they need to explain to them a plausible reason for accepting or rejecting them. Human beings in nature are not comfortable with accepting any suggestion without a good reason or a certain valid procedure. This is even more true if sensitive information is being processed or when handling with huge amount of money and resources.

Secondly, neural networks usually require much more data to perform its computations with high accuracy. On the other hand, traditional learning algorithms can be solved with less data and can still achieve a decent accuracy.

Lastly, the cost of neural network's computation power is much more expensive than traditional algorithms. Several deep learning algorithms can take several weeks to train completely from scratch and the amount of resources as well as the number of layers of the network it is being trained on adds on to the costs. Most traditional algorithms do not have such high costs and would normally take a few minutes to a few hours or days to compute the results. Depending on the nature of usage, traditional algorithms may still be preferred over lesser time cost even though if it means to sacrifice some accuracy.

## 1.6 Structure of Report

The report is structured into four main sections and one appendix section. The main sections consisting of Related Works, Implementation, Results and Discussion. Related works covers published papers which attempted research areas similar to the experiments conducted in the report. Implementation details the process creation of the hybrid model, as well as the necessary pre-processing of datasets that are done beforehand. The acquired results are presented in the Results section and a discussion among the attempted approaches is reflected in the Discussion section.

The appendix section consists of a compiled version of results attempted, steps to compile the program and execute the classifier, and lastly the bibliography.

# Chapter 2

# Related Works

Sentiment analysis is currently a well-discussed topic of interest in the field of machine learning, with a considerable number of published papers detailing the different approaches to analyse the data as well as their comparisons. The resources below are found to share similar ideas on how a hybrid model can be achieved by tapping into well-known approaches and leveraging on their experimental results.

## 2.1 Lexicon-based Approaches

The lexicon-based approach[9] determines the sentiment or polarity of opinion through the comparison of lexicons with the targeted words in the data set. This method as mentioned in the previous section can result in low recall for entity-level sentiment analysis.

The paper by V. Hatzivassiloglou and K. R. McKeown[10] describes a basic sentiment classification framework which boasts a classification prediction rate of up to 90% on the positive or negative semantic orientation of conjoined adjectives. However, the results are constrained with the knowledge that adjectives appear in a modest number of conjunctions in the corpus used.

M. Hu and B. Liu[9] describe in detail their methodology of manually creating a lexicon using useful words with regards to the topic that they are tackling. This will help enable the algorithms to discriminate positive from negative sentiment. The lexicons, together with other data such as predefined grammar rules, an English dictionary that provides the base language data are being computed against several scoring classifiers to determine the sentiments of stock postings on an investor bulletin.

The paper by T. Nasukawa and J. Yi[11] proposes a lexicon-based framework to extract sentiments associated with polarities for specific subjects from a document and has boasted a accuracy of 75-95%. Again, the high accuracy of results is diminished by removal of ambiguous cases and the limited domain scope of the classification, damaging the recall rate in return.

The paper by A. Devitt and K. Ahmad[12] notes the impact of being over reliant on lexical resources, where any hand-coded or corpus-derived lexicon will have a certain degree of error or consistency. The given solution is to spread the risk associated from only using a single lexical resource to drawing from multiple sources. This is further demonstrated in detail from the paper S.-M. Kim and E. Hovy[13]. These studies serve as the basis for the decision to combine several lexicon resources into one feature vector.

## 2.2 Learning-based Approaches

B. Pang, L. Lee, and S. Vaithyanathan [1] provide an in-depth comparison on three learning-based classifiers: maximum entropy classification, naive Bayes (NB) and support vector machines (SVM). They have concluded that these 3 machine learning techniques outperforms human-produced baseline with SVM coming out on top but underperform when it is executed on sentiment-based classification than on traditional topic-based classification. Using learning-based classifiers has produced a common phenomenon in documents where the text serves as a deliberate contrast to the actual context. Furthermore, the paper suggests that models using features introduced with unigram model combination have shown to be effective in the past.

A. Go, R. Bhayani, and L. Huang[14] have proven that using unigrams with combination of traditional learning algorithms such as Naïve Bayes, MaxEnt and SVM perform much better than their baseline of including Twittratr's list of keywords consisting of 174 positive words and 185 negative words. Experimental results in the paper also shows that using bigrams alone results in the drop of accuracy in the case of using both MaxEnt or SVM and that using bigrams as feature is not useful because the feature space is very sparse. In addition, the

paper has evaluated that using POS tags as features are not useful and saw a decrease in the accuracy for SVM and Naïve Bayes algorithms and is consistent with the results shown by Pang and Lee[1].

These studies serve as the basis for the decision regarding choice of a learning model for use in the implementation. It also complements the decision to include n word gram model as an additional filter in processing the data and removing POS tags to be used as a feature in the hybrid model constructed.

## 2.3 Hybrid Approach

The paper by L. Zhang, R. Ghosh, M. Dekhil, M. Hsu, and B. Liu [2] proposes a method of adopting a lexicon-based approach to perform entity-level sentiment analysis with a traditional learning-based classifier trained to assign polarities to the entities in the newly identified data source. The experimental results show that their proposed method outperforms any other learning-based or lexicon-based method in terms of accuracy. The paper has also noted that the supervised method they have implemented using Maximum Entropy performs poorly. The paper provides valuable insight into the underlying principles of combining the techniques, but the proposed approach only makes use of the traditional machine learning algorithm to label the data sets and not use as a model for evaluating the actual sentiment of the data.

The paper by S. Tan, Y. Wang, and X. Cheng[15] presents useful information and ideas regarding the combination of learning-based and lexicon-based techniques using a centroid classifier. However, it does not go into detail about the specifics of used implementations.

These serve as the basis for the decision to continue the implementation of the hybrid approach where the objective of the report is to increase the accuracy of hybrid approaches by experimenting different methods of combining lexicon-labelled data with traditional learning-based algorithms identified in this report.

# Chapter 3

# Implementation

The implementation of the proposed approach consists of mainly four parts. The first part treats the data source to ensure that it conforms to the structure of the input lexicon-based or learning-based classifier and removes unwanted entities. The second part includes the addition of new features into the data that will help in improving the accuracy of the prediction as discussed in the Related Works Section. The third part introduces the usage of different lexicons to be processed as individual features and combined to a single feature vector. Lastly, the fourth part encompasses the usage of traditional learning algorithms together with the derived features.

## 3.1 Programming Language and Frameworks

The language of choice for processing of data is Python[1] mainly because of its flexibility to handle and process data easily with a wide range of packages that works hand in hand with data analysis and machine learning. Furthermore, many well-known machine learning libraries are based-off the language itself, showing its popularity among the field of data analytics and artificial intelligence. The main machine learning framework used is Weka [2][3] with java command line implementation, which has a comprehensive collection of data pre-processing and modelling techniques. Furthermore, it has a package manager to allow easier

[1] Python https://www.python.org/downloads/release/python-370/
[2] Weka https://waikato.github.io/weka-wiki/
[3] Java https://www.java.com/en/download/

installation of extension packages, such as the AffectiveTweets package[4] and LibLINEAR package[5] which will be using in the experiments.

## 3.2 Data Pre-processing

### 3.2.1 Cleaning and arranging the data

The datasets[6] include a corpus of tweets and has altogether 8 files, separated into 4 training and 4 test data sets as shown in Figure 1. Each of the data files for training and test data is tagged to 1 of the 4 emotions, namely:

- anger
- fear
- joy
- sadness

The corpus of tweets is collected by polling the Twitter API for tweets that include emotion-related words such as '#angry', 'annoyed', 'panic', 'happy', 'elated', 'surprised', etc. Each of the training data set contains thousands of unique tweets that are share the same emotion as 1 of the above mentioned. The test data set contains tens of thousands of tweets which will be used to evaluate the accuracy of the training model developed.

For every data set, there are 4 column headers:

- ID – The tweet ID that is fetched from the Twitter API
- Tweet – The actual content of the tweet
- Affect Dimension – The respective emotion related to the tweet
- Intensity – A continuous value of the intensity of the Affect Dimension from a scale of 0 to 1.

[4] Affective Tweets https://affectivetweets.cms.waikato.ac.nz/
[5] LibLINEAR https://www.csie.ntu.edu.tw/~cjlin/liblinear/
[6] Datasets https://competitions.codalab.org/competitions/17751#learn_the_details-datasets

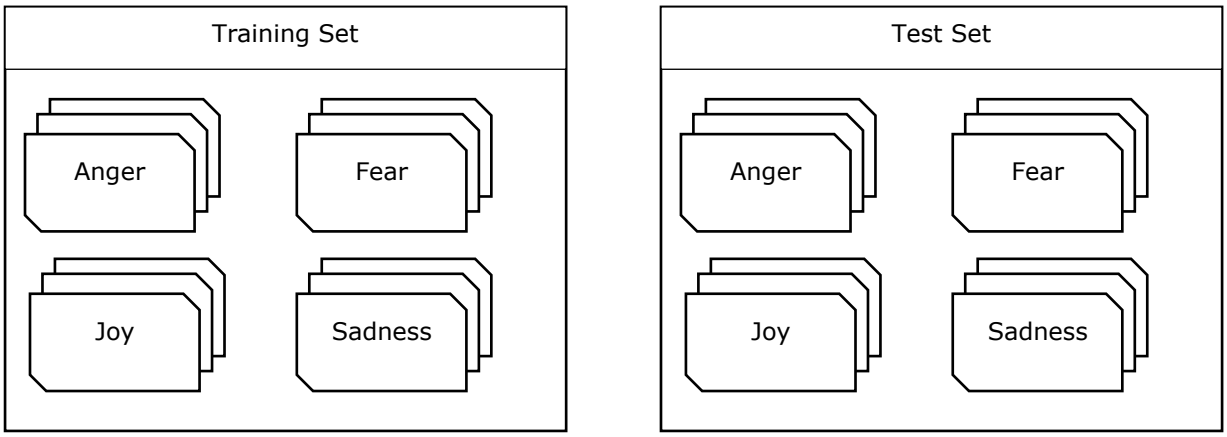


*Figure 1. An illustration of how the data set is being split and categorized*

Before the data sets are being evaluated, it must be converted into an ARFF (Attribute-Relation File Format) file for it be compatible for usage with the Weka software. An ARFF file is an ASCII text file that describes a list of instances sharing a set of attributes and were developed by the Machine Learning Project at the Department of Computer Science of The University of Waikato for use with the Weka machine learning software. Figure 2 shows the raw dataset while Figure 3 shows the dataset in ARFF format converted using codes.

```
ID  Tweet   Affect Dimension    Intensity Score
2017-En-10264   @xandraaa5 @amayaallyn6 shut up hashtags are cool #offended anger   0.562
2017-En-10072   it makes me so fucking irate jesus. nobody is calling ppl who like hajime abusive stop with the strawmen lmao   anger   0.750
2017-En-11383   Lol Adam the Bull with his fake outrage...  anger   0.417
2017-En-11102   @THATSSHAWTYLO passed away early this morning in a fast and furious styled car crash as he was leaving an ATL strip club. That's rough stuff
anger   0.354
```

*Figure 2. An illustration of a raw data file*

```
@relation EI-reg-En-anger-train.txt

@attribute ID string
@attribute Tweet string
@attribute Affect_Dimension string
@attribute Intensity_Score numeric

@data
'2017-En-10264','@xandraaa5 @amayaallyn6 shut up hashtags are cool #offended','anger',0.562
'2017-En-10072','it makes me so fucking irate jesus. nobody is calling ppl who like hajime abusive stop with the strawmen lmao','anger',0.750
'2017-En-11383','Lol Adam the Bull with his fake outrage...','anger',0.417
'2017-En-11102','@THATSSHAWTYLO passed away early this morning in a fast and furious styled car crash as he was leaving an ATL strip club. That"s rough
stuff','anger',0.354
```

*Figure 3. An illustration of an ARFF data file*

### 3.2.2 Pre-processing of data

*Part-of-Speech Tagging (POS)*

The process allows the program to automatically tag each word in accordance with its syntactic function such as: noun, pronoun, adverb, adjective, verb, interjection, intensifier, etc. The goal of a Part-of-Speech Tagging is to extract patterns using regular expressions in the target text based on analysis of frequency distributions of these part-of-speech. Both papers by E. Kouloumpis, T. Wilson, and J. D. Moore [16] and A. Go, R. Bhayani, and L. Huang[14] reported a decrease in performance using POS tagging. Furthermore, lexicons resources will be used and some of these resources have already taken care of certain word pre-processing such as part of speech and stemming. As such, the author decides not to implement POS tagging in the subsequent experiments.

*Stemming*

Stemming is a procedure of removing the affixes from a word. There are cases of over stemming where accuracy is reduced, and under-stemming recall is low. The overall impact of stemming depends on the dataset and stemming algorithm. It may be exhaustive to find an efficient solution towards optimal accuracy and recall. Furthermore, lexicons resources will be used and some of these resources have already taken care of certain word pre-processing such as part of speech and stemming. As such, the author decides not to implement stemming in the subsequent experiments.

*Stop-words removal*

Stop words are commonly used words which carry a connecting function in the sentence, and usually hold very little meaning by itself. There is no definite list of stop words and are removed from the text before classification since they have a high frequency of occurrence in the text, but do not affect the final sentiment of the sentence.

Stop words are considered to be implemented in the hybrid model as the author believes that it will help in improving the quality of the raw data. Furthermore, in order to represent as many words per sentence as possible by the corresponding word embeddings, the number of unique features has to be reduced. Using stop words helps will help to solve this issue as demonstrated in the example below.

| No implementation of stopwords | Implementation of stopwords |
|---|---|
| Phrase 1 - "I am angry"<br>Phrase 2 - "She is angry" | |
| Number of Features: 2<br>Feature 1 - "I am angry"<br>Feature 2 - "She is angry" | Number of Features: 1<br>Feature 1 – "angry" with the stopwords "I, am, she, is" removed |

Furthermore, if a classifier is to decide among 2 phrases "I am angry" and "angry", it can potentially select the phrase with a higher occurrence of the common English words. This can be distracting and can prevent a model from deciding on the correct class membership. Using stopwords helps to improve learning rate and reducing number of features in use to allow prediction to be more accurate since "noise" or distracting features are removed.

For experiment purposes, stop-words from the Bow library[7] are used, which is available in the Weka package. The stop list is the SMART system's list which consists of 524 common words such as "the" and "of". From Figure 4, using stop-words did improve the overall accuracy of the prediction as compared to the implementation of certain individual features.

[7] http://www.cs.cmu.edu/~mccallum/bow/

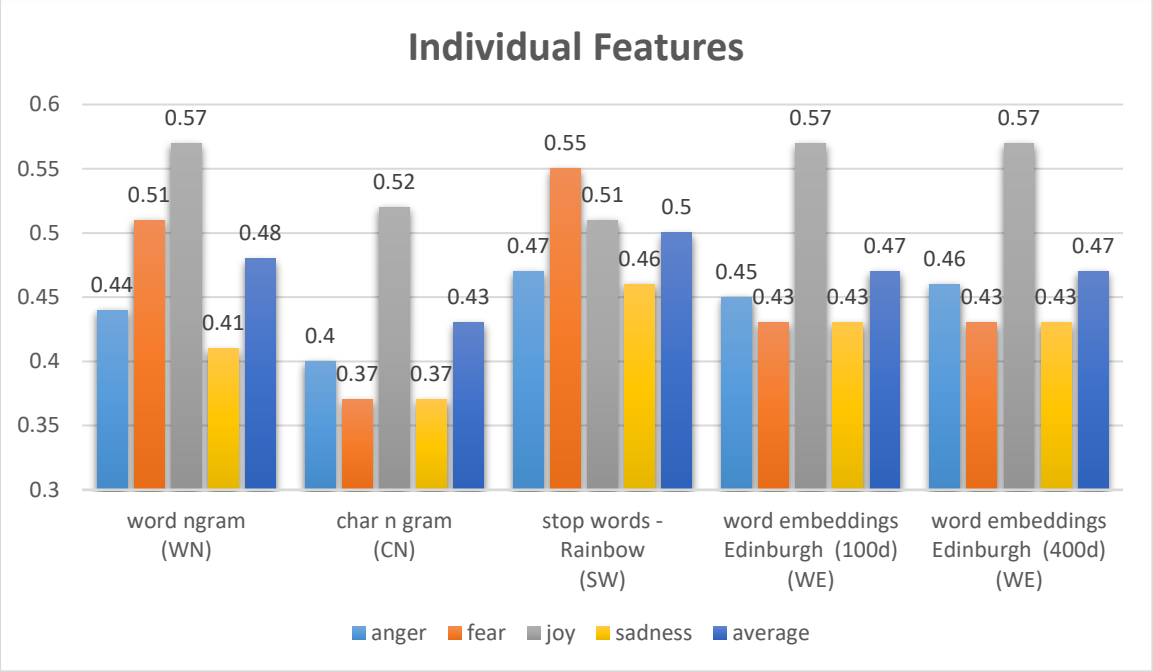


*Figure 4. A bar graph illustrating the improvement of accuracy in using stop-words*

### 3.2.3 Feature Generation and Selection

*Tokenisation into N-grams*

Tokenisation is a procedure of creating a bag-of-words from a text. The incoming string gets broken into many pieces, such as sentences and words and does it by separating words using spaces and punctuation.

Tokenisation may prove to play an essential role in sentiment analysis than in other areas of natural language processing. This is because sentiment information is often sparsely and unusually represented. The following are some examples whereby tokenization generally helps in gathering sentiments. The scenarios below demonstrate reasons as to why tokenisation has been considered to be implemented in the hybrid model.

Emoticons – Emoticons are extremely commonly used on many social media platforms. They represent emotions expressed by the users through a series of characters that mimics the human facial expression.

Informative HTML Tags – Some HTML mark-up tags such as *strong, b, italics* are often indicators of sentiment. Their opening or closing elements can be treated as individual tokens.

Capitalization - Using Capitalization on words are often seen as an indication to emphasise certain words or to mimic a "loud" tone to the directed audience. This can tend to be seen as a correlation with sentiment information.

In a basic polarity predictive model to predict the sentiment labels of documents, single words used as features is generally the approach being used. Single words that produces an indication of affect, e.g. bad, good, awesome separates the overall sentiment of the text. However, there are certain drawbacks to using single words as features such as Negations. Negation words are often seen in text and results in a complete change of sentiments. Negation word "not" for example, if appended together with a single feature word "good", produces a negative sentiment from 2 words "not good". If basic tokenisation is used, both words "not" and "good" will be treated as two separate entities and produce and inaccurate results and lead to misclassification. Using frequent n-grams as features can overcome this problem, where depending on the value of n used, will be able to capture Negations perfectly.

A point to take note when using n-grams as features is that it will blow up the feature space tremendously, especially on large data sets. This can usually be solved by having other good feature selection methods to reduce the feature space before building the model.

As the purpose of n-grams is to capture the language structure from the statistical point of view, the choice of the value of n must be carefully considered. If the value of n is too small, the model may fail to capture important difference such as the negation example as explained above. On the other hand, an n value that is too big will result in only very small occurrence of features - capturing only very specific contexts and fail to capture certain small length words that may appear as a subtext of the n gram.

From Figure 5, unigrams and bigrams for word grams produces a better result as compared to words grams with n>=3.

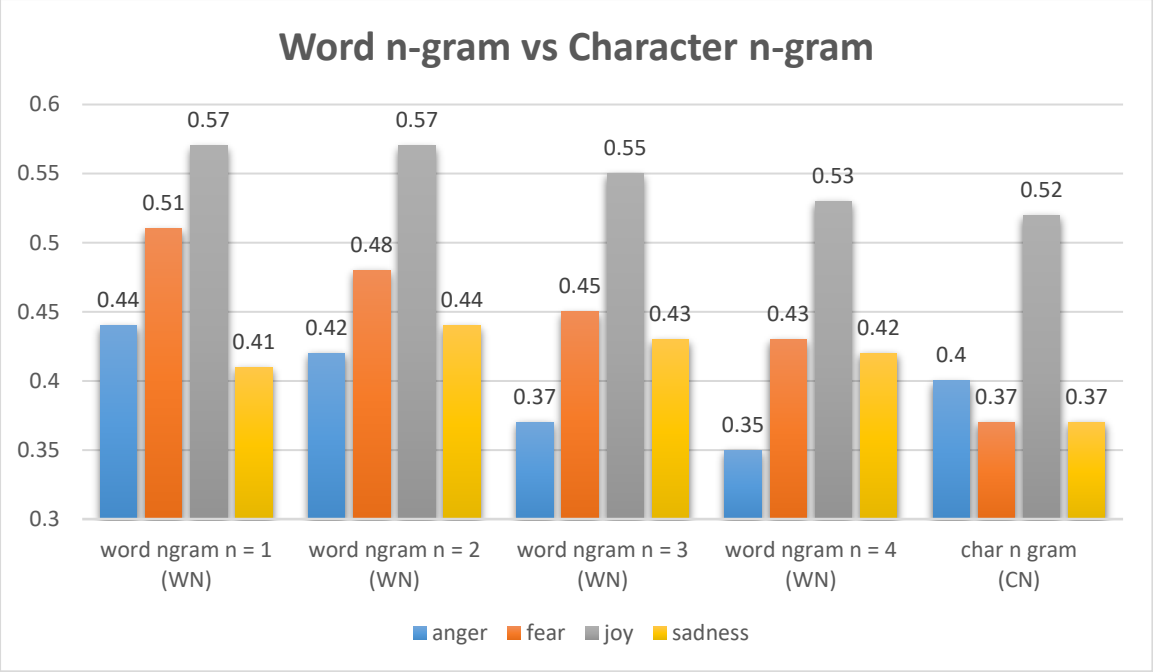


*Figure 5. A bar graph illustrating the accuracy of n gram feature*

*Word Embeddings*

Word embeddings transform human language meaningfully into a numerical form – every word is converted to a set of numbers, or specifically N dimensional vector. Although every different word gets a unique vector, similar words may contribute to having values of vector that are closer to each other. The goal of word embeddings is to capture semantic relationships between words.

Word embeddings is considered as one of the implementation feature in the hybrid model as it solves two problems that are notoriously hard when analysing sentiments on tweets – mainly short text and misspelling of words. Word embeddings groups similar words together – "hello" and "hellooo" may be two different words during tokenisation, but if implemented with word embeddings, the model classifies these two words of the same cluster with similar distance from each other.

There are ready-to-use corpora with pre-trained word embeddings. For example, Weka includes a package with pre-trained word vectors taken from the Edinburgh

corpus[8], with embeddings trained up to 10 million tweets. The parameters were calibrated for classifying words into emotions. The Edinburgh embeddings are obtained by training based on a prediction based vector - skip-gram on the Edinburgh corpus. The advantages of using skip-gram include being able to capture two semantics for a single word and outperforming every other method generally. For example, a single word – apple, will have two vector representations, company and fruit.

As word embeddings represent word meaning based on their occurrence in the text, having a text corpus which is very large and diverse for creating word embeddings will be useful as the word embedding vectors will represent the general or dominant meaning of a word. Such pre-trained embedding corpora can be used as similarly as dictionaries which return vector embeddings for the requested words and save a considerable amount of computing time.

Manual annotation of human-labelled training data is commonly present in conventional machine learning models. Such action often results in limited amount of data being annotated which reduces the accuracy of classification models. Word embeddings helps in alleviating this problem as they provide vector representation on words that are not represented or scarcely represented in the training data based on their similarity with other words. This is useful especially in short texts such as the twitter dataset which the author is using in the experiments.

From Figure 6, the accuracy of predictions has slightly increased for anger, joy and sadness, with fear not being able to accurately depict the similarity of fear vectors.

[8] Edinburgh corpus http://www.aclweb.org/anthology/W/W10/W10-0513.pdf

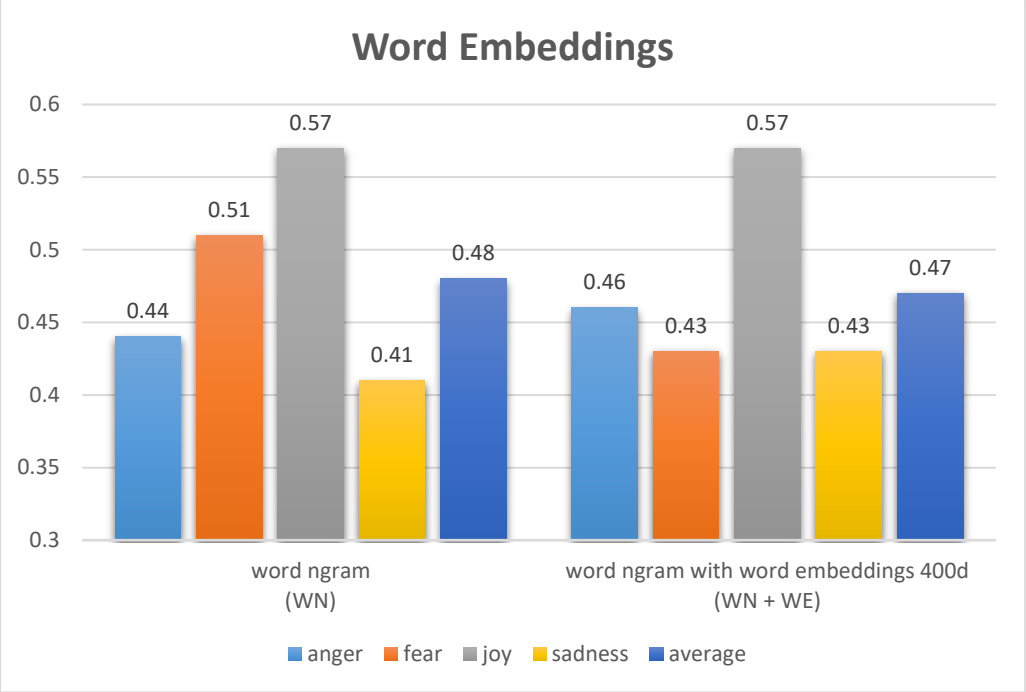


*Figure 6. A bar graph illustrating the accuracy of implementing word embeddings*

*Affect lexicons*

A lexicon is the vocabulary of a language or subject. Different lexicon resources provide different form of annotating the properties of the lexical entries. Using lexicons as the approach to analyse sentiments has been prominent for several years, as classifier-based methods are not optimally suitable due to domain specificity and lack of context.

In domain-specificity, classifying normal text documents is not an issue as the classifier learns several features that are domain specific and may not hold in other domains or cause a drift of sentiments. The problem can be seen in classifying sentiments such as this example below:

| Domain: | Book Review | Movie Review |
|---|---|---|
| Sentence: | Go Read the book | |
| Polarity: | Positive | Negative |

Reading a book may be a positive statement when in context of a book review, indicating that the book is indeed worthwhile to read. However in the context of a

movie review, it as a negative sentiment, indicating that the movie has a negative rating and people are better off reading the book instead. Furthermore in the issue of context in classification, a normal classifier would group phrases such as "good" and "not good" together. This would produce a good accuracy disregarding the polarities of both words. However in sentiment analysis, this results in a very poor prediction model.

The use of sentiment lexicons solves these problem by creating a self-defined resource with several possible forms to detect the polarity of words, including:

- Fixed categorisation into positive or negative
- Finite number of graded categorisation, e.g. strongly positive, positive, neutral, negative, strongly negative
- A real value denoting sentiment strength e.g. interval of [-1,1]

Many researchers have created their own sentiment lexicons according to the forms mentioned above, which the author will be implementing in the hybrid model. These are the description of lexicons that are used in the experiments, with additional package such as the SentiStrength[9] package which calculates the positive and negative sentiment strengths for a tweet.

MPQA[10]: counts the number of positive and negative words from the MPQA subjectivity lexicon.

Bing Liu[11]: counts the number of positive and negative words from the Bing Liu lexicon.

AFINN[12]: calculates positive and negative variables by aggregating the positive and negative word scores provided by this lexicon.

Sentiment140[13]: calculates positive and negative variables by aggregating the positive and negative word scores provided by this lexicon created with tweets annotated by emoticons.

[9] SentiStrength http://sentistrength.wlv.ac.uk/
[10] MPQA http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/
[11] Bing Liu https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon
[12] AFINN https://github.com/fnielsen/afinn
[13] Sentiment140 http://saifmohammad.com/WebPages/lexicons.html#NRCTwitter

NRC Hashtag Sentiment lexicon[14]: calculates positive and negative variables by aggregating the positive and negative word scores provided by this lexicon created with tweets annotated with emotional hashtags.

NRC Word-Emotion Association Lexicon[15]: counts the number of words matching each emotion from this lexicon.

NRC-10 Expanded[16]: adds the emotion associations of the words matching the Twitter Specific expansion of the NRC Word-Emotion Association Lexicon.

NRC Hashtag Emotion Association Lexicon[17]: adds the emotion associations of the words matching this lexicon.

SentiWordNet[18]: calculates positive and negative scores using SentiWordnet. A weighted average of the sentiment distributions of the synsets for word occurring in multiple synsets is calculated. The weights correspond to the reciprocal ranks of the senses to give higher weights to most popular senses.

[14] NRC Hashtag Sentiment lexicon http://saifmohammad.com/WebPages/lexicons.html#NRCTwitter
[15] NRC Word-Emotion Association Lexicon http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm
[16] NRC-10 Expanded http://www.cs.waikato.ac.nz/ml/sa/lex.html#emolextwitter
[17] NRC Hashtag Emotion Association Lexicon http://saifmohammad.com/WebPages/lexicons.html#HashEmo
[18] SentiWordNet http://sentiwordnet.isti.cnr.it/

## 3.3 Lexicon-based method

The lexicon-based method utilises an external package of Weka, namely Affective Tweets TweetToLexiconFeatureVector function. From Figure 7, the training dataset is passed into the lexicon-based classifier which contains the TweetToLexiconFeatureVector function. The classifier pre-processes the test data into individual entries of text and extends it to the Lexicon Evaluators. The Lexicon Evaluators then computes the sentiment score using various Lexicon resources and the respective evaluator functions. The final values are sent back to the classifier as feature vector values and the classifier proceeds by labelling the test data with the combined feature vector value. The labelled test data is evaluated against the labelled gold test data [19] to compare the accuracy.

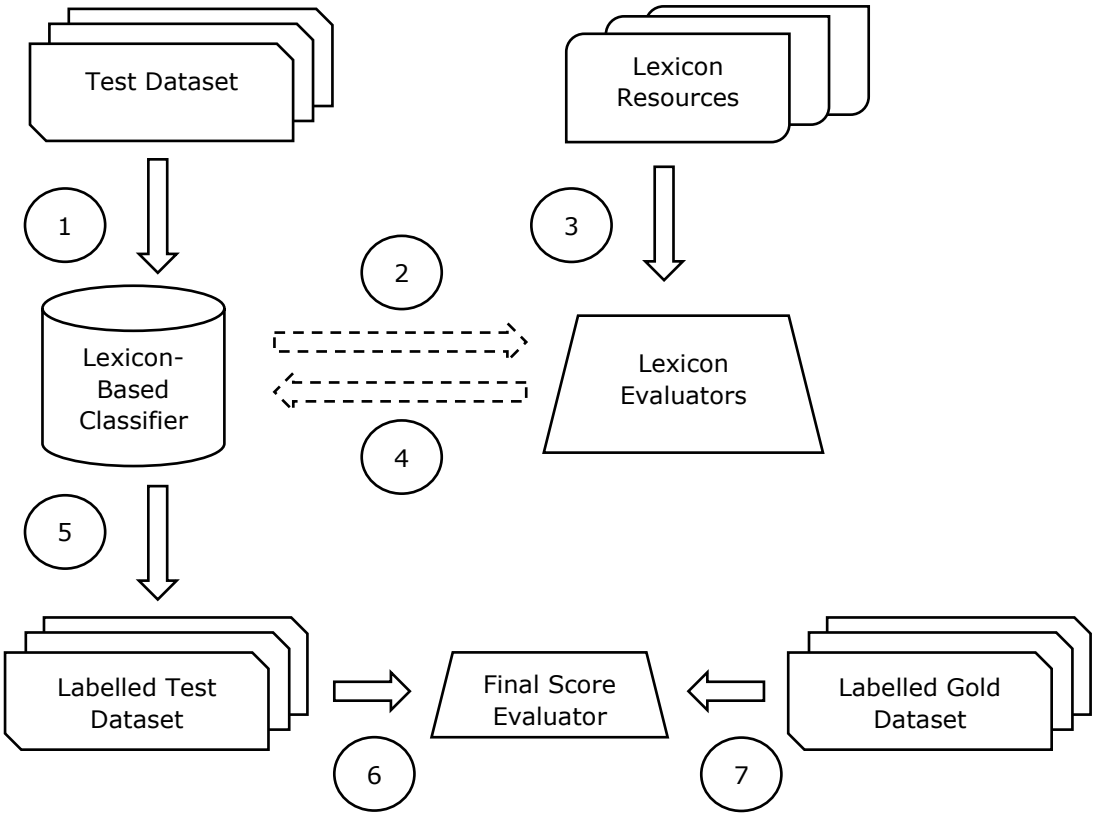


*Figure 7. A flow diagram of the lexicon-based method*

[19] Labelled gold test data given by competition organizer

## 3.4 Learning-based method

The learning-based method utilises available traditional regression algorithms in Weka as well as the L2-regularized L2-loss SVM regression model in the LIBLINEAR external package of Weka. Different classifier algorithms are also tested to evaluate the model which yields the best result. The following classifiers are tested:

- L2-regularized L2-loss SVM regression model with the regularization parameter C set to 1, implemented in LIBLINEAR.
    - L2-regularlization is known as ridge regression
    - L2-loss is calculated by sum of the all the squared differences between the true value and the predicted value
- Random Forest
- Additive Regression / Gradient Boosting
- AdaBoost
- Bagging
- Linear Regression

Linear classification has become one of the most promising learning techniques for large sparse data with huge number of features, which coincides with the type of datasets being experimented on. Linear classification is recommended for text classification as most of text classification problems are linearly separable and contains a lot of features. As such, the main algorithm that is proposed by the author is linear SVM implemented in LIBLINEAR.

The idea behind SVM is to find the hyperplane that maximizes the margin around the hyperplane that separates the different classes. Using a linear classifier often gives bad results as it is not easy to accurately separate using a linear decision boundary. This is solved by mapping the original non-linear separable data into a higher dimension in when it can be separated linearly. This allows features to features mapping at the expense of being computational expensive proportional to the amount of features and data.

However, the author understands that the number of features is large as from the experiments, and sees that mapping data to a higher dimensional space is not needed as the nonlinear mapping does not improve the performance. Using a

linear kernel tends to perform very well in this kind of situation such as document classification. In sentiment analysis, text are being analysed in various levels, mainly document-level, and sentence-level aspect level. In short texts such as the twitter dataset that the author is experimenting with, the analysis falls under document level analysis which complements well with the type of learning model suggested.

To curb the computational cost, LIBLINEAR is introduced. LIBLINEAR provides efficiency for training large-scale problems at O(n) cost as compared to a general SVM solver at approximately $O(n^2)$ to $O(n^3)$. For Logistic regression, LIBLINEAR implements a trust region Newton method. It is accompanied with regularization and loss functions which will help to reduce the amplitude of the model's coefficients, reducing overfitting/multicollinearity and may lead to a more stable and robust model.

Furthermore, accompanied by many published papers, SVM has proven to show a better score as compared to any other algorithms used. In order to compensate the amount of computational time, data pre-processing are implemented to reduce the feature space as much as possible.

All the learning-based classifiers are implemented using the same approach as shown in Figure 8; the training data is pre-processed before it is sent to the learning-based classifier. The test set is passed into the learning-based classifier and is evaluated by the classifier model. The test set is automatically labelled by the classifier model before the score is being evaluated against the gold test data to predict the accuracy.

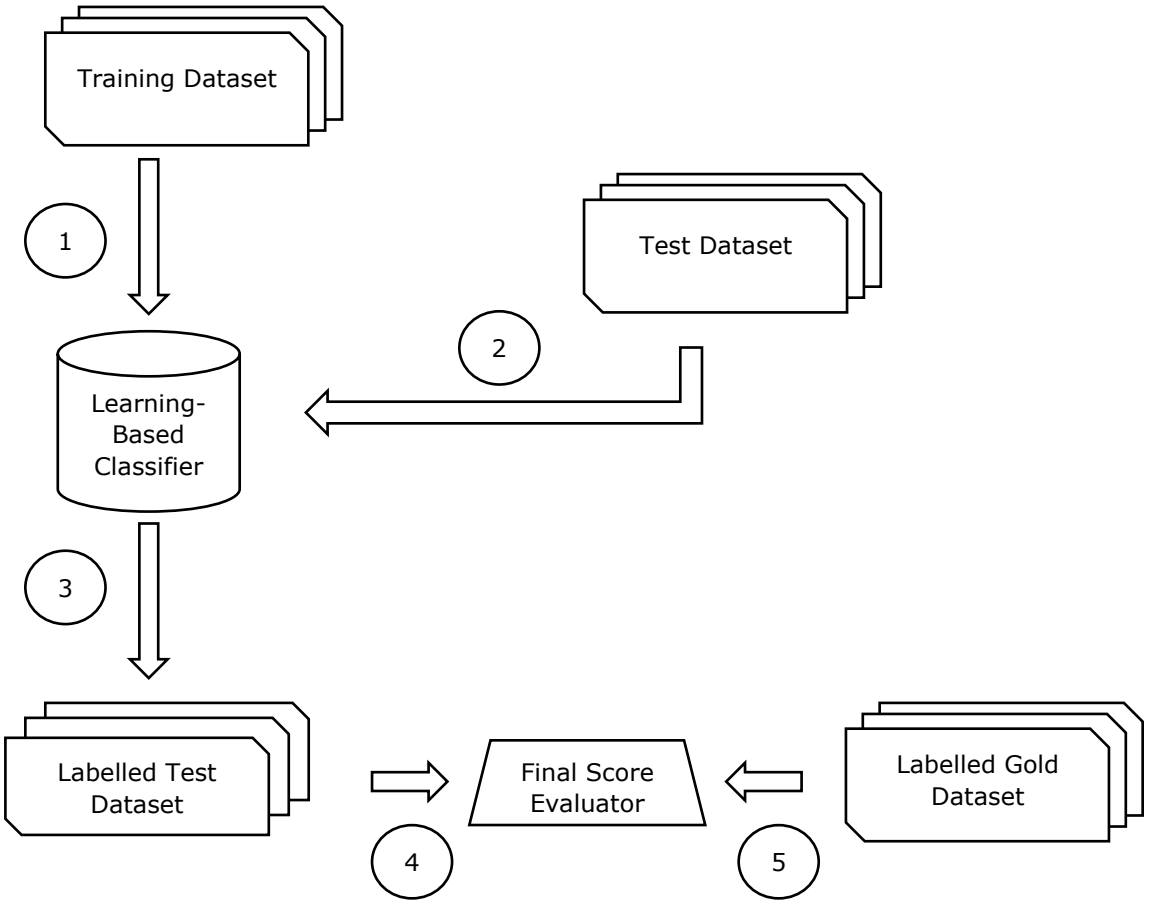


*Figure 8. A flow diagram of the learning-based method*

## 3.5 Hybrid method

The hybrid method is achieved by using the output feature vector from the lexicon-based method as an input feature for the learning-based method as demonstrated in Figure 9.

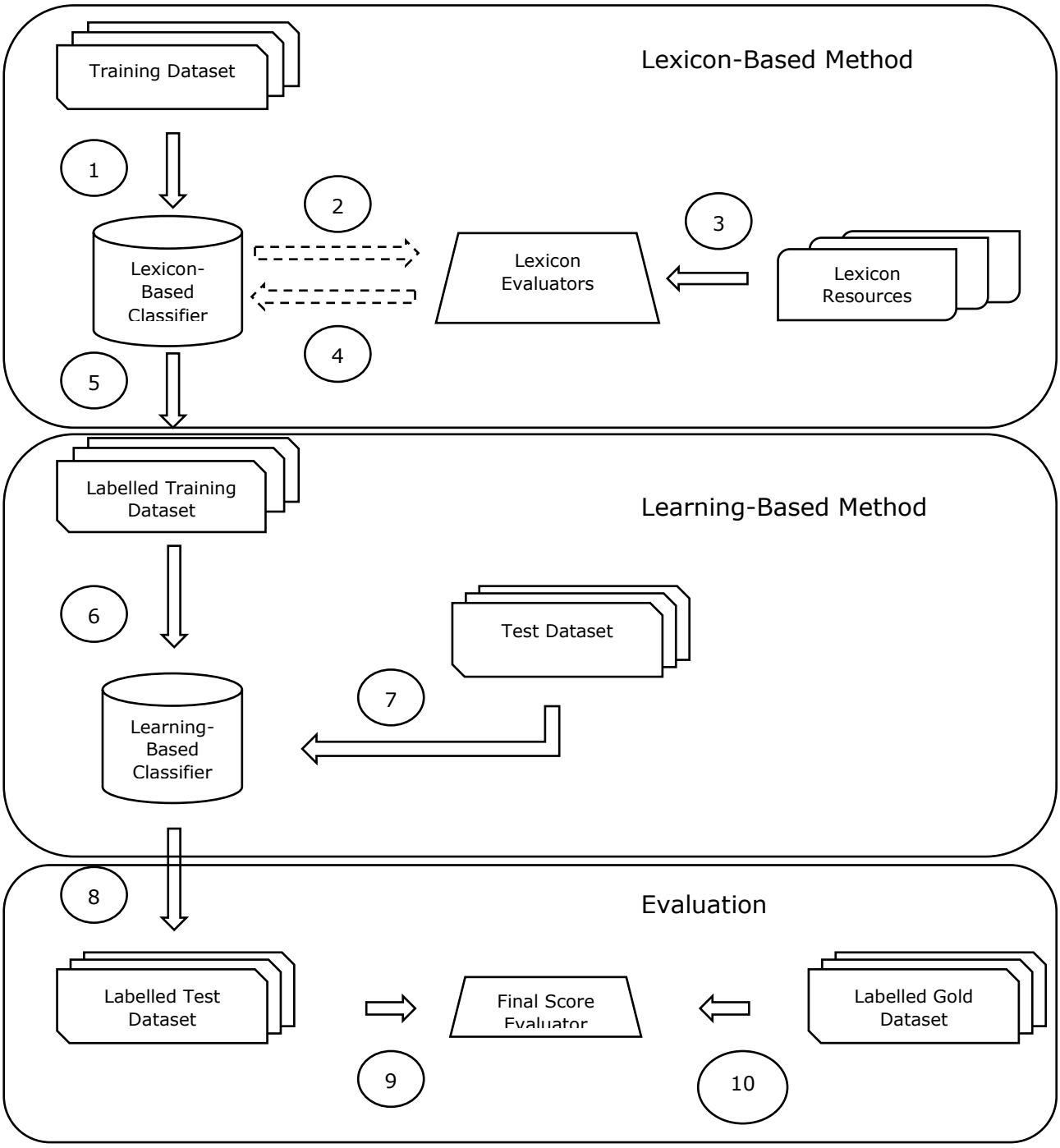


*Figure 9. A flow diagram of the hybrid method*

# Chapter 4

# Results

The evaluation metrics is based on calculating the Pearson Correlation Coefficient using the scikit-learn library [20] with the Gold ratings/labels that can be taken from the competition page. The score ranges on a continuous scale of 0 to 1, with 0 being the least accurate and 1 being the most accurate comparison. Highlighted in bold refers to scores that scored the highest for the respective emotions.

| Pearson Correlation r | | | | | |
|---|---|---|---|---|---|
| | anger | fear | joy | sadness | Average |
| ***Individual feature sets*** | | | | | |
| word ngrams (WN) | (n=1)0.44<br>(n=2)0.42<br>(n=3)0.37<br>(n=4)0.35 | (n=1)0.51<br>(n=2)0.48<br>(n=3)0.45<br>(n=4)0.43 | (n=1)0.57<br>(n=2)0.57<br>(n=3)0.55<br>(n=4)0.53 | (n=1)0.41<br>(n=2)0.44<br>(n=3)0.43<br>(n=4)0.42 | 0.48<br>0.48<br>0.45<br>0.43 |
| char. ngrams (CN) | 0.40 | 0.37 | 0.52 | 0.37 | 0.43 |
| Stop words (SW) - Rainbow | 0.47 | 0.55 | 0.51 | 0.46 | 0.50 |
| word embeddings (WE) - Edinburgh 100d | 0.45 | 0.43 | 0.57 | 0.43 | 0.47 |
| word embeddings (WE) – | 0.46 | 0.43 | 0.57 | 0.43 | 0.47 |

[20] Scikit http://scikit-learn.org/stable/

| | | | | | |
|---|---|---|---|---|---|
| Edinburgh 400d | | | | | |
| All Lexicons (L) | 0.60 | 0.59 | 0.61 | 0.59 | 0.61 |

*Figure 10. A table comparing the accuracy of data pre-processing and features*

## 4.1 Lexicon-Based Approach

### 4.1.1 Table Comparison of lexicon resources

| ***Individual Lexicons*** | | | | | |
|---|---|---|---|---|---|
| | anger | fear | joy | sadness | Average |
| Mpqa | 0.12 | 0.24 | 0.30 | 0.13 | 0.20 |
| BingLiu | 0.22 | 0.30 | 0.33 | 0.20 | 0.26 |
| Afinn | 0.33 | 0.30 | 0.37 | 0.23 | 0.31 |
| Sentiment140 | 0.31 | 0.34 | 0.34 | 0.41 | 0.35 |
| NRC-Hash-Sent | 0.37 | 0.27 | 0.38 | 0.39 | 0.35 |
| NRC10 | 0.20 | 0.29 | 0.33 | 0.17 | 0.25 |
| NRC-Expanded | 0.27 | 0.25 | 0.40 | 0.27 | 0.29 |
| NRC-Hash-Emo | **0.50** | **0.49** | 0.45 | **0.47** | **0.48** |
| SentiWordNet | 0.13 | 0.18 | 0.32 | 0.16 | 0.26 |
| SentiStrength | 0.43 | **0.49** | **0.47** | 0.45 | 0.46 |

*Figure 11. A table comparing the accuracy of different individual lexicons*

## 4.2 Learning-based Approach

### 4.2.1 Table Comparison of learning-based classifiers

| ***Individual Classifiers (Using WE(Edinburgh) + SW)*** | | | | | |
|---|---|---|---|---|---|
| | anger | fear | joy | sadness | Average |
| SVM Regression | 0.51 | **0.46** | 0.52 | **0.45** | 0.48 |
| Random Forest | 0.46 | 0.41 | 0.50 | 0.36 | 0.43 |
| Additive Regression/ Gradient Boosting | **0.52** | **0.46** | **0.55** | 0.43 | **0.49** |
| AdaBoost | 0.18 | 0.23 | 0.36 | 0.20 | 0.24 |
| Bagging (REPTree) | 0.32 | 0.33 | 0.45 | 0.27 | 0.34 |
| Linear Regression | 0.47 | 0.44 | 0.49 | 0.37 | 0.45 |

*Figure 12. A table comparing the accuracy of different learning-based algorithms*

## 4.3 Hybrid Approach

### 4.3.1 Table Comparison on different combinations of pre-processors

| ***Combinations of features and pre-processors*** | | | | | |
|---|---|---|---|---|---|
| | anger | fear | joy | sadness | Average |
| WN + CN + WE | 0.40 | 0.46 | 0.52 | 0.38 | 0.44 |
| WN + CN + L | 0.56 | 0.60 | 0.61 | 0.51 | 0.57 |
| WE + L | 0.67 | 0.63 | **0.64** | **0.61** | **0.64** |
| WN + WE + L | 0.61 | **0.65** | **0.64** | 0.55 | 0.61 |
| WN + CN + WE + L | 0.56 | 0.60 | 0.61 | 0.52 | 0.57 |
| WE + L + SW | **0.68** | 0.64 | 0.63 | **0.61** | **0.64** |

*Figure 13. A table comparing the accuracy of different features and pre-processors combinations*

### 4.3.2 Table Comparison on different combinations of various learning classifiers

| ***Individual Classifiers with Lexicons (Using WE(Edinburgh) + L + SW)*** | | | | | |
|---|---|---|---|---|---|
| | anger | fear | joy | sadness | Average |
| SVM Regression | **0.68** | **0.64** | **0.63** | **0.61** | **0.64** |
| Random Forest | 0.60 | 0.54 | 0.60 | 0.55 | 0.60 |
| Additive Regression/ Gradient Boosting | 0.66 | 0.59 | 0.60 | 0.57 | 0.61 |
| AdaBoost | 0.66 | 0.59 | 0.60 | 0.57 | 0.61 |
| Bagging (REPTree) | 0.58 | 0.59 | 0.57 | 0.53 | 0.57 |
| Linear Regression | 0.59 | 0.58 | 0.56 | 0.52 | 0.56 |

*Figure 14. A table comparing the accuracy of different learning-based algorithm with the hybrid approach*

## 4.4 Scoreboard

The column entries are arranged in the following order:

- Name of User who submitted the entries
- Number of submissions
- Date of last entry
- Team Name
- Average Pearson Correlation Coefficient Score for all emotions combined
- Pearson Correlation Coefficient Score for anger
- Pearson Correlation Coefficient Score for fear

- Pearson Correlation Coefficient Score for joy
- Pearson Correlation Coefficient Score for sadness

The Pearson Correlation Coefficient Score is set from a continuous range of 0 to 1, truncated to three decimal places.

### 4.4.1 Baseline Scores by Organisers

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| SVM_Unigrams_Baseline | 1 | 12/28/17 | AIT2018 Organizers | 0.435 (15) | 0.395 (14) | 0.468 (15) | 0.449 (15) | 0.427 (14) |
| Random_Baseline | 3 | 12/11/17 | AIT2018 Organizers | -0.020 (16) | -0.088 (16) | -0.093 (16) | 0.103 (16) | -0.003 (16) |

*Figure 15. Baseline Scores for SemEval 2018 Task 1*

Figure 15 shows the scores of the models submitted by the organisers during the pre-evaluation period. The first column shows the type of baseline implemented.

### 4.4.2 Submission Score

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 15 | derrickpehjh | 1 | 09/10/18 | - | 0.639 (15) | 0.668 (14) | 0.633 (16) | 0.640 (15) | 0.614 (17) |

*Figure 16. Personal Score for SemEval 2018 Task 1*

Figure 16 shows the score that was achieved by the author using the hybrid approach.

### 4.4.3 Top Scorers

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | venkatesh-1729 | 1 | 02/06/18 | | 0.799 (1) | 0.827 (1) | 0.779 (1) | 0.792 (1) | 0.798 (1) |
| 2 | psyml | 1 | 02/01/18 | psyML | 0.765 (2) | 0.788 (2) | 0.748 (2) | 0.761 (2) | 0.761 (2) |
| 3 | dominikstanojevic | 7 | 06/10/18 | Katić's Angels | 0.740 (3) | 0.748 (5) | 0.731 (4) | 0.739 (4) | 0.740 (3) |

*Figure 17. Top 3 Scores for SemEval 2018 Task 1*

Figure 17 shows the top three scorers for SemEval 2018 Task 1.

### 4.4.3 Evaluation of top scorers

Taking only into consideration the participants who scored better than the submitted result by the author, all but one - namely Dmitry and Lidia, attempted the task using different deep learning methods which are CNN, Bidirectional LSTM (BLSTM), LSTMCNN and a CNN-based Attention model (CA). The team headed by Dmitry and Lidia is currently the only team that attempted the task without any implementation of deep learning methods and only used the combination of manually compelled sentiment lexicons with word embeddings. The ranking for Dmitry and Lidia is ranked 1 position higher than the author's submitted result.

The above scenario demonstrates that deep learning techniques are more favoured and scored better in the SemEval2018 Competition Task 1 – Affect in Tweets, as compared to a similar competition task in SemEval2017 where not many participants utilised deep learning methodology in their experiments.

Team Dmitry and Lidia had a similar setup as what the author has proposed, using the combinations of word embeddings and lexicon features, coupled with a learning-based algorithm. The difference between the implementation was the type of lexicons selected as well as the type of word embeddings used. For the learning-based classifier, the team implemented the Gradient Boosting Regressor which came in second out of the learning algorithms that was proposed to be implemented in the experiments.

# Chapter 5

# Discussion

## 5.1 Lexicon Resources

From Figure 11, it can be deduced that using lexicon resources alone provides poor accuracy of results, with none of the lexicon resources providing a minimum average of 0.5 out of 1. As mentioned in the previous sections, the quality of content in the lexicon resource itself plays a big role in the prediction model of a lexicon-based approach. If the word extracted from the target text is found in the lexicon, the sentiment score of that word in added to the total sentiment score of the text.

For example:

@PageShhh1 I know you mean well [-0.22], but I\"m offended [+0.92]. Prick [+0.32].

The sentiment score is the summation of values as seen in the example. For this example, the text expresses a positive anger score which means that the text suggests an angry tone.

However, due to the nature of the data source, the assumption can be made that many of the words in the tweets are either spelled incorrectly or shortened to fit the character limitation of a tweet message. Hence, a decrease of accuracy can may be seen, especially for texts that are crawled from Twitter. Using the same example:

@PageShhh1 I know you mean well [-0.22], but I\"m offended [+0.92]. Prickssssss.

This time, the sentiment score is reduced as the lexicon is not being able to pick up the word “Prickssssss” from its resources. This results in a different result as compared to the previous example.

Furthermore, resources such as Sentiment140, NRC Hashtag Sentiment lexicon and NRC Hashtag Emotion Association Lexicon have performed relatively better than the other lexicon resources. These resources are created based on tweet messages itself and not chunks of formal texts. This enables emotions in tweets to be picked up from lexicon resources more frequently than lexicon resources that are not built from tweets.

The NRC Hashtag Emotion Association Lexicon has the best average score of 0.48 out of the other lexicon resources. A probable inference can be deduced that the lexicon resource itself is built based on not only tweets, but from the hashtags of these tweets. Hashtags are commonly used in tweets to tag messages with similar topics. To use the hashtags properly, people are less inclined to shorten their tags or misspell it to correctly reflect the related topic. This standardises the emotions that are gathered in this specific lexicon resource and hence has a higher probability of matching entries between the lexicon resource and the targeted tweet message.

NRC Hashtag Sentiment lexicon has also proven to give a better score of 0.35 accuracy among the other lexicon resources, following behind NRC Hashtag Emotion Association Lexicon with 0.48 accuracy. It uses a similar concept as NRC Hashtag Emotion Association Lexicon, but instead of associating specific emotions to the targeted words, it provides an association of only positive or negative sentiments. Although it has a larger number of terms as seen in Figure 18[21], it does not perform better as only capturing words of positive or negative sentiment does not reflect the accuracy on capturing specific emotions such as fear, surprise, anger etc. which depending on the intensity of the emotion, may fall under the middle spectrum of the emotion scale.

| | **NRC Hashtag Emotion Lexicon** | **NRC Hashtag Sentiment Lexicon** |
|---|---|---|
| unigrams | 16,862 | 54,129 |
| bigrams | - | 316,531 |
| pairs | - | 308,808 |

*Figure 18. A table comparing the number of terms in specific NRC Resource*

[21] NRC Emotion and Sentiment Lexicons
https://saifmohammad.com/WebPages/AccessResource.htm

## 5.2 Learning-based Algorithms

Based on the average scores in Figure 1, it can be concluded that the SVM Regression and Gradient Boosting Algorithm outperform the rest of the algorithms. The results are based using word embeddings and stop words as pre-processors and only using the emotion intensity and the only feature vector for the respective classifiers.

For SVM Regression, the algorithm performs well on fear and sadness, but loses out on anger and joy. For gradient boosting, the algorithm performs well on the all the emotions and only loses on the sadness emotion. Despite that, it still achieves the highest average score of 0.49 accuracy among all the tested algorithms.

SVM can learn complex decision boundaries in a higher dimensional space. This makes them extremely powerful compared to linear classifiers like Linear Regression. However, SVMs do not scale well with number of observations O(n^2) and become slow very quickly. This is noticed when SVM took a considerable amount of time to compute its prediction compared to the other algorithms while the experiments are conducted.

Out of the four ensemble learning algorithms – Random Forest, Gradient Boosting, AdaBoost and Bagging (REPTree which is the Weka implementation of a standard decision tree), Gradient Boosting outperforms the other three variations.

One probable reason that AdaBoost did not perform well as compared to Gradient Boosting – another boosting algorithm, is due to its weakness of being sensitive to noisy data and outliers. With the text in the data set acting as a non-numerical and categorical feature, the efficiency of the algorithm is reduced by these outliers as the algorithm tries to fit every point perfectly. Due to the amount of noisy data, it would take more counts of estimators until the complete training data fits without any error or until a specified maximum number of estimators. Hence the number of estimators may have been insufficient to provide an optimal score. However, having more estimators comes with the disadvantage of utilising more time to train the model. An example of AdaBoost with decision stump fitting can be seen in Figure 19.

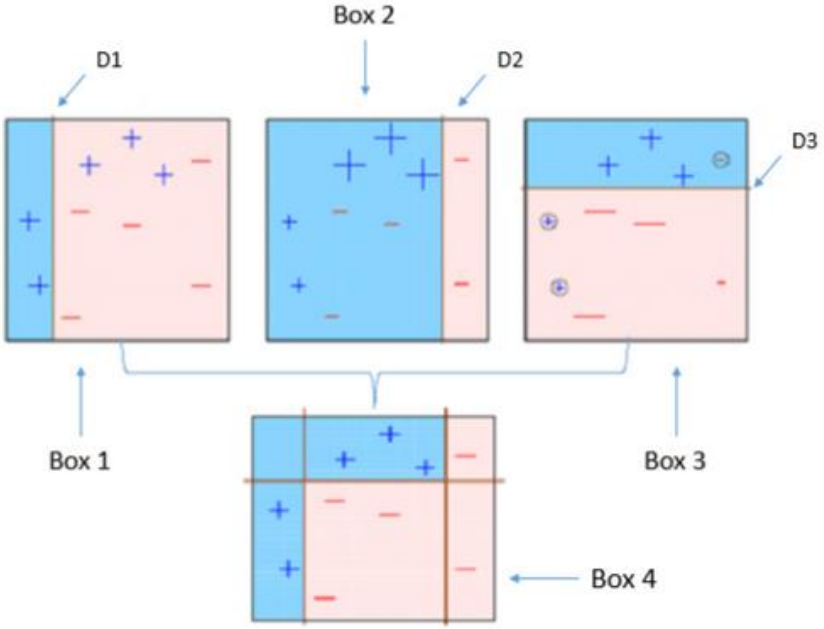


*Figure 19. A diagram of AdaBoost implementation with decision stump taken from https://hackernoon.com/boosting-algorithms-adaboost-gradient-boosting-and-xgboost-f74991cad38c*

Gradient Boosting is another boosting algorithm like AdaBoost where it also tries to create a strong learner from an ensemble of weak learners. However, it is different in certain aspects such that it implements an optimisation problem - taking up a loss function and tries to optimise it. Instead of AdaBoost where the weights are increased to increase the chance of the misclassified samples from being selected correctly – as seen in Figure 19, it minimises the loss function – the difference between the actual value and predicted value by adding more weak learners. The new weak learners are added to concentrate on the areas where the existing learners are performing poorly. The algorithm is less susceptible to noise and outliers as compared to the AdaBoost algorithm which may explain the vast difference in scores for this experiment.

Bootstrap Aggregation (Bagging) decreases the variance of the prediction by combining repetitions to produce multi sets of the same size as the original data. Both bagging and random forest utilizes decision trees, where the latter is the improvement by selecting a random subset of features out of the total. The best split feature from the subset is used to split each node in a tree, unlike in bagging

where all features are considered for splitting a node. The improvement in the algorithm can be noticed from the results shown in Figure 12.

## 5.3 Hybrid Approach

Besides combining the lexicon-based approach and the learning-based approach to from a hybrid model, pre-processing the data is still a necessary implementation to improve the generalizability of the model. Noise, redundant values and outliers are some components that may severely affect the performance of a model. To determine the best combination of pre-processors and features, the author has included the top performing individual features as shown in Figure 4 as well as the best suitable pre-processing tools as discussed in section 3.2.2.

From Figure 13, the combination of Word Embeddings, Lexicons with stop-words provides the best average score. Even though the exclusion of stop words provides the same average score, it has been noted in the experiments that utilising stop-words produced slightly better results.

After determining the best combination of features and pre-processors, another round of experiment is conducted to test the traditional algorithms on the new determined combination of word embeddings, lexicons and stop-words.

The results slightly differ as compared to the results in Figure 12 of section 4.2. In Figure 12, the gradient boosting algorithm performs better than the other algorithms are tested. However, in Figure 14 of section 4.3.2, the same learning-based algorithms are used but with additional processing and features included. The results show that SVM Regression performs better than the other learning-based algorithms. The addition of pre-processors implementation and features reduces the amount of external detrimental factors such as noise and outliers which may have allowed a better fit in the non-linear kernel.

## 5.4 Advantages of different models

Figure 20 confirms with the hypothesis in section 1.3 that the hybrid approach outperforms both lexicon approach and learning algorithm approach. The results conclude that the learning algorithm or hybrid approach may be preferred most of the time, with hybrid classifiers gaining an edge over learning algorithms for not having the need to manually annotate every data which consumes too much time and resources.

However, the results are confined to optimal pre-processing procedures done on the dataset which may express a slight bias in the results. Furthermore, results from related works such as the paper from Hu and Liu[9] suggest that lexicon-based classifiers can also achieve a high accuracy. From their experimental results, the lexicon-based classifier has achieved an average accuracy of 84% across several review domains. This presents the possibility of using lexicon-based approach or hybrid approach if manual annotating of the data for learning-based classifier is too expensive in terms of time-cost.

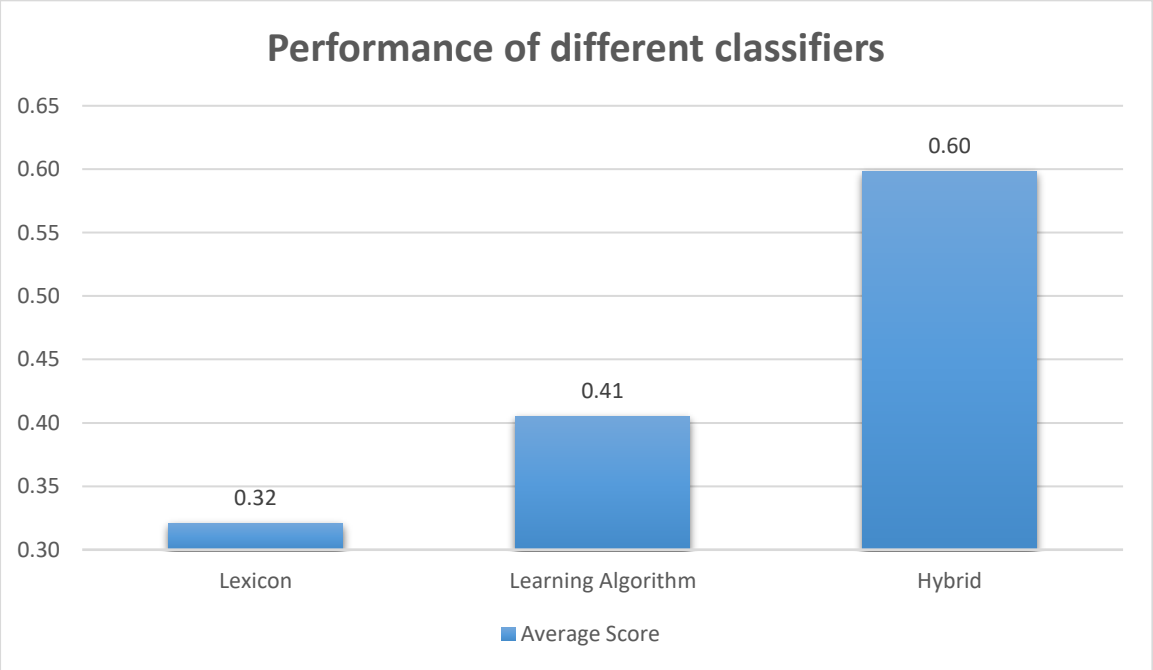


*Figure 20. A bar graph illustrating the performance accuracy of different classifier approaches*

## 5.5 Possible Improvements

A custom implementation of a lexicon that specifically captures slangs or words with emotions that are more commonly found in tweets can be considered. From the previous experimental results in this report, most of the lexicons that are constructed from tweets perform better than general lexicons. This suggests that the quantity of words that is captured by lexicons plays an important part in the author's findings and results. Furthermore, instead of implementing available lexicon resources as used in the experiments, the author suggests creating a self-defined lexicon resource using the dataset itself which may see a better improvement of performance in terms of accuracy.

From combining various procedures of processing the data before classification, natural language processing and feature generation such as implementing stopwords and having extracting n word gram features has proved to aid in improving the accuracy of results to a certain extent. More complicated detection of word structures such as sarcasm and irony may lead to a better performance of the classifiers.

Instead of using a pre-trained word embedding corpus, a new corpus trained using the available data itself can be considered. This is applicable if the amount of data gathered is sufficient and will provide a better vector relation is using a test data set from a similar source.

Lastly, based on the results that have been gathered from various participants of the same task, a new model requiring the use of deep learning methods can be considered. This is being considered that not only the top scorers of the competition, but almost all the competitors who ranked higher have achieved their results using deep learning methods. This is also complemented with the fact that a much larger percentage of participants engaged in using deep learning methods compared to the participants who did a similar task in the previous year.

## 5.6 Conclusion

Based on the results observed from the following experiments, the proposed hybrid approach helps to save a considerable amount of time labelling the data as opposed to the learning-based approach. Furthermore, the wide gap in results comparing lexicon versus both learning-based and hybrid approach suggests that the latter is much preferred when deciding on a model to conduct sentiment analysis. On the other hand, the difference in results between learning-based approach and hybrid approach cannot be a one-off factor in determining an appropriate model as the results demonstrated by the experiments in this report were based off certain constraints such as the source of data and the type of algorithms and the procedures of pre-processing the data. Therefore, applications that are looking to construct a model for sentiment related analysis should consider reviewing their own source of data before determining the optimal solution for their classifier model.

# Appendix

## Complied Results

| Pearson Correlation r | | | | | |
|---|---|---|---|---|---|
| | anger | fear | joy | sadness | Average |
| ***Individual feature sets*** | | | | | |
| word ngrams (WN) | (n=1)0.44<br>(n=2)0.42<br>(n=3)0.37<br>(n=4)0.35 | (n=1)0.51<br>(n=2)0.48<br>(n=3)0.45<br>(n=4)0.43 | (n=1)0.57<br>(n=2)0.57<br>(n=3)0.55<br>(n=4)0.53 | (n=1)0.41<br>(n=2)0.44<br>(n=3)0.43<br>(n=4)0.42 | 0.48<br>0.48<br>0.45<br>0.43 |
| char. ngrams (CN) | 0.40 | 0.37 | 0.52 | 0.37 | 0.43 |
| Stop words (SW) - Rainbow | 0.47 | 0.55 | 0.51 | 0.46 | 0.50 |
| word embeddings (WE) - Edinburgh 100d | 0.45 | 0.43 | 0.57 | 0.43 | 0.47 |
| word embeddings (WE) – Edinburgh 400d | 0.46 | 0.43 | 0.57 | 0.43 | 0.47 |
| All Lexicons (L) | 0.60 | 0.59 | 0.61 | 0.59 | 0.61 |

| | | | | | |
|---|---|---|---|---|---|
| ***Individual Lexicons*** | | | | | |
| Mpqa | 0.12 | 0.24 | 0.30 | 0.13 | 0.20 |
| BingLiu | 0.22 | 0.30 | 0.33 | 0.20 | 0.26 |
| Afinn | 0.33 | 0.30 | 0.37 | 0.23 | 0.31 |
| Sentiment140 | 0.31 | 0.34 | 0.34 | 0.41 | 0.35 |
| NRC-Hash-Sent | 0.37 | 0.27 | 0.38 | 0.39 | 0.35 |
| NRC10 | 0.20 | 0.29 | 0.33 | 0.17 | 0.25 |
| NRC-Expanded | 0.27 | 0.25 | 0.40 | 0.27 | 0.29 |
| NRC-Hash-Emo | 0.50 | 0.49 | 0.45 | 0.47 | 0.48 |
| SentiWordNet | 0.13 | 0.18 | 0.32 | 0.16 | 0.26 |
| SentiStrength | 0.43 | 0.49 | 0.47 | 0.45 | 0.46 |
| ***Combinations of features and pre-processors*** | | | | | |
| WN + CN + WE | 0.40 | 0.46 | 0.52 | 0.38 | 0.44 |
| WN + CN + L | 0.56 | 0.60 | 0.61 | 0.51 | 0.57 |
| WE + L | 0.67 | 0.63 | 0.64 | 0.61 | 0.64 |
| WN + WE + L | 0.61 | 0.65 | 0.64 | 0.55 | 0.61 |
| WN + CN + WE + L | 0.56 | 0.60 | 0.61 | 0.52 | 0.57 |
| WE + L + SW | 0.68 | 0.64 | 0.63 | 0.61 | 0.64 |

| | | | | | |
|---|---|---|---|---|---|
| ***Individual Classifiers with Lexicons (Using WE(Edinburgh) + L + SW)*** | | | | | |
| SVM Regression | 0.68 | 0.64 | 0.63 | 0.61 | 0.64 |
| Random Forest | 0.60 | 0.54 | 0.60 | 0.55 | 0.60 |
| Additive Regression/ Gradient Boosting | 0.66 | 0.59 | 0.60 | 0.57 | 0.61 |
| AdaBoost | 0.66 | 0.59 | 0.60 | 0.57 | 0.61 |
| Bagging (REPTree) | 0.58 | 0.59 | 0.57 | 0.53 | 0.57 |
| Linear Regression | 0.59 | 0.58 | 0.56 | 0.52 | 0.56 |
| ***Individual Classifiers without Lexicons (Using WE(Edinburgh))*** | | | | | |
| SVM Regression | 0.51 | 0.46 | 0.52 | 0.45 | 0.48 |
| Random Forest | 0.46 | 0.41 | 0.50 | 0.36 | 0.43 |
| Additive Regression/ Gradient Boosting | 0.52 | 0.46 | 0.55 | 0.43 | 0.49 |
| AdaBoost | 0.18 | 0.23 | 0.36 | 0.20 | 0.24 |
| Bagging | 0.32 | 0.33 | 0.45 | 0.27 | 0.34 |
| Linear Regression | 0.47 | 0.44 | 0.49 | 0.37 | 0.45 |

# Setup

## Prerequisites

## Prerequisites

Environment: Windows

Hardware Specification: Minimally 8 GB of RAM and 5GB Space Installation

Installation Drive: C Drive

## Installation

1. Clone git repository: https://github.com/derrickpehjh/An-Affect-Intensity-Tool-by-Combining-Lexicon-and-Learning-Based-Approaches

2. Weka - https://www.cs.waikato.ac.nz/~ml/weka/downloading.html

3. Python 3.7 - https://www.python.org/downloads/

4. Java - https://www.java.com/en/download/

5. Associated python libraries
    1. python -m pip install --user numpy scipy sklearn

6. AffectiveTweets Package with external packages (LibLinear and Edinburgh corpus)
    1. java -cp $WEKA_PATH/weka.jar weka.core.WekaPackageManager -install-package AffectiveTweets

    In case of having problems with the Weka packages repository, install the package as follows: C:\Program Files\Weka-3-8

2. java -cp $WEKA_PATH/weka.jar weka.core.WekaPackageManager -install-package https://github.com/felipebravom/AffectiveTweets/releases/download/1.0.1/AffectiveTweets1.0.1.zip

   LibLinear

   java -cp $WEKA_PATH/weka.jar weka.core.WekaPackageManager -install-package LibLINEAR

   Edinburgh Corpus

   https://github.com/felipebravom/AffectiveTweets/releases/download/1.0.0/w2v.twitter.edinburgh10M.400d.csv.gz

   Download the file from the given link to C:\Users\{user}\wekafiles\packages\AffectiveTweets\resources

Note: Please add Python and Java to your Windows Environment to allow the programs to run in command prompt.

## Instructions

1. Go to "#Run" Folder in the git repo folder and execute #Config (Run this first as admin).bat with admin privileges
2. Run any of the 7 scripts available in the" #Run" Folder
3. To see the accuracy results, go to the "#Results" Folder
4. To see the content of the labelled data, go to the "Output" Folder

## Source Codes

https://github.com/derrickpehjh/Hybrid-Affect-Intensity-Tool

# Bibliography


[1] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*. Association for Computational Linguistics, pp. 79–86, 2002.

[2] L. Zhang, R. Ghosh, M. Dekhil, M. Hsu, and B. Liu, "Combining Lexicon-based and Learning-based Methods for Twitter Sentiment Analysis," 2011.

[3] K. Hiroshi, N. Tetsuya, and W. Hideo, "Deeper sentiment analysis using machine translation technology," *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, Geneva, Switzerland, p. 494, 2004.

[4] E.-J. Lee and S. Y. Oh, "To Personalize or Depersonalize? When and How Politicians' Personalized Tweets Affect the Public's Reactions," *J. Commun.*, vol. 62, no. 6, pp. 932–949.

[5] T. M. Sykora; Jackson, "How our tool analysing emotions on Twitter predicted Donald Trump win," *Conversat.*, 2016.

[6] J. Bollen, A. Pepe, and H. Mao, "Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena," *CoRR*, vol. abs/0911.1, 2009.

[7] S. M. Mohammad and F. Bravo-Marquez, "Emotion Intensities in Tweets," *CoRR*, vol. abs/1708.0, 2017.

[8] S. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, "SemEval-2018 Task 1: Affect in Tweets," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 1–17.

[9] M. Hu and B. Liu, "Mining and summarizing customer reviews," *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, Seattle, WA, USA, pp. 168–177, 2004.

[10] V. Hatzivassiloglou and K. R. McKeown, "Predicting the semantic

orientation of adjectives," *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Madrid, Spain, pp. 174–181, 1997.

[11] T. Nasukawa and J. Yi, "Sentiment analysis: capturing favorability using natural language processing," *Proceedings of the 2nd international conference on Knowledge capture*. ACM, Sanibel Island, FL, USA, pp. 70–77, 2003.

[12] A. Devitt and K. Ahmad, *Sentiment Polarity Identification in Financial News: A Cohesion-based Approach*. 2007.

[13] S.-M. Kim and E. Hovy, "Determining the sentiment of opinions," *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, Geneva, Switzerland, p. 1367, 2004.

[14] A. Go, R. Bhayani, and L. Huang, *Twitter sentiment classification using distant supervision*, vol. 150. 2009.

[15] S. Tan, Y. Wang, and X. Cheng, "Combining learn-based and lexicon-based techniques for sentiment detection without using labeled examples," *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, Singapore, Singapore, pp. 743–744, 2008.

[16] E. Kouloumpis, T. Wilson, and J. D. Moore, "Twitter Sentiment Analysis: The Good the Bad and the OMG!" AAAI Press, pp. 538–541, 2011.