ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 12 (2022) No. 5, pp. 73-90
DOI:10.33168/JSMS.2022.0505

Preprocessing Impact on Sentiment Analysis Performance on Malay Social Media Text

Ian Ho[1+], Hui-Ngo Goh[2], Yi-Fei Tan[1]

[1] Faculty of Engineering, Multimedia University, Malaysia
[2] Faculty of Computing and Informatics, Multimedia University, Malaysia

1201402876@student.mmu.edu.my (corresponding author)

Abstract. The preprocessing stage in any natural language processing field has been widely accepted to be a requirement to achieve the best performance in machine learning tasks. In a low resource language like Malay, majority of recent studies had implemented generalized methods for preprocessing like stopword removal using a Malay-translated English stopword list and simple text normalization algorithms. This study was done to explore the extent of impact of preprocessing on supervised sentiment classification in the social media domain, primarily on Malay, while considering various factors like general vs. domain-specific stopword removal, feature vectorizers, and dataset size. The impact of normalization and stopword removal was found to be minor and it was surprising to find that sentiment classification with an unprocessed dataset achieved a similar f1-score in the 85% range to a processed dataset. The choice of feature vectorization method and type of classification model had more positive impact on classification performance.

Keywords: Malay text, pre-processing, sentiment analysis, opinion mining, domain specific.

1. Introduction

Sentiment analysis (also known as opinion mining) is a subset of natural language processing (NLP) that enables the association of emotion to text. The fundamental value of a sentiment analysis model is to significantly reduce the manual labour of combing through large amount of text data and providing sentiment-related insights.

Progress in the field of text NLP has been moving forward rapidly in the past decade with many state-of-art techniques being developed. Word embeddings such as Word2Vec and contextual models like BERT have allowed the study of sentiment analysis to advance to a point where a BERT model tested across various domains achieved an average accuracy of 90.12% on sentiment classification (Du et al., 2020). These studies are usually based on English and have a wealth of resources such as standardised datasets, well maintained lexicons, and preprocessing tools such as the Natural Language Tool Kit (NLTK) library. However, the field of sentiment analysis in other languages, specifically Bahasa Malaysia (BM), has not reached the same level of maturity. This has hindered the progress of research especially when examining text data from a multilingual society like Malaysia.

Over the past decade, several studies were carried out with the aim of developing models and tools for sentiment analysis on BM text. The main challenge faced by researchers was the lack of a standardised dataset and preprocessing tools. Majority of research was focused on lexicon-based solutions (Alsaffar & Omar, 2015; Azlan et al., 2016; Chekima & Alfred, 2018; Imanina Zabha et al., 2019; Sham Awang Abu Bakar et al., 2019) and simple machine learning models (Al-Saffar et al., 2018; Samsudin et al., 2011, 2013). However, the preprocessing steps performed varied between studies without any clear documentation on the impact of each preprocessing step. The focus of this study is to examine the various preprocessing methods employed in previous works on the Malay language to understand their impact on sentiment analysis classification performance.

This paper is organized by first presenting related work in the field of Malay text preprocessing and preprocessing impact on other languages. The experimental setup is then shown alongside supporting theory for preprocessing methods, stopword properties, feature vectorization techniques, and choice in supervised classifier. This is followed by the results and discussion section and is concluded after.

2. Related Works

In this section, previous work done in the field of Malay text preprocessing is presented. Additionally, preprocessing methods that were implemented in sentiment classification studies are also mentioned. One of the earlier studies performed on the Malay language is by Baldwin and Awab (Baldwin & Ad Awab, 2006). They developed a corpus analysis tool that incorporated a tokenizer, lemmatizer and a

partially completed part-of-speech (POS) tagger. The lemmatizer[1] was based on the KAMI lexicon (KAmus Melayu-Inggeris) (Quah et al., 2013) and was reported to be 85% accurate at lemmatizing Malay text.

Samsudin et al (Samsudin et al., 2011) performed exploratory research in opinion mining on online Malay movie reviews without any preprocessing. This early work did not have access to any preprocessing tools as none were available for use in the Malay language. After achieving 68% accuracy on a linear support vector machine (SVM) classifier with Term Frequency – Inverse Document Frequency (TFIDF) weights, they hypothesized that preprocessing was an important step in improving their classifier's performance. The researchers followed up with another study describing a rule-based normalization algorithm to standardise common noisy word patterns observed in Malay social media (Samsudin et al., 2012). These rules were defined by observing the top 5000 noisy terms in their corpus and used to transform text that matched those rules into their corrected form. For example:

$$\text{yg} \rightarrow \text{yang},$$

$$\text{anak2} \rightarrow \text{anak-anak},$$

$$\text{askm} \rightarrow \text{assalamualaikum}$$

They reported an increase of 5% in accuracy when text was preprocessed in this manner.

Saloot et al (Saloot et al., 2014) performed a comprehensive study that proposed a normalization architecture which standardised noisy text with reference to a colloquial dictionary and Bahasa WordNet, affix stemming, and translation of English words to Malay. They reported a BLEU score of 0.81, indicating that the normalized text was transformed with high accuracy from the reference noisy text.

Other than text normalization and correction, stopword removal was another text preprocessing technique commonly used. Stopwords are words that do not carry sentiment or contextual information. However, unlike the English language, the Malay language had no commonly accepted, precompiled stopword list. Additionally, the lack of open-source tools for spelling correction and text normalization meant that precompiled Malay stopword lists translated from English would not be sufficient to account for the scale of noise found in informal Malay text. Chekima (Chekima & Alfred, 2016) attempted to alleviate this by proposing a statistically generated stopword list from a large corpus obtained from Dewan Bahasa and Pustaka (DBP); The Institute of Language and Literature for Malay. The statistical method involved an aggregation of word frequency, word entropy, and word variance calculations to produce a general stopword list.

It is commonly agreed that text normalization, spelling correction, stemming/lemmatization, and stopword removal in general will increase performance

[1] github.com/averykhoo/malay-toklem/tree/master/eval

in text analysis tasks. In the case of sentiment analysis on Malay text, these studies (Al-Saffar et al., 2018; Alsaffar & Omar, 2015; Azlan et al., 2016; Chekima & Alfred, 2018; Imanina Zabha et al., 2019; Jun Ying et al., 2020; Sham Awang Abu Bakar et al., 2019) reported their preprocessing steps but did not compare the final results to a baseline prior to preprocessing. In a low resource language like Malay, it is important to explore the effects of preprocessing on the performance of sentiment classification. This is because it will enable researchers to avoid large unnecessary efforts to develop their own preprocessing modules in future studies due to the lack of open-source tools.

Saif et al (Saif et al., 2014) explored the impact of different English stopword lists on the same dataset for a sentiment classification task. The lists consisted of precompiled commonly accepted stopword lists, a list generated based on Zipf's Law, a list generated based on Term Based Random Sampling, and a list generated with the Mutual Information method. Despite the popular use of precompiled stopword lists in text preprocessing, Saif observed that it has a negative impact on sentiment classification performance. Additionally, Saif concluded that stopword removal generally had low impact on sentiment classification. This finding was echoed by Zhao's (Zhao, 2015) findings in which the study investigated different preprocessing methods and their effect on sentiment classification performance in English Twitter text. The observations were that URL, numbers and stopword removal did not affect the performance of classifiers. However, acronym expansion (e.g. lol → laugh out loud) did positively affect performance, albeit only on one of the classifiers used (Naïve Bayes) and only by 6%.

Pradana and Hayaty (Pradana & Hayaty, 2019) carried out a similar study on Indonesian Twitter text where they compared the impact of preprocessing by varying the use of stemming and stopword removal. There were no major differences in performance between the use of stemming + stopword removal and no stemming + no stopword removal. However, they did observe a minor decrease in performance on their linear SVM classifier when only stopword removal was performed.

3. Experimental Setup

Figure 1 shows the overall framework. Sentiment classification will be carried out with variations of normalization, stopword removal with multiple lists (precompiled and statistically generated), different feature vectorization models and sentiment models. These variations will be discussed below.

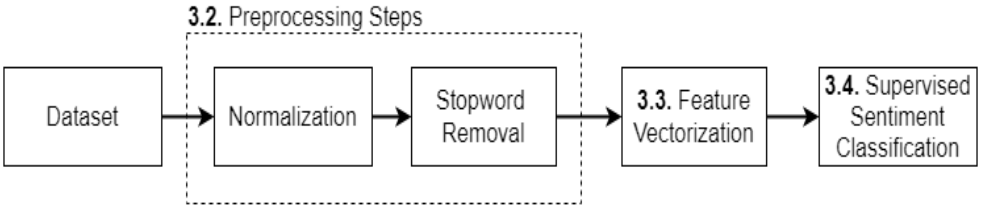

Fig. 1: Overview of preprocessing setup.

3.1. Dataset

In general, sentiment classification is sensitive to the domain it is being used in. This is also true for stopword removal effectiveness. In this paper, we study the effect preprocessing on two very different datasets to validate that the effect of preprocessing is not confined to a certain type of text or language. The MALAYA Twitter dataset (Husein, 2018) contains two million tweets and Pang & Lee's movie review dataset only contains two thousand movie reviews. To investigate if the impact of preprocessing is different on different dataset sizes, a 20000-tweet subset of the MALAYA Twitter dataset was sampled. Although there are 10 times more documents in the MALAYA subset, the documents in the Twitter domain are generally much shorter in length compared to a typical movie review. In Pang & Lee's dataset, the reviews had around 600 words on average contrasted with MALAYA's dataset that had 15 words per tweet on average. The MALAYA subset size was chosen to account for the document length difference and to attain a similar vocabulary size as Pang & Lee's dataset. Therefore, the baseline tests and experiments were performed on:

- MALAYA Twitter Dataset (2 million Malay tweets) – **MAL2M**[2]
- Subset of MALAYA Twitter Dataset (20000 Malay tweets) – **MAL20K**
- Pang and Lee's Movie Review Dataset (2000 English movie reviews) – **P&L**[3]

3.2. Preprocessing steps

3.2.1. Normalization

The first part of preprocessing is denoted as 'Normalization' in Figure 1. In TABLE 1, a sample from each dataset and their normalized form is shown. The methods implemented defined under normalization in this paper are listed below:

- Special character and number removal.
- URLs, social media references (@mentions, #hashtags, etc.) and html tags removal.
- Stemming/lemmatizing
- Malay: Baldwin's malay-toklem stemmer (Baldwin & Ad Awab, 2006)
- English: NLTK WordNet lemmatizer (Bird et al., 2009)
- Duplicate character removal (e.g. coooool → cool).
- Simple abbreviation/contraction expansion
- Malay: xbaik → tidak baik
- English: don't → do not

[2] https://github.com/huseinzol05/malaya
[3] https://www.nltk.org/nltk_data/

Table 1: A sample from each dataset showing an unprocessed and normalized form of the sentence.

Dataset	Unprocessed	Normalized
MALAYA Twitter	@mention Nak xnak berjual ps4, jarang2 main. Tapi nanti bosan pulak https://t.co/lbcLuUl0wH	nak tidak nak jual ps4 jarang main tapi nanti bosan pulak
Pang and Lee's Movie Reviews	There wasn't a lot of censorship when it showed in the cinema.	there was not a lot of censorship when it show in the cinema

3.2.2. Stopword removal

Stopwords do not convey any significant information in terms of sentiment analysis (e.g., common stopwords – 'is', 'the'). In this study, two types of stopword lists will be used. The first type is a precompiled list. The Malay and English stopword lists are:

- Malay: MALAYA stopword list (1057 stopwords)2
- English: NLTK stopword list (179 stopwords)

The second type of stopword list is a domain-specific list generated from the corpus directly using aggregated statistical measures also used in other related work (Asubiaro & Latunde, 2013; Chekima & Alfred, 2016; Tijani & Onashoga, 2017). These statistical measures are:

3.2.2.1. Word frequency

Zipf's Law states that the frequency of any word is inversely proportional to its frequency rank. Stopwords are often repeated multiple times within a sentence, between sentences, and across the corpus, thus, appear frequently.

3.2.2.2. Word variance

$$s^2 = \frac{\sum(x-\mu)^2}{N} \quad (1)$$

s^2: variance
x: relative frequency of word
μ: mean relative frequency
N: total number of documents in corpus

In statistics, variance is a measure of the spread of a random variable from their sample's mean as seen in Equation (1). A higher variance implies a flatter and more even distribution across a sample. In text analysis, word variance is based off its

frequency. Stopwords tend to appear frequently within and across documents in a corpus and hence have a higher variance compared to other words.

3.2.2.3. Word entropy

$$H(w) = -\sum_{i=1}^{n} P(w_i) log P(w_i) \quad (2)$$

H(w): entropy of w
n: number of occurances of w in corpus
P(w): probability of occurance in a document calculated as:

$$\frac{frequency\ of\ occurance\ w\ in\ doc}{frequency\ of\ occurance\ in\ w\ corpus}$$

As defined by Claude Shannon in 1948, entropy in information theory is a measure of randomness and amount of information in a signal. The value of entropy in Equation (2) is defined to be high when the incoming signal has a low amount of information and is maximised when the probability of each outcome is distributed uniformly. The contrast with word variance is that entropy is a measure based on probability. However, the concept is similar. Stopwords have a higher probability of appearing across each document in a corpus and hence tends towards an even probability distribution across the whole corpus. As stated earlier, stopwords also carry little to no information. Therefore, a high entropy can indicate a potential stopword.

Table 2: Domain specific stopword lists' sizes statistically generated.

Dataset	No. of Stopwords
MALAYA Twitter	298
Pang & Lee	329

A list of potential stopwords from each statistical measure is produced. Then, a single domain-specific stopword list is formed by taking the intersection of the top 500 words in each list. This is because most stopword lists range between 200 to 450 words [14]. The stopword lists generated are then manually filtered to remove sentiment words. The number of stopwords leftover after filtering is present in TABLE 2.

3.3. Feature vectorization

The final step prior to machine learning is to prepare the processed datasets in a format that can be understood by a classification model. In NLP, this is done by transforming words or documents into a numerical vector format. To identify the different levels of impact through the choice of feature vectorizers compared to preprocessing, Term Frequency – Inverse Document Frequency, Word2Vec and Doc2Vec were used.

3.3.1. Term frequency – inverse document frequency (TF-IDF)

The TF-IDF technique computes a weight for each unique word in the corpus to form a vector that is essentially a weight Bag-of-Words. This vector corresponds to the number of unique words present in the corpus and hence, is prone to the curse of dimensionality. The weight of a term in a document is the product of term frequency (TF) and inverse document frequency (IDF):

$$TF - IDF(t,d) = TF(t,d) * IDF(t) \quad (3)$$

t: term
d: document

$$TF(t,d) = \frac{f(t)}{n} \quad (4)$$

f(t): frequency of t in document
n: total number of terms in document

$$IDF(t) = log\left(\frac{1+N}{1+DF(t)}\right) + 1 \quad (5)$$

N: total number of documents in corpus
DF(t): number of documents term t is present

As shown in Equations (3), (4), and (5), term frequency, TF(t, d) computes the normalised frequency of a term in the document with respect to the total number of terms in the document. Term frequency alone outputs high values for terms that occur very frequently which gives stopwords a higher weight (higher weight, higher importance). To associate higher weights to terms with more significance than stopwords, the IDF(t) component is required to invert the relationship between the term frequency score and term importance. The log function is used to dampen the scaling effect of the IDF(t) with corpus size. A value of '1' is added to both the numerator and denominator to prevent zero divisions. This can be thought of adding an extra document to the corpus that contained every term in the vocabulary. Another '1' is added to the log component to account for terms that occur in every document (causing the log component to output 0) so that it does not get ignored.

3.3.2. WORD2VEC

In 2013, Mikolov et al developed the Word2Vec model (Mikolov et al., 2013). The model is used to compute vector representations for each word in a corpus. Mikolov et al introduced two variations of the Word2Vec model in their study, the Continuous Bag-of-Words (CBOW) model and Continuous Skip-gram model, of which the latter was used in this study. Contrasted with the TF-IDF model, Word2Vec does not suffer from the curse of dimensionality when vocabulary size increases. The vector representations are defined when the model is initialised. A higher number of

dimensions increases the computational complexity but also increases the quality of relationships represented by the vectors.

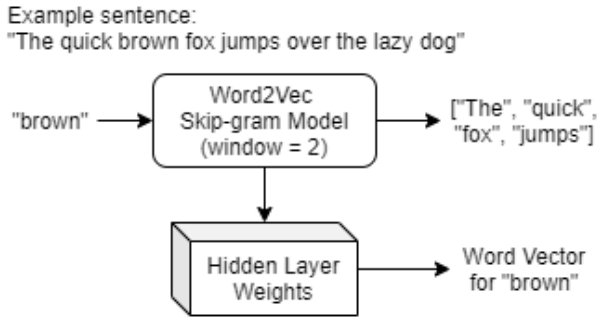

Fig. 2: Word2Vec Skip-gram model. Weights learned from predicting the surrounding words represent the word vector.

The skip-gram model generates a vector representation for a word from the model's hidden layer weights learned by predicting the surrounding context of the word within a certain range. This is visualised in Figure 2. The word embeddings formed in this way produces intuitive relationships between words, although this is dependent on the size and quality of corpus it is trained on. Words with similar linguistic properties are close in the vector space. A common example exhibiting the relationship between words in the vector space is king + woman - man = queen.

In this study, the word embeddings for each word in a document were averaged. This is required to account for the varying document length. As for the hyperparameters used: min_count = 5, window = 5, size = 300, alpha = 0.03, negative = 20.

Word vector averaging has been criticised by Le in [24] because it causes the vector representation in a document to lose word-order just as Bag-of-Word models do. Hence, this study also employed Doc2Vec which will be looked at next.

3.3.3. DOC2VEC

Le and Mikolov expanded on Word2Vec to produce more accurate vector representations for paragraphs (Le & Mikolov, 2014) due to the weakness of averaging word vectors. Inspired by the Word2Vec CBOW model, the Doc2Vec Distributed Memory Model of Paragraph Vectors (PV-DM) was developed to compute a paragraph vector in a similar way.

The Word2Vec CBOW model randomly initialises word vectors for each word and performs a prediction task to predict a word given the context around it. The randomized word vectors are then iterated over, maximizing an average log probability, $\frac{1}{T}\sum_{t=k}^{T-k} log\, p(w_t|w_{t-k},\dots,w_{t+k})$, to obtain the final word vector

representation for each word. Since word semantics could be captured resulting from the prediction task, the Doc2Vec PV-DM model includes a randomly initialised vector representation for each document in the corpus to be included into the same prediction task.

Finally, the vector representation of each document can be treated as a feature for the document and can be used in the same way averaged Word2Vec embeddings are used in classification tasks. As for the hyperparameters used to train the Doc2Vec model in this study: min_count = 5, vector_size = 300, alpha = 0.065, negative 5.

3.4. Supervised sentiment classification

Two different supervised classifiers will be implemented in this study; A linear classifier, binary logistic regression, and a non-linear one, random forest. The main goal of preprocessing is to reduce the amount of noise being fed to the model. Thus, these basic models were chosen to observe their performance difference and robustness to noise.

Logistic regression is composed of independent variables that describes the linear relationship to a dependent variable. To put it simply, since this study's focus is not on how logistic regression functions, the model is set up to determine a set of weights that minimizes a loss function when predicting a class compared to its true label. The important point to know is that logistic regression calculates a linear boundary separating the two classes. This means that logistic regression will perform poorly if a clear linear boundary cannot be formed between the two classes. An advantage of using logistic regression is that it outputs the predicted class as a probabilistic value which can then be observed to understand if the classes are linearly separable or not. The important hyperparameter configured for this model was to use L2 regularization which adds a coefficient penalty to the higher terms in the model.

A simple decision tree is like a binary tree, where starting at the root node, the tree is recursively split through a series of decision nodes until the predicted value is found. This is the non-linear aspect in the random forest model. A single decision tree is prone to overfitting because of its nature and does not perform well in unseen scenarios. To overcome this, an ensemble of decision trees is used which forms the base of the random forest model and 'bagging' is usually implemented to smoothen out the decision boundary and avoid overfitting. 'Bagging' is randomly sampling with replacement where each decision tree is trained on a random subset of data as well as feature variables. Then, each predicted value from the ensemble of decision trees will then be averaged to output the predicted class. The number of estimators set for the model trained in this study was n_estimators = 100, and the max depth allowed was not limited.

4. Experimental Results

All experiments were done with a 75/25 train-test split with no overlap. The baseline tests were done with the datasets in their original form with no normalization and stopword removal. For the rest of the experiments, the dataset was normalized (see B. for normalization steps). All combinations of stopword lists, feature vectorization methods, and classification models were performed in the experiments.

Table 3: F1-score of experimental results. SR is an abbreviation for stopword removal.

LOGISTIC REGRESSION				
VECTORIZER	PROCESSING COMBINATION	MAL2M	MAL20K	P&L
TF-IDF	Baseline	-	0.89	0.85
	Normalised (N)	-	0.90	0.84
	N + General Stopword Removal	-	0.89	0.83
	N + Statistical Stopword Removal	-	0.91	0.85
Word2Vec	Baseline	0.60	0.62	0.63
	Normalised (N)	0.60	0.60	0.60
	N + General Stopword Removal	0.60	0.59	0.62
	N + Statistical Stopword Removal	0.59	0.59	0.64
Doc2Vec	Baseline	0.84	0.77	0.85
	Normalised (N)	0.85	0.83	0.84
	N + General Stopword Removal	0.84	0.82	0.82
	N + Statistical Stopword Removal	0.84	0.85	0.83
RANDOM FOREST				
VECTORIZER	PROCESSING COMBINATION	MAL2M	MAL20K	P&L
TF-IDF	Baseline	-	0.84	0.74
	Normalised (N)	-	0.85	0.78
	N + General Stopword Removal	-	0.88	0.79
	N + Statistical Stopword Removal	-	0.89	0.84
Word2Vec	Baseline	0.71	0.63	0.63
	Normalised (N)	0.71	0.63	0.57
	N + General Stopword Removal	0.72	0.63	0.58
	N + Statistical Stopword Removal	0.71	0.63	0.60
Doc2Vec	Baseline	0.83	0.74	0.72
	Normalised (N)	0.83	0.79	0.73
	N + General Stopword Removal	0.83	0.81	0.70
	N + Statistical Stopword Removal	0.84	0.82	0.75

Table 4: Vocabulary size of unprocessed and normalized corpora.

Dataset	Vocabulary Size	
	Unprocessed	Normalized
MAL2M	400k	170k
MAL20K	52k	27k
P&L	39k	34k

Table 5: Effectiveness of stopword removal between general and statistical stopword lists.

Dataset	Stopword List	Word Count Reduction (%)
MAL2M	MALAYA (General)	34.5
	Statistical	40.1
P&L	NLTK (General)	47.2
	Statistical	53.8

The MAL2M dataset was not experimented on with the TF-IDF vectorizer. The TF-IDF vectorizer produces a vocabulary vector with each unique word representing one dimension in the vector. The issue with this is that a vocabulary size of 100,000 words will be represented by a 100,000-dimension vector in TF-IDF and so due to memory limitations, this method was not used for the MAL2M dataset.

5. Discussion

An initial look at the results indicates that normalization and stopword removal have very little impact on the sentiment analysis task. The differences are mainly seen between feature extraction methods. Other than that, the performance difference between classification models is generally due to dataset size. Despite all of this, there still are some notable points that can be discussed.

5.1. Classification performance difference between vectorizers

Firstly, the classification performance with Word2Vec will be addressed. It is suspected that averaging word vectors in a document might be the reason that the classifiers are unable to define a boundary between the positive and negative classes. This is clearly the case with logistic regression, a linear classifier. This is further supported by the results showing an increase in the f1-score, by 11%, when averaged word2vec vectors in MAL2M were classified with the random forest classifier, which is non-linear.

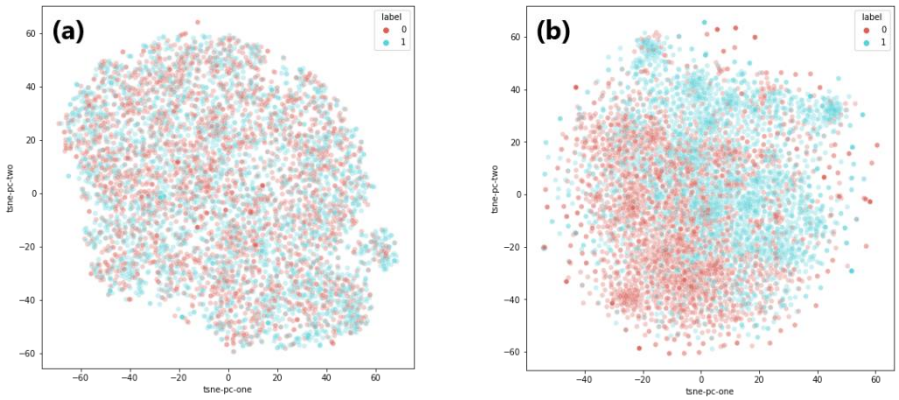

Fig. 3: (a) - TSNE representation of
averaged Word2Vec MALAYA Twitter dataset vectors and their sentiment polarity
(perplexity: 50, iteration: 4000, input: 20 PCA components with 98.2% explained variation)

(b) - TSNE representation of averaged Doc2Vec MALAYA Twitter dataset vectors and their sentiment polarity (perplexity: 50, iteration: 4000, input: 50 PCA components with 27.5% explained variation).

The lack of performance increase in the tests with MAL20K and P&L with the random forest classifier is probably due to the small dataset size. The random forest classifier could not determine a clear boundary between the classes with limited data points because of the high non-linearity of the word2vec vectors.

When using TSNE to plot the vectors from the various feature vectorizers, it can be clearly observed as to why Doc2Vec was performing better compared to Word2Vec in Figure 3. This visualisation supports the study by Le in which Doc2Vec was proposed where vector averaging will not perform as well because the semantics learned about each word is lost when averaged (Le & Mikolov, 2014). No class separation can be found on the Word2Vec TSNE plot (Figure 3a). Conversely, a rough linear separation can be observed in the Doc2Vec TSNE plot (Figure 3b). The same TSNE visualisation could not be done for the TF-IDF vector as it is a large sparse dataset and is computationally costly to fit a vector with the size of a corpus' vocabulary into the dimension reduction model.

5.2. Choice of classification model and feature vectorizer

In terms of classification models for sentiment analysis, it is commonly accepted that there is no one 'best' classifier for the task. This can be seen in the results as well that change in dataset size and feature vectorizers can affect the results significantly. In this study, the main factors identified for choice in classification model is the processing time. When observing the time taken for each model to be trained, it varied between datasets due to dataset and vocabulary size. However, the random forest model consistently took more time to train (up to 10x longer) compared to the logistic regression model.

As for the feature vectorizers, TF-IDF produced vectors that performed the best for the classification tasks with a logistic regression model. To visualise the difference between all three methods, a plot of the probability distribution output from the logistic regression model for each method can be seen in Figure 4. Figure 4b and Figure 4c reflect the TSNE plot in Figure 3 well.

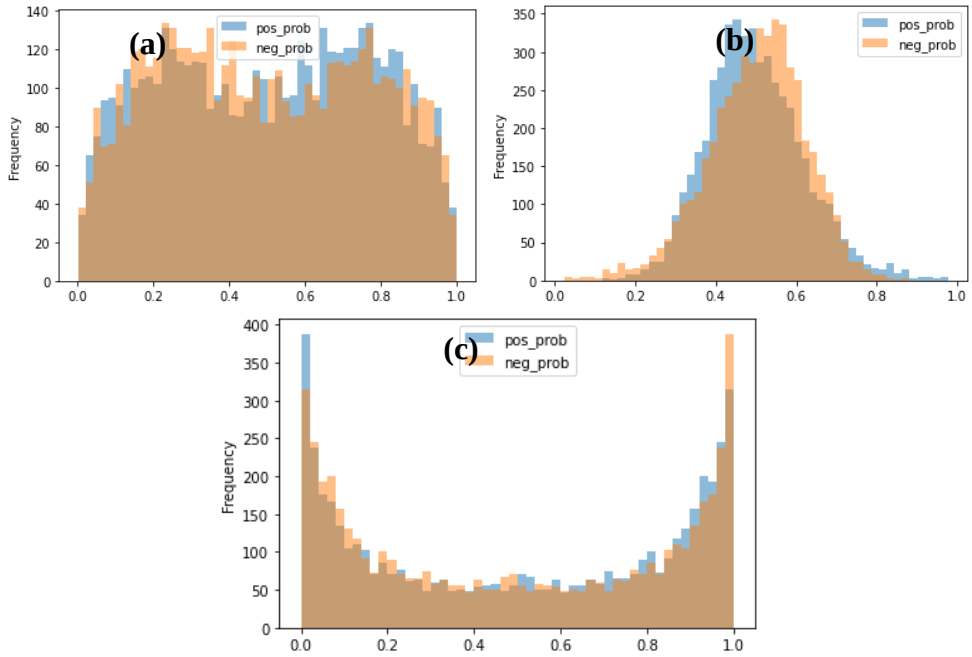

Fig. 4: Logistic regression probability distribution with (a) TF-IDF, (b) Word2Vec, and (c) Doc2Vec

The averaged Word2Vec vectors were not linearly separable and hence, as seen in Figure 4b, the logistic regression model could not distinguish the class apart as most of the probability of confidence appeared at the threshold. Conversely, when Doc2Vec was used, the logistic regression model output probabilities that were skewed to both ends (Figure 4c). This shows that averaged Word2Vec vectors might not be suitable for linear models when performing sentiment classification. This is supported by the fact that there is an increase of around 10% in the f1-score when averaged Word2Vec vectors were classified with the random forest model which is non-linear. Therefore, choice in classification model will depend on feature vectorizer used. In this study, Doc2Vec and TF-IDF were shown to be suitable vectorizers for linear classification models.

In Figure 4a, the probability distribution output of the logistic regression model trained with TF-IDF vectors displays a flatter distribution across the plot. In future work, it will be interesting to see if these distributions can be associated to a multiclassification task where a flatter distribution could imply different levels of sentiment positivity and negativity in a document.

5.3. Normalization impact

When comparing classification results between an unprocessed dataset and a normalized dataset, the impact on the classification performance is present but minor. In most cases, normalization contributed to around a 1% in f1-score increase which

is not significant. However, the biggest impact normalization had was in the reduction of vocabulary size as seen in TABLE 4.

A decrease of up to 67.5% in feature dimensions through normalization was observed for the MAL2M dataset whilst unaffecting the performance of a classifier. This is an improvement in terms of optimization. This is because a reduction in vocabulary size directly correlates to a reduction in processing and training time for the word embedding models (Word2Vec and Doc2Vec). Additionally, for TF-IDF, a smaller vocabulary would mean a lower dimensional vector representing each document in the dataset.

The reduction in vocabulary size on the P&L English movie review dataset was not as drastic because the movie review domain is generally less noisy in terms of abbreviation and slang compared to a twitter dataset. It could also be due to the Malay language on social media being generally noisier compared to the English language in the same space (Abu Bakar et al., 2020).

5.4. Stopword removal impact

The experimental results show that stopword removal has some impact on the sentiment classification performance, albeit low. However, it is interesting to observe that employing a precompiled list for stopword removal decreases classification performance and in contrast, a domain-specific stopword list generated statistically improves classification performance slightly.

The biggest impact from stopword removal can be observed in TABLE 3 with the random forest classifier on TF-IDF vectors from the P&L dataset. The classification f1-score increased by 6% when comparing the experiment that had no stopword removal to the one with a domain-specific stopword removal process. A similar pattern is seen as well with the same setup but on the MAL20K dataset with an increase of 4% in f1-score. The same pattern, where domain-specific stopword removal improves classification performance slightly, can be seen with Doc2Vec as well for these two smaller datasets. This could be associated with the reduction in dimensionality when removing stopwords. For the MAL2M dataset, stopword removal had no significant impact on classification performance.

Stopword removal also contributed to a reduction in the overall word count. This optimizes overall processing and word embedding model training time. As seen in Table 5, the stopword list generated from the three statistical metrics performed better at reducing the number of stopwords present in each dataset. Therefore, the impact of stopword removal on classification is only significant for smaller datasets. Finally, stopword removal should be done using a domain-specific stopword list as shown in the results where a generalised stopword list would negatively impact the classification performance.

6. Conclusion

In conclusion, this study was to investigate the impact of normalization and stopword removal on sentiment analysis. The focus was on Malay text. It can be concluded that normalization plays a bigger role in terms of optimizing processing times by reducing feature dimensions (in terms of vocabulary size) by up to 67%. As for stopword removal, it was shown that the impact was only significant for smaller datasets and that domain-specific stopword removal increases the classification performance, whereas a generalised stopword list reduces classification performance slightly. Although stopword removal had the least amount of impact on classification performance, these findings do not conclude that stopword removal can be dismissed. Rather, this study presents the level of impact each processing step contributes to the sentiment classification performance.

A recent study by Jun Ying et al in 2020 trained a deep learning convolutional neural network on the same MAL2M dataset [15]. In this study, the authors performed preprocessing steps such as removing hashtags, numbers, html tags and urls, lower-casing, stemming, lemmatization, general stopword removal, and tokenization. The best result reported was an accuracy of 77.6%. It was surprising to observe that a simpler method where a random forest model was trained could perform better with an f1-score of 0.84. This supports the conclusion in the following paragraph.

For a low resource language like Malay, the choice of feature vectorization method and type of classification model will have more impact on classification performance in sentiment analysis tasks compared to preprocessing methods implemented. Therefore, the degree of positive impact on classification performance in descending order is feature vectorization, classification model, normalization method, and stopword removal.

References

Abu Bakar, M. F. R., Idris, N., Shuib, L., & Khamis, N. (2020). Sentiment analysis of noisy malay text: state of art, challenges and future work. *IEEE Access*, *8*, 24687–24696. DOI: https://doi.org/10.1109/ACCESS.2020.2968955.

Al-Saffar, A., Awang, S., Tao, H., Omar, N., Al-Saiagh, W., & Al-bared, M. (2018). Malay sentiment analysis based on combined classification approaches and Senti-lexicon algorithm. *PLOS ONE*, *13*(4), e0194852. DOI: https://doi.org/10.1371/journal.pone.0194852.

Alsaffar, A. & Omar, N. (2015). Integrating a Lexicon based approach and K nearest neighbour for Malay sentiment analysis. *Researchgate.Net*. DOI: https://doi.org/10.3844/jcssp.2015.639.644.

Asubiaro, T. & Latunde, E. (2013). Entropy-Based Generic Stopwords List for

Yoruba Texts Entropy-Based Generic Stopwords List for Yoruba Texts Asubiaro, Toluwase Victor. In *International Journal o.f Computer and Information Technology*. www.ijcit.com1065

Azlan, A., Tan, Y. F., Lam, H. S., & Soo, W. K. (2016). Sentiment analysis for telco popularity on twitter big data using a novel Malaysian dictionary. *Frontiers in Artificial Intelligence and Applications*, *282*, 112–125. DOI: https://doi.org/10.3233/978-1-61499-637-8-112.

Baldwin, T., & Ad Awab, S. '. (2006). *Open Source Corpus Analysis Tools for Malay*. http://www.anu.edu.au/asianstudies/proudfoot/MCP/Q/

Bird, Steven, Lopwer, E., & Klein, E. (2009). *Natural Language Processing with Python*. O'Reilly Media Inc.

Chekima, K., & Alfred, R. (2016). An automatic construction of Malay stop words based on aggregation method. *Communications in Computer and Information Science*, *652*, 180–189. DOI: https://doi.org/10.1007/978-981-10-2777-2_16.

Chekima, K., & Alfred, R. (2018). Sentiment analysis of malay social media text. *Researchgate.Net*, *488*, 205–219. DOI: https://doi.org/10.1007/978-981-10-8276-4_20.

Du, C., Sun, H., Wang, J., Qi, Q., & Liao, J. (2020). Adversarial and domain-aware BERT for cross-domain sentiment analysis. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 4019–4028. DOI:https://doi.org/10.18653/v1/2020.acl-main.370.

Husein, Z. (2018). *MALAYA*. GitHub. https://github.com/huseinzol05/malaya.

Imanina Zabha, N., Ayop, Z., Anawar, S., Hamid, E., & Zainal Abidin, Z. (2019). Developing cross-lingual sentiment analysis of malay twitter data using lexicon-based approach. In *IJACSA) International Journal of Advanced Computer Science and Applications*, 10(1). www.ijacsa.thesai.org.

Jun Ying, O., Mun, M., Ahmad Zabidi, im, Ramli, N., & Ullah Sheikh, U. (2020). Sentiment analysis of informal Malay tweets with deep learning. *IAES International Journal of Artificial Intelligence (IJ-AI*, *9*(2), 212–220. DOI: https://doi.org/10.11591/ijai.v9.i2.pp212-220.

Le, Q. & Mikolov, T. (2014). Distributed representations of sentences and documents. In *31st International Conference on Machine Learning, ICML 2014*, 4.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. http://ronan.collobert.com/senna/.

Pradana, A. W. & Hayaty, M. (2019). The effect of stemming and removal of stopwords on the accuracy of sentiment analysis on indonesian-language texts.

Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, *4*(4), 375–380. DOi: https://doi.org/10.22219/kinetik.v4i4.912.

Quah, C. K., Bond, F., & Yamazaki, T. (2013). Design and construction of a machine-tractable Malay-English Lexicon design and construction of a machine-tractable Malay-English Lexicon. January 2002.

Saif, H., Fernandez, M., He, Y., & Alani, H. (2014). On stopwords, filtering and data sparsity for sentiment analysis of twitter. In *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*. http://lrec2014.lrec-conf.org/en/.

Saloot, M. A., Idris, N., & Mahmud, R. (2014). An architecture for Malay tweet normalization. *Information Processing and Management*, *50*(5), 621–633. DOI: https://doi.org/10.1016/j.ipm.2014.04.009.

Samsudin, N., Puteh, M., & Hamdan, A. R. (2011). Bess or xbest: Mining the Malaysian online reviews. *Conference on Data Mining and Optimization*, 38–43. DOI: https://doi.org/10.1109/DMO.2011.5976502

Samsudin, N., Puteh, M., Hamdan, A. R., Zakree, M., & Azri, A. (2012). *Normalization of Common NoisyTerms in Malaysian Online Media*. http://www.cari.com.

Samsudin, N., Puteh, M., Hamdan, A. R., Zakree, M., & Nazri, A. (2013). Mining opinion in online messages. In *IJACSA) International Journal of Advanced Computer Science and Applications*, 4(8). www.ijacsa.thesai.org.

Sham Awang Abu Bakar, N., Aziehan Rahmat, R., & Faruq Othman, U. (2019). Polarity classification tool for sentiment analysis in Malay language. *IAES International Journal of Artificial Intelligence (IJ-AI*, *8*(3), 258–263. DOI: https://doi.org/10.11591/ijai.v8.i3.pp258-263.

Tijani, O. D. & Onashoga, S. A. (2017). An auto-generated approach of stop words using aggregated analysis. *13th International Conference on Information Technology Innovation for Sustainable Development*, *July*.

Zhao, J. (2015). Pre-processing boosting twitter sentiment analysis? *Proceedings - 2015 IEEE International Conference on Smart City, SmartCity 2015, Held Jointly with 8th IEEE International Conference on Social Computing and Networking, SocialCom 2015, 5th IEEE International Conference on Sustainable Computing and Communic*, 748–753. DOI: https://doi.org/10.1109/SmartCity.2015.158.