Exploring Edit Distance for Normalising Out-of-Vocabulary Malay Words on Social Media

Raja Roza Athirah[1], *Lay-Ki* Soon[*, 1, 2], *Su-Cheng* Haw[1]

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

[2] School of Information Technology, Monash University, Malaysia

Abstract. Users interact using short-formed words and abbreviations and this results in a message full of noisy words that are not recognized by the system's knowledge. The aim of this research is to overcome the limitations that still bar the progression of normalizing Malay noisy words from social media platforms. The testing data gathered is 25,000; 15,000 Tweets from Twitter and 10,000 comments from Facebook respectively. Pre-processing steps were carried out to clean the entire dataset which consists of unique 179,786 words. 36,587 out-of-vocabulary (OOV) Malay terms were then extracted and checked against an in-vocabulary (IV) Malay corpus using the Levenshtein edit distance formula and character manipulation rules. The resultant output is 3,964 unique IV Malay words. Based on the results, the usage of edit distance and rules can be further improved to elevate the normalisation of the ever changing colloquial terms of the Malay language.

1 Introduction

Social media has become one of the most accessed platforms in today's technologically advanced society. Malaysia, in particular, is ranked among the top ten countries whose citizens frequently use social media applications on a daily basis such as Facebook and Twitter. Due to the aforementioned applications' expansion of platforms, users can nowadays communicate through their smartphones. Hence, for the sake of convenience, users tend to use "texting language" and minimize their efforts in typing long and proper words. This gives rise to words that appear alien to machine's understanding.

Both Twitter and Facebook serve as important resources for many Natural Language Processing (NLP) studies such as information retrieval, sentiment analysis and the focus of this particular study, text normalisation. This is not to be confused with spell correction studies, where they share similarities in their attempts to detect and correct OOV words. Normalisation gains the upper hand in which the context of the text and its features should be recognisable [1].

The normalisation process proposed in this paper acts as a model to standardize noisy terms. By taking into account the usage of the Levenshtein distance, the process will undergo a series of actions consisting of the splitting of characters, transposition, deletion and addition. These actions will result in word comparisons between the OOV Malay words and the Malay corpus. The resultant output will return a document with the IV Malay words. Besides, the implementation of sets of rules that further heightens the precision between OOV and IV matches. Sections 1.1 and 1.2 present thorough explanations regarding the aforementioned processes and methods.

The remainder of this paper is organized as follows: Section 2 discusses background studies and related works on normalizing noisy words. Section 3 presents the theoretical framework and methodology to improve on the limitations faced by the previous works [2]. Section 4 presents the implementation of the overall normalisation process using the experimental dataset. Section 5 presents the evaluation and the findings after the normalisation process. Section 6 sees to the discussions pertaining to the results obtained and lastly, section 7 concludes the paper.

1.1 Levenshtein Edit Distance

The Levenshtein Distance is the most common metric used, so it is usually referred as "edit distance". Edit distance have different definitions that use different sets of string operations. The Levenshtein uses operations such as removal, insertion and substitution of characters into strings. It is also widely used for a variety of analytical research such as spell checking, speech recognition and plagiarism detection. By applying this approach to this study, it matches terms with the least amount of edit distance; comparing between the dataset of noisy text and the IV Malay corpus.

The driving force towards incorporating Levenshtein distance as the formula to normalise words stem from Maarif et. al aimed to determine the complexity algorithm of each of the sub-algorithms that branched from the edit distance tree such as the Levenshtein Distance (LD), the

* Corresponding author: soon.layki@monash.edu

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

Jaro Winkler Distance (JWD), the Mahalanobis Distance, the Soundex Distance and the N-Gram Distance [3]. The importance of the study was to find out which edit distance was best suited for processing longer sentence comparison in correcting grammar in a Sign Language Synthesizer as proposed by the study. It was concluded that the Levenshtein Distance, Soundex Distance and N-gram Distance were the most efficient ones. Hence, Levenshtein Distance was chosen in this research.

1.2 Character Manipulation Rules

One of the recent works by Samsudin et al. investigated the pattern of abbreviations most used by Malaysians on social media and to create an artificial abbreviation list in order to improve the normalisation process of noisy texts [2]. They have generated a list of abbreviations following a set of rules that encompasses the way Malaysians write noisy terms. These rules are very thorough and comprehensive, and have been used in the study itself during the normalisation process. It is concluded that by generating the abbreviation list and implementing it along with a list of translated noisy terms, the level of accuracy of the procedure increases. Listed below are the set of rules followed by the team of researchers in making the Artificial Abbreviation list.

The first set is focused on rules that are related to manipulation of characters. This is the set of rules that will be adapted into this paper:

1. Remove all vowels (ie. "*mkn*", when it should be "*makan*").
2. Use only the first and last letter if either of them is not a vowel (ie. "*kg*", shortened version of "*kampong*").
3. Replace the last letter with '*e*' if it originally ends with '*a*' such as "*ape*" ("*apa*").
4. Add the letter '*k*' at the end of a word that ends with the letter 'a' such as "*juga*" ("*jugak*").
5. Remove the first vowel in a word if it starts with a consonant (ie. "*klau*", when it should be "*kalau*").
6. Remove the last vowel of a word if it ends with a consonant such as "*ank*" ("*anak*").
7. Use only the first and last letter such as "*pi*" ("*pergi*").
8. If the term ends with '*ar*', replace it with the character '*o*' such as "*sabo*" ("*sabar*").
9. If the term starts with '*ha*', drop the character '*h*' such as "*ari*" ("*hari*").
10. Use a character in replacement to a word with similar phonetic sounds such as "*x*" ("*tidak*"), "*n*" ("*dan*"), "*g*" ("*pergi*") and many more.

The second set of rules is on the manipulation of syllables of a word:

1. Use the first syllable (ie. "*sem*" which is supposed to be "*semester*").
2. Use the last syllable (ie. "*ngan*", which is "*dengan*"). If the last syllable of a word ends with '*a*', replace it with '*e*' (ie. "*je*" which is "*sahaja*") or add '*k*' (ie. "*gak*" which is an abbreviated version of "*juga*").
3. Use the first character of each syllable in a word such as "*spt*" ("*seperti*"). If the syllable starts with a group of consonant, the second character will be used such as "*tgk*" ("*tengok*").

The last set of rules is about the manipulation of the syllables and characters of a word. The list is as stated below:

1. Use the first character of each syllable and the last character (if the word is a consonant) such as "*byk*" ("*banyak*") and "*tgh*" ("*tengah*").
2. Use the first character and the last syllable such as "*bleh*" ("*boleh*") or "*bru*" ("*baru*"). If the new term ends with an '*a*', replace it with the character '*e*' such as "*bpe*" ("*berapa*") and "*mne*" ("*mana*").
3. Use the last syllable but replace the first character of the last syllable with the first character of the word such as "*tak*" ("*tidak*") and "*tgok*" ("*tengok*").

Based on the results obtained in their study, an average percentage of correctly defined noisy texts achieved 76%, while the average percentage of incorrectly identified noisy texts achieved 34%. Thus, it proves sufficient and relevant to reference the list of character manipulation rules into our research.

2 Related Work

Saloot et al. aimed to build a Twitter corpus comprising of Malay tweets [4]. They investigated the methods on creating a reliable Tweet corpus by following specialized corpus guidelines. This is then adapted in an advanced work [5], where they aimed to implement the Twitter corpus into the normalisation process by corpus analysis. We took into consideration the analysis carried out by them, where they extracted standard and colloquial characteristics and adopted some of their pre-processing activities.

A proposal brought forth by Muhamad et al. in 2017 deduced that for normalisation, the size of the dictionary database is very important [6]. The proposal paper is considered at a preliminary stage as it is without experimental results. They decided to implement a decision making mechanism based on Language Model, in which they explained, "A statistical language model [will] be able to represent the conditional probability on the next words based on sequence [of the] words." In order to achieve this, they developed the language model using N-grams and took into account of using feature selection.

Another related study by Basri et al. delved into the development of a spell checking system by proposing an approach to reduce the amount of misspelled words that are not in the lexicon [7]. Based on their proposed approach to this problem, their pre-processing modules also act as reference material with that of our own.

Among the many studies made in this area, Omar et al. proposed to create a Malay abbreviation corpus [8]. The study made sure to follow certain rules in order to construct the corpus, such as avoiding approaches that could minimize the efficiency of the corpus and chose to only follow best practices. This study incorporated the use of compiling IV words from Bahasa Wordnet and checked their authenticity with the main source of data, which is DBP.

3 Research Methodology

The theoretical framework of this research will commence operation once a test data—whether it be an extracted Tweet or Facebook post— from the gathered dataset is sent to the proposed framework. The noisy term will then undergo pre-processing phases in order to eliminate whitespaces and other unnecessary details that are not required for normalisation purposes. Pre-processing is comprised of case-folding, tokenization and punctuation removal and English word removal.

As shown in Fig. 1, the input text in the system will first go through case-folding, or the act of converting capital letters to lower case letters. Next, it will enter the tokenization phase, where it will first remove periods at the end of lines of the text and proceed to remove any and all special characters, including punctuation. The last step in pre-processing is eliminating English terms. The text inputted will be checked against an English dictionary; if the term is found, then we can remove it. Other than that, if it is not matched to any words in the existing dictionary, it will be considered OOV English words by checking in the prepared OOV dictionary, and will be removed all the same. Else, if there are still leftover English noisy terms, it will have to be eliminated manually.

Once all pre-processing phases have been completed, an analysis on the most frequently appeared noisy terms is carried out. The frequency of each noisy term is checked against both Malay OOV dictionary and English OOV dictionary to see if it exists. However, in this preliminary paper, the analysis is on the general overall amount of most frequent words that appear in the dataset. A Python library called Langid.py will be an assisting feature in determining the overall score of confidence that the dataset extracted is centred on Malay language rather than any other language [4, 5].

The normalisation process is conducted by using Python programming language and implementing the Levenshtein distance algorithm to match the words with similar traits between the document and the dictionary. A noisy term may be matched to more than one term from the dictionary. An example of such a situation is as follows:

The noisy term passed into the algorithm is "*kwn*". Based on this, the algorithm may find similarity of the term with "*kawan*" or "*kahwin*". Depending on which term has the highest percentage or possibility of similarity to the noisy term, the algorithm should automatically choose the proper term with the highest similarity trait, which is "*kawan*". Any discrepancies in word matching should be revised as a post-normalisation confidence checking step.

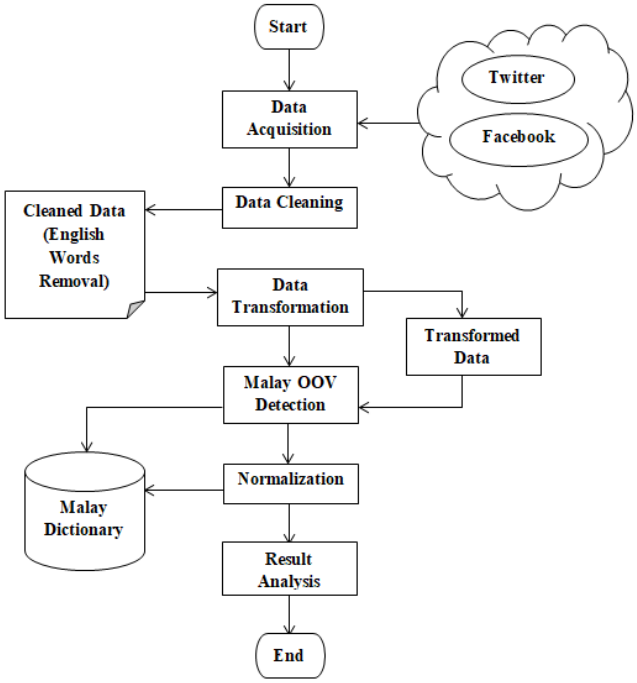

Fig. 1. Overview of the normalisation process.

3.1. Pre-processing

In order to carry out an efficient normalisation process, there are certain steps required to be adhered to, namely the pre-processing phase. During this particular phase, there are many different steps to take into consideration. However, in this research, we will focus on incorporating three pre-processing steps, as illustrated in Fig. 2:

1. Convert to lower case
2. Tokenization and punctuation removal
3. English word removal

3.1.1 Convert to Lower Case

This process involves the conversion of any and all capital letters to lowercase because uppercase do not have any orthographic value [5]. This particular method is also incorporated in web search engines, where when users type, for example, "*malaysia*", the search engine will automatically register that the user is interested in any sites related to the country, "*Malaysia*".

3.1.2 Tokenization

Tokenization is described as the act of chopping up a character sequence into pieces, namely tokens. A token is an instance of a sentence that is grouped together in a semantic way for the purpose of processing. Sometimes this process might remove certain characters, such as punctuation and so on [9]. The special symbol removal

process can also be incorporated along during tokenization (ie. !, ”, \, ^, %, $, #, @).

Although punctuation provides comprehensive grammatical context, it serves no function during noisy text normalisation and thus need to be cut out.

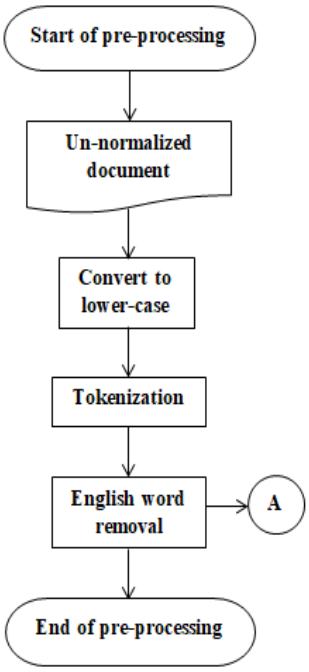

Fig. 2. Pre-processing flow.

3.1.3 English Word Removal

The significance of removing any and all English words from the dataset is to set our focus on normalizing Malay noisy terms only. By safely assuming that the words that contain in the document are solely in Malay, the normalisation process will be much smoother in a way that the processing does not have to consider identifying another language or mapping a term incorrectly. Figure 3 illustrates the overall flow in English words removal.

4 Implementation

Extraction of data from Twitter and Facebook were scraped by using the existing and public codes available. By signing up to be a Twitter and Facebook developer, the application program interface (API) credentials, which are crucial to be supplied into the codes, are obtained and thus the extraction process commences by inserting public and valid user IDs.

In the case of Twitter, in which we utilize the existing tweepy library, a separate text file is created and stores the Twitter users’ handle names. The scope of retrieving the information was through typing a few keywords into the search bar, which are “*makan*”, “*mkn*” and “*makanan*”. The user’s Tweets that appear at the top of the feed are taken for testing. In this case, the top five users were selected and their handle names were used to collect all of their Tweets. The reason the scope is centred on Tweets that contain the meanings of the words “eat” and “food” is mainly because Malaysia is known to have a diverse range of delicacies, and judging by the search results, a lot of citizens relay their expressions and thoughts on that particular subject. By receiving a large number of positive results that adhere to our query, it is easier to control the nature of the data extracted.

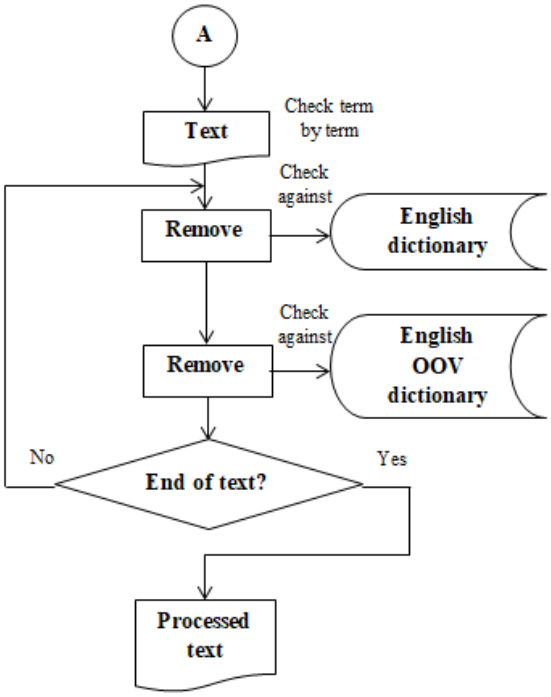

Fig. 3. Flow of the English removal pre-processing step.

Facebook, on the other hand, requires the name of the user and in order to extract the comments as well— this, as of now only applies to one post per user only— requires the user ID (userid) and the post ID (postid). The aforementioned information are joined by an underscore ‘_’ and fed into the code to extract the desired data (ie. userid_postid). For this site, we queried for “*makan*” and “*makanan*”. The same rules apply; the first five Facebook pages are selected to extract their comments on their five most recent posts. It is important to note that should the query return a page of a public figure or artist, we will ignore those pages. This is due to the fact that a public figure, such as Malaysia’s Prime Minister’s Facebook page, is not riddled with noisy text. A politician or any other public figure will write a post in formal Malay, something that does not contribute much to this research.

The total number of scraped Tweets surmounts to 15000 while the total amount of Facebook comments on several posts is 10000.

4.1 Normalisation

After the complete cleaning of the full dataset extracted, the remaining terms were checked against a downloaded Malay dictionary to acquire the most frequent terms. Due to an incomplete dictionary being used, there may be

some IV words that were included into the OOV most frequent words. For illustration purposes, only top 10 most frequent words from the dataset are shown in Table 1.

Table 1. Top 10 most frequent noisy terms.

Rank	Word	Frequency
1	mkn	1137
2	yg	917
3	je	614
4	nk	326
5	jom	301
6	korang	243
7	gak	200
8	dgn	197
9	mcm	179
10	mknn	174

The top 100 most frequent words were then extracted to create a cluster bar chart.

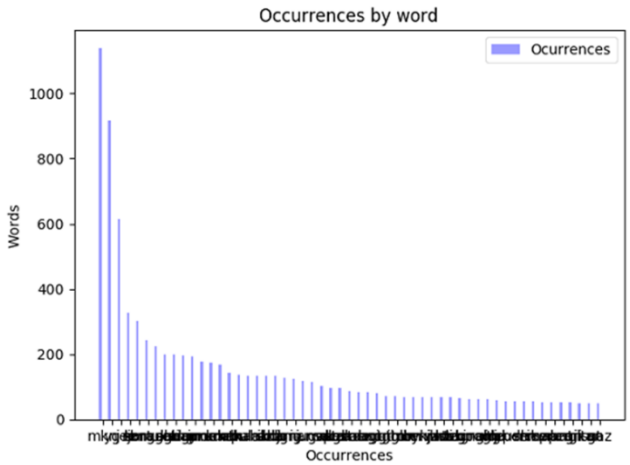

Fig. 4. Cluster bar chart detailing the occurrences versus the frequency of words.

By adopting P. Norvig's Levenshtein distance algorithm [10], it greatly influences the research. In the block of code, Norvig designs and formula that gets all possible variants for a word by implementing splits, deletion, transposition, replacement and insertion processes.

Other than referencing the spelling corrector's algorithm, this project also incorporated the Artificial Abbreviation list rules that was brought forward by Samsudin et al. [2]. However, not all of the rules were applied, seeing as there were rules that manipulated syllables. In order to apply the rules in that manner or manipulation, the model must be trained to recognize Malay syllables because English and Malay syllables are totally different from each other; hence the logic was not implemented. In addition, a much larger dataset is encouraged to be used in order to train a model. This can be a future work to extend this paper.

Besides the rules proposed by Samsudin et al. [2], a list of rules created during the course of this project were applied. The rules that were applied were technically a stemming approach, where the algorithm searches through each term for suffixes or affixes— in Malay, these are called '*imbuhan*'. Examples of '*imbuhan*' are '*men-*', '*ber-*', '*-kan*' and so on and so forth. By stripping these substrings and leaving the root of the word, the algorithm produces a more accurate result.

The list below shows the '*imbuhan*' substrings that were implemented into the algorithm:

1. If a word begins with either '*ber-*' or '*ter-*'
2. If a word begins with either '*di-*' or '*me-*'
3. If a word ends with either '*-kan*' or '*-nya*'

The rules by N. Samsudin et al. [2] that were applied into the algorithm are:

1. Remove all vowels (ie. "*mkn*", when it should be "*makan*").
2. Replace the last letter with '*e*' if it originally ends with '*a*' such as "*ape*" ("*apa*").
3. Add the letter '*k*' at the end of a word that ends with the letter '*a*' such as "*juga*" ("*jugak*").
4. If the term ends with '*ar*', replace it with the character '*o*' such as "*sabo*" ("*sabar*").
5. If the term starts with '*ha*', drop the character '*h*' such as "*ari*" ("*hari*").

5 Experimental Results

The results gained from the preliminary efforts in analysing, cleaning and doing small pre-processing tasks are presented by implementing language detection. In order to determine the overall confidence that the extracted dataset conforms to the criteria where it is centred on Malay language, this paper utilizes the langid.py tool [11]. This tool is a straightforward and a state-of-the-art language detection library that implements a normalisation on-log-probability that produces a confidence score, where 1 is the complete confidence and 0 is a failed identification.

The tool is tested on the text version of the dataset. As shown in Table 5, the Malay language is detected with a high confidence score rather than English. Hence, it can be validated that the dataset acquired is within the desired scope for this study.

Table 5. Language detection evaluation.

Language code (ISO 639-1)	Language name	Confidence score
ms	Malay	0.99
en	English	0.01

5.1 Evaluation of results

After successfully running the normalisation algorithm, a manual evaluation was carried out in order to determine whether the matched IV word is correct. The first execution was done without applying the aforementioned

rules. This gave a result of 8 predicted matched words while the remaining 92 is incorrectly predicted. The table below shows the pairings according to the comparison.

Table 6. Pairings of predicted word comparisons based on edit distance algorithm.

No.	OOV	IV
1	nk	nak
2	dh	dah
3	wkwk	wakwak
4	pegi	pergi
5	makana	makanan
6	menikmati	nikmati
7	nggak	enggak
8	dikasihi	dikasih

The second execution was made with the implementation of the rules. The results obtained were positive, with an increase in word prediction analysis and a better matching between stemmed terms. A successful 16 predicted matched words results were obtained, with a slight decrease in incorrectly matched words of 84 words.

Table 7. Pairings of predicted word comparisons based on edit distance and rules.

No.	OOV	IV
1	nk	nak
2	setiap	tiap
3	mknn	makan
4	dh	dah
5	menonton	tonton
6	wkwk	wakwak
7	pegi	pergi
8	membuat	buat
9	ternyata	nyata
10	sehari	hari
11	klang	kelang
12	menikmati	nikmati
13	nggak	enggak
14	selama	lama
15	dikasih	kasih
16	yaa	ya

Based on the analysis of the predicted words, it can be suggested that the Levenshtein edit distance is not the most suitable approach to the problem of normalizing noisy terms. It does, however, create a benchmark that can be further improved. This is due to the fact that the edit distance mentioned only returns the word with the least amount of edit, and that may or may not return the most accurate predicted word. On the other hand, by implementing some of the Artificial Abbreviation rules set by Samsudin et al.'s [2] and rules that were created during the course of this study, the results improved significantly.

6 Discussion

The implementation of the character manipulation rules and '*imbuhan*' rules performed better than only relying on the Levenshtein edit distance processes. This can be further illustrated with the word "*membuat*", as it is stemmed and returns "*buat*", which is the correct root word. Although it is true that "*membuat*" should exist as it is also a correct Malay word, it is important to note that the Malay corpus used have a limitation of words. The normalisation model would have been beneficial if it used a more complete and concise Malay corpus such as the Kamus Dewan & Pustake (DBP). However, this would require more time to acquire as the DBP is not available for electronic usage. Considering the vast repository of Malay words in the DBP, even if there were a downloadable version of the dictionary, the terms would have its definitions illustrated in colloquial words, too, and this would render the normalisation process inefficient. Furthermore, massive size of information in the corpus would only compromise the speed of the normalisation process.

A post-processing step is also crucial to ensure that the top 100 OOV words are indeed not included in the dictionary. The limitations faced are of course, an incomplete dictionary that does not include relevantly correct IV words, and that caused the generated OOV words to include IV words. Other than that, the applications of the rules missed some aspects and were not carried out during the normalisation process. An example of this is the noisy term "pulak", which is supposed to be "pula". Recall that one of the rules set by Samsudin et al. [2] was to remove words that end with '*k*' so that it gets to be matched correctly— this did not work during the process. Therefore, a more thorough post-processing approach should be done to avoid these limitations. The evaluation on the predicted words analysis was done manually.

7 Conclusion

The experimental data used in the experiments was extracted successfully followed by a preliminary test to ascertain that the coverage of the data was within the project scope, such as searching the keywords in order to scrape data from. The confidence score obtained by utilizing the language identification tool also aided the study in confirming that the progress is heading in the right direction. Although the cleaning of the dataset was imperfect, it managed to garner positive results based on the proposed algorithm that applied both the rules by Samsudin et al. [2] and our own rules.

The challenges that were met were mainly the imperfect dictionary used and the incomplete cleaning of the dataset. The edit distance used also did not produce the desired result, but acted as an important benchmark for this project. Nevertheless, this study can be further improved by researching and applying a more powerful and accurate edit distance, as well as implementing

machine learning to train the model in Malay linguistics. This can deal with the problem surrounding the noisy data that were influenced with syllable manipulation, and can even extend further towards recognizing and correcting slang words and even dialect-based words.

Acknowledgements

This work is partially funded by Fundamental Research Grant Scheme (FRGS) by Malaysia Ministry of Higher Education (Ref: FRGS/1/2017/ICT02/MMU/02/6 | 185265-207827).

References

1. M. A. Saloot, N. Idris, A. Aw, Noisy Text Normalization Using an Enhanced Language Model, *International Conference on Artificial Intelligence and Pattern Recognition*, 111-122 (2014)
2. N. Samsudin, M. Puteh, A. R. Hamdan, M. Z. A. Nazri, Normalization of noisy texts in Malaysian online reviews, *Journal of ICT* **12**, 147-159 (2013)
3. H. A. Maarif, R. Akmeliawati, Z. Z. Htike, T. S. Gunawan, Complexity Algorithm Analysis for Edit Distance, *Computer and Communication Engineering (ICCCE)*, 135-137 (2014)
4. M. A. Saloot, N. Idris, A. Aw, D. Thorleuchter, Twitter corpus creation: The case of a Malay Chat-style-text Corpus (MCC), *Digital Scholarship in the Humanities,* **31 (2)**, 227-243 (2014)
5. M. A. Saloot, N. Idris, and R. Mahmud, An architecture for Malay Tweet normalization, *Inf. Process. Manag.*, **50 (5)**, 621–633 (2014)
6. N. A. B. Muhamad, N. Idris, M. A. Saloot, Proposal: A Hybrid Dictionary Modelling Approach for Malay Tweet Normalization", *Journal of Physics: Conference Series* **806 (1)** (2017)
7. S. B. Basri, R. Alfred, and C. K. On, Automatic spell checker for Malay blog, *Control System, Computing and Engineering (ICCSCE)*, 506–510 (2012)
8. N. Omar, A. F. Hamsani, N. A. S. Abdullah, S. Z. Z. Abidin, Construction of Malay Abbreviation Corpus Based on Social Media Data, *Journal od Engineering and Applied Sciences* **12 (3)**, 468-474, (2017)
9. C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press. (2008)
10. P. Norvig, How to Write a Spelling Corrector (2007)