LexToPlus: A Thai Lexeme Tokenization and Normalization Tool

Choochart Haruechaiyasak and Alisa Kongthon
Speech and Audio Technology Laboratory (SPT)
National Electronics and Computer Technology Center (NECTEC)
Thailand Science Park, Klong Luang, Pathumthani 12120, Thailand
{choochart.har, alisa.kon}@nectec.or.th

Abstract

The increasing popularity of social media has a large impact on the evolution of language usage. The evolution includes the transformation of some existing terms to enhance the expression of the writer's emotion and feeling. Text processing tasks on social media texts have become much more challenging. In this paper, we propose *LexToPlus*, a Thai lexeme tokenizer with term normalization process. LexToPlus is designed to handle the intentional errors caused by the repeated characters at the end of words. LexToPlus is a dictionary-based parser which detects existing terms in a dictionary. Unknown tokens with repeated characters are merged and removed. We performed statistical analysis and evaluated the performance of the proposed approach by using a Twitter corpus. The experimental results show that the proposed algorithm yields an accuracy of 96.3% on a test data set. The errors are mostly caused by the out-of-vocabulary problem which can be solved by adding newly found terms into the dictionary.

1 Introduction

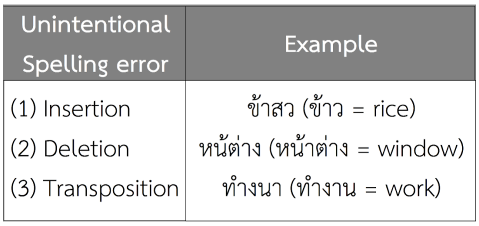
Thailand is among the top countries having a large population on social networking websites such as *Facebook* and *Twitter*. The recent statistics show that the number of social media users in Thailand has reached 18 millions (approximately 25% of the total population) as of the first quarter of 2013[1]. In addition to the enormous amount of texts being created daily, another challenging issue is the language usage on social media is much different from the traditional and formal language. Social media texts include chat message, SMS, comments and posts. These texts are usually short and noisy, i.e., contain some ill-formed, out-of-vocabulary, abbreviated, transliterated and homophonic transformed terms. These special characteristics are due to many reasons including inconvenience in typing on virtual keyboards of smartphones and intentional transformation of existing terms to better express the emotion and feeling of the writers. As a result, performing basic text processing tasks such as term tokenization has become much more challenging.

Tokenizing Thai written texts is more difficult than languages in which word boundary markers are placed between words. Thai language is considered as an unsegmented language in which words are written continuously without the use of word delimiters. Word segmentation is considered a basic yet very important NLP task in many unsegmented languages. The main goal of word segmentation task is to assign correct word boundaries on given text strings. Previous approaches applied to Thai word segmentation can be broadly classified as *dictionary-based* and *machine learning*. The dictionary-based approach relies on a set of terms from a dictionary for parsing and segmenting input texts into word tokens. During the parsing process, series of characters are looked up on the dictionary for matching terms. The performance of the dictionary-based approach depends on the quality and size of the word set in the dictionary. Recent works in Thai word segmentation have adopted machine learning algorithms. The machine learning approach relies on a model trained from a corpus by using sequential labeling algorithms. Using the annotated corpus in which word boundaries are explicitly marked with a special character, the algorithm could be applied to train a model based on the features (e.g., character types) surrounding these boundaries.

[1] Zocial Inc. blog, *http://blog.zocialinc.com/thailand-zocial-award-2013-summary/*

The 4th Workshop on South and Southeast Asian NLP (WSSANLP), International Joint Conference on Natural Language Processing, pages 9–16,
Nagoya, Japan, 14-18 October 2013.

The errors caused during the tokenization process can be categorized into two classes, *unintentional* and *intentional*. The unintentional errors are the typographical errors caused by careless typing (Peterson, 1980; Brill and Moore., 2000). This type of errors has been rigorously studied in the area of word editing and optical character recognition (OCR). There are three cases of typographical errors: *insertion*, *deletion* and *transposition*. Insertion error is caused by additional characters in a word. Deletion error is caused by missing characters in a word. Transposition error are caused by swapping of characters in the adjacent positions. Table 1 shows some examples of Thai word errors for all cases. The correct spellings are shown in parentheses with translations.



Unintentional Spelling error	Example
(1) Insertion	ข้าสว (ข้าว = rice)
(2) Deletion	หน้ต่าง (หน้าต่าง = window)
(3) Transposition	ทำงนา (ทำงาน = work)

Table 1: Unintentional spelling error types and examples

The scope of this paper does not include the unintentional errors which have been well studied. Instead we focus on intentional errors, i.e., words in which users intentionally create and type. Based on our preliminary study, the intentional errors can be classified into four categories: *insertion*, *transformation*, *transliteration* and *onomatopoeia*. The intentional insertion error is caused by typing repeated characters at the end of a word to emphasize the emotion or feeling. The transformation error is caused by alteration of existing terms and can be categorized into two subtypes: *homophonic* and *syllable trimming*. The homophonic terms refer to terms with the same or similar pronunciation to existing terms. The syllable trimming is a transformed term by deleting one or more syllables from an existing term for the purpose of reducing the keystrokes. The transliterated terms are created by using the Thai character set to create new terms from other languages. The last intentional error is the onomatopoeia terms which are created to phonetically imitate various sounds.

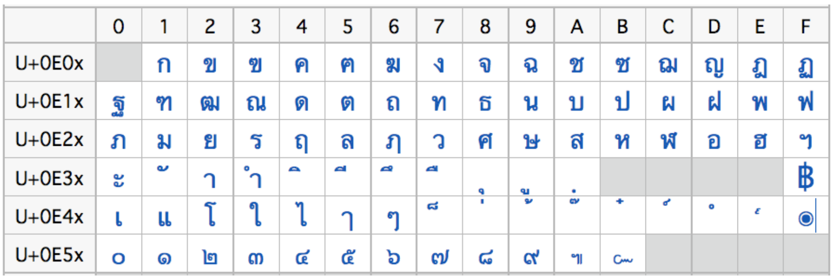
In this paper, we propose a solution for tokenizing and normalizing texts with the intentional insertion errors, i.e., users insert repeated characters at the end of words. The statistics on a 2-million Twitter corpus show that this type of errors accounts for approximately 4.8% of corpus size. Our proposed method is a longest matching dictionary-based approach with a rule-based normalization process. From our initial evaluation, the dictionary-based approach can handle the case of repeated characters better than the machine-learning based. More analysis and discussion will be given in the paper.

The remainder of this paper is organized as follows. In next section, we review some related works in word segmentation, text tokenization and term normalization for both segmented and unsegmented languages. In Section 3, we first give a formal definition of the tokenization task. Then we present the proposed algorithm for implementing LexToPlus. In Section 4, we give the performance evaluation by using a data set collected from Twitter. Some examples of errors are presented with some discussion. Section 5 concludes the paper with the future work.

2 Related work

Many techniques for word segmentation and morphological analysis have been reported for unsegmented languages. Peng et al. applied the linear-chain CRFs model for Chinese word segmentation (Peng et al., 2004). Their proposed model included a probabilistic new word detection method to further improve the performance. For Thai word segmentation, many previous works also applied machine learning algorithms to train the models. Meknavin et al. combined the model of word segmentation with the POS tagging (Meknavin et al., 1997). Their proposed model solved the ambiguity problem by using a feature-based model. Kruengkrai and Isahara applied the CRFs to train a word segmentation model for Thai language (Kruengkrai and Isahara, 2006). Two path selection schemes based on confidence estimation and Viterbi were proposed to solve the ambiguity problem. Haruechaiyasak et al. compared the performance among the dictionary-based approach and many machine learning techniques such as CRFs and the Support Vector Machines (SVMs) (Haruechaiyasak et al., 2008). The CRFs model was reported to outperform the dictionary-based approach and other machine learning algorithms.

While the majority of previous works focused on formal written texts, some works in text to-



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0E0x		ก	ข	ฃ	ค	ฅ	ฆ	ง	จ	ฉ	ช	ซ	ฌ	ญ	ฎ	ฏ
U+0E1x	ฐ	ฑ	ฒ	ณ	ด	ต	ถ	ท	ธ	น	บ	ป	ผ	ฝ	พ	ฟ
U+0E2x	ภ	ม	ย	ร	ฤ	ล	ฦ	ว	ศ	ษ	ส	ห	ฬ	อ	ฮ	ฯ
U+0E3x	ะ	ั	า	ำ	ิ	ี	ึ	ื	ุ	ู	ฺ					฿
U+0E4x	เ	แ	โ	ใ	ไ	ๅ	ๆ	็	่	้	๊	๋	์	ํ	๎	๏
U+0E5x	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙	๚	๛				

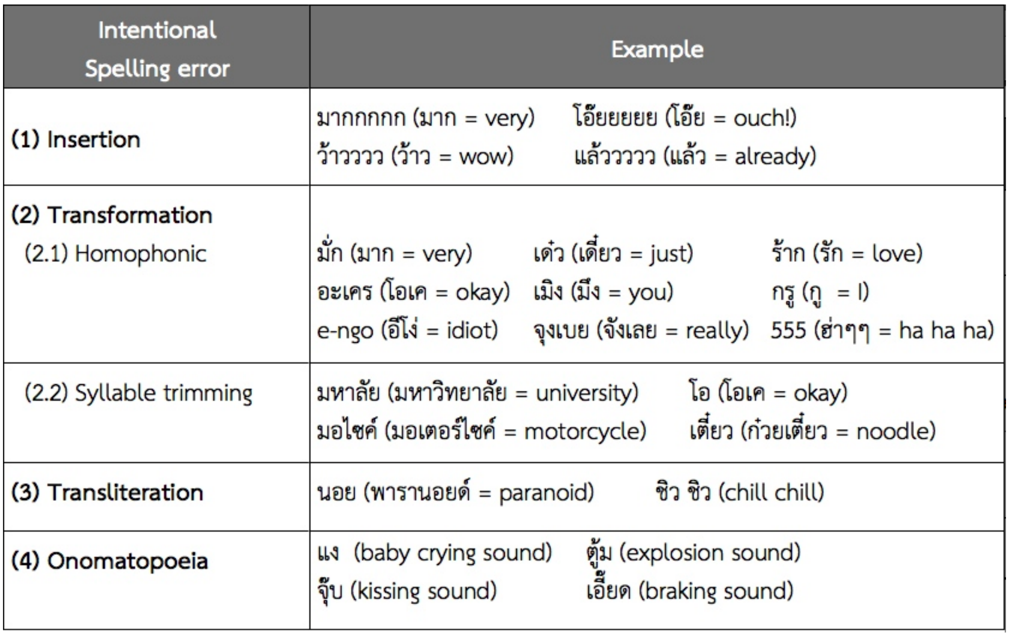
Figure 1: Thai Unicode Characters

kenization have expanded the scope into real-world texts which often contains many out-of-vocabulary terms. Tokenization for short and noisy texts requires an additional process of text normalization. Early works for text normalization were focused on news text, newsgroups posts and classified ads. Sprout et al. performed a study on non-standard words (NSWs), including numbers, abbreviations, dates, currency amounts and acronyms (Sprout et al., 2001). They applied several techniques including n-gram language models, decision trees and weighted finite-state transducers and showed that the machine learning approaches yielded better performance than the rule-based approach.

More recent works in text normalization focused on real world texts which contain informal language, such as SMS, chat and social media. Aw et al. proposed an approach for normalizing SMS texts for machine translation task (Aw et al., 2006). The normalization task is viewed as a translation problem from the SMS language to the English language. The results showed that normalizing SMS texts before performing translation significantly improved the performance based on the BLEU score. Costa-Jussa and Banchs proposed an approach for normalizing short texts, i.e., SMS (Costa-Jussa and Banchs, 2013). The proposed approach adopted the idea from statistical machine translation (SMT) by combining statistical and rule-based techniques.

Han et al. proposed a classification-based approach to detect ill-formed words, and generates correction candidates based on morphophonemic similarity (Han and Baldwin, 2011; Han et al., 2013). The proposed approach was evaluated on both SMS and Twitter corpus. The best performance was achieved with the combination of dictionary lookup, word similarity and context support modelling. Hirst and Budanitsky proposed a method for detecting and correcting spelling errors (Hirst and Budanitsky, 2005). The proposed method is based on the identification of tokens that are semantically unrelated to their context. The method also detects tokens which are spelling variations of words that would be related to the context. Liu et al. identified and distinguished nonstandard tokens found in social media texts as intentionally and unintentionally (Liu et al., 2012). A normalization system was proposed by integrating different techniques including the enhanced letter transformation, visual priming, and string/phonetic similarity. The proposed system was evaluated on SMS and Twitter data sets. The results showed that the proposed system achieved over 90% word-coverage across all data sets.

Another related research is the study of onomatopoeia in which terms are created to phonetically imitate different sounds. Research in onomatopoeia has recently gained much attention for Japanese language. Asaga et al. presented an online onomatopoeia example-based dictionary called ONOMATOPEDIA (Asaga et al., 2008). The proposed approach includes the extraction and clustering of sentences containing onomatopoeias as learning examples. Uchida et al. studied some Japanese onomatopoeias which contain various emotions (Uchida et al., 2001). Users are asked to rate emotion in each onomatopoeia. Correct identification of emotion from onomatopoeia could be used in advanced semantic analysis. Kato et al. proposed an approach to extract onomatopoeia found in food reviews (Kato et al., 2012). The extracted onomatopoeia terms can help users search for food or restaurants. In Thai language, many onomatopoeia terms are found in chat and social media texts. We classify onomatopoeia as another type of intentional errors while performing tokenization. Details are given and discussed in later section of the paper.



Intentional Spelling error	Example
(1) Insertion	มากกกกก (มาก = very) โอ๊ยยยยย (โอ๊ย = ouch!) ว้าวววว (ว้าว = wow) แล้ววววว (แล้ว = already)
(2) Transformation (2.1) Homophonic	มั่ก (มาก = very) เด๋ว (เดี๋ยว = just) ร๊าก (รัก = love) อะเคร (โอเค = okay) เมิง (มึง = you) กรู (กู = I) e-ngo (อีโง่ = idiot) จุงเบย (จังเลย = really) 555 (ฮ่าๆๆ = ha ha ha)
(2.2) Syllable trimming	มหาลัย (มหาวิทยาลัย = university) โอ (โอเค = okay) มอไซค์ (มอเตอร์ไซค์ = motorcycle) เตี๋ยว (ก๋วยเตี๋ยว = noodle)
(3) Transliteration	นอย (พารานอยด์ = paranoid) ชิว ชิว (chill chill)
(4) Onomatopoeia	แง (baby crying sound) ตู้ม (explosion sound) จุ๊บ (kissing sound) เอี๊ยด (braking sound)

Table 2: Intentional spelling error types and examples

3 Tokenization and normalization for Thai texts

Thai language has its own set of characters or alphabets. It has 44 consonants, 15 vowel symbols and four tone marks. Consonants are written horizontally from left to right, with vowels placed in four positions: above, below, left and right of the corresponding consonant. Tone marks can be placed in the position above of consonants and some vowels. There are 87 valid characters in Unicode system (shown in Figure 1).

3.1 Problem definition

The problem of tokenization for unsegmented languages can be defined as follows. Given a string of *N* words, $w_0 w_1 \ldots w_{N-1}$, each word w_i consists of a series of characters, $c^i_0 c^i_1 \ldots c^i_{|w_i|-1}$, where $|w_i|$ is the number of characters in w_i. The tokenization task is to assign a word boundary between two words, e.g., $w_i|w_j$, where $|$ represents a word boundary character.

From our preliminary study, the errors from tokenizing Thai texts from social media texts are due to four cases: insertion, transformation, transliteration and onomatopoeia. These error types are considered as *intentional errors* as opposed to the unintentional errors previously mentioned. Intentional errors are caused by users intentionally create, alter and transform existing words on different purposes. Table 2 lists all error types with some examples. The original terms or brief descriptions are shown in parentheses with translations. Each error type is fully explained as follows.

1. **Insertion**: This type of error is caused by repeated characters at the end of a word. Using the above problem definition, the error can be described as follows, $c^i_0 c^i_1 \ldots c^i_{|w_i|-1} c^i_{|w_i|-1} c^i_{|w_i|-1} c^i_{|w_i|-1}$, in which the last character $c^i_{|w_i|-1}$ is repeated more than once. This error type also appears in English, e.g., *whatttt*, *sleepyyy* and *loveeee*.

2. **Transformation**: This error type is caused by transformation of existing terms and can be categorized into two following types.

   Homophonic: The homophonic terms refer to terms with the same or similar pronunciation. A homophonic term is normally created by replacing an original vowel with a new vowel which has similar sound. Some examples in English are *luv (love)*, *kinda (kind of)* and *gal (girl)*.

   Syllable trimming: The syllable trimming is a transformed term by deleting one or more syllables from an existing term for the purpose of reducing the keystrokes.

3. **Transliteration**: Thai Transliterated terms are newly created terms converted from other

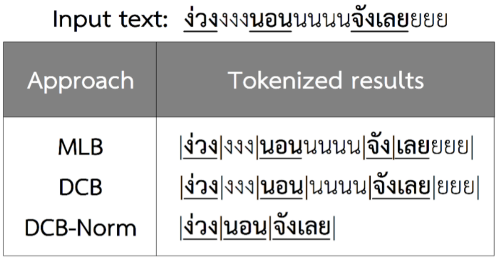
language scripts. Transliterated terms are commonly found in modern Thai written texts, e.g., chat and social media. Most of the terms are transliterated from English terms including named entities such as company and product names.

4. **Onomatopoeia**: Thai onomatopoeia terms are created by using the Thai character set to form new terms to imitate different sounds in nature and environment including humans and animals. Onomatopoeia terms are typically used in chat and social media texts to make the communications between users more vivid. For example, to make the kissing action sound more realistic, the word *joob* in Thai (or *smooch* in English) which imitates the kissing sound is normally used.

3.2 The proposed solution

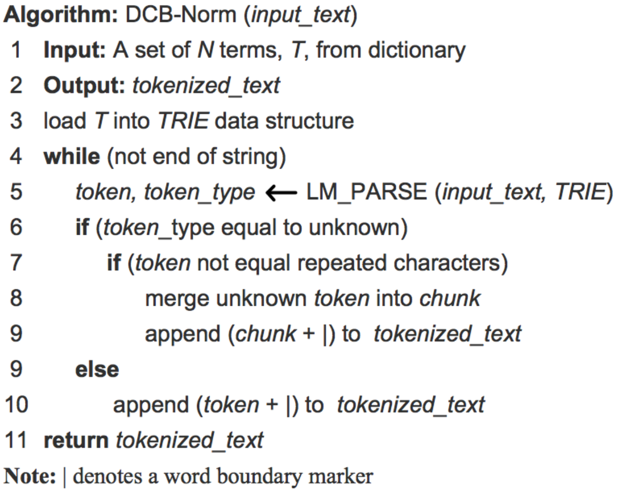
To select an appropriate approach for tokenizing and normalizing social media texts, we first perform a comparison between two approaches, dictionary-based (DCB) and machine learning based (MLB) (Haruechaiyasak et al., 2008). For machine learning based approach, we adopt the conditional random fields (CRFs) algorithm to train the tokenization model. The dictionary-based approach is a lexicon-based parser which solves the ambiguity with a longest matching heuristic. Table 3 shows tokenized results from different approaches. The input text consists of three words. Each word contains the insertion of some repeated characters at the end. The correct terms are bolded and underlined. The MLB approach cannot correctly assign the word boundaries between words. The first problem is due to the repeated characters at the end of each word. All repeated characters are recognized as part of the word. The second problem is due to the out-of-vocabulary which results in a word is incorrectly tokenized as two separating words.

The DCB approach can correctly tokenize all the terms which are included in the dictionary. The repeated word-ending characters are merged into a chunk. To further normalize the output words, we propose an algorithm DCB-Norm, which is a dictionary-based tokenization with a term normalization process. The DCB-Norm algorithm is shown in Figure 2. The algorithm performs text parsing with longest matching strategy (*LM_PARSE*). The strategy is used to solve the ambiguity problem in which there are more than one possible path to select in the parsing tree. The longest matching uses the heuristic such that longer terms contain better semantic than shorter terms. Figure 3 illustrates the dictionary-based tokenization with longest matching. From the example, there are four possible path to select. The longest matching strategy select the path (shown with a thick solid line) which contains a term with the largest length.


Input text: **ง่วง**งงง**นอน**นนนน**จังเลย**ยยย

<table>
<tr><th>Approach</th><th>Tokenized results</th></tr>
<tr><td>MLB</td><td>|<b>ง่วง</b>|งงง|<b>นอน</b>นนนน|<b>จัง</b>|<b>เลย</b>ยยย|</td></tr>
<tr><td>DCB</td><td>|<b>ง่วง</b>|งงง|<b>นอน</b>|นนนน|<b>จังเลย</b>|ยยย|</td></tr>
<tr><td>DCB-Norm</td><td>|<b>ง่วง</b>|<b>นอน</b>|<b>จังเลย</b>|</td></tr>
</table>

Table 3: An example of tokenized results



```
Algorithm: DCB-Norm (input_text)
Input: A set of N terms, T, from dictionary
Output: tokenized_text
load T into TRIE data structure
while (not end of string)
    token, token_type ⟵ LM_PARSE (input_text, TRIE)
    if (token_type equal to unknown)
        if (token not equal repeated characters)
            merge unknown token into chunk
            append (chunk + |) to  tokenized_text
    else
        append (token + |) to  tokenized_text
return tokenized_text
```

Note: | denotes a word boundary marker

Figure 2: DCB-Norm algorithm

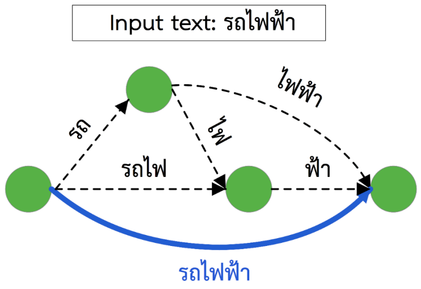


Figure 3: An example of tokenizing parse tree.

A set of terms in a lexicon is stored in a trie, an efficient data structure in terms of storage space and retrieval time (Frakes and Baeza-Yates, 1992). Figure 4 illustrates a trie storing an example set of terms. The algorithm begins by parsing the input text. A chunk of characters are looked up with terms stored in TRIE. If the character chunk are not found in TRIE and consists of same characters, it will be neglected. On the other hand, if a character chunk is found in TRIE, a word boundary marker (denoted with |) will be assigned at the end of the chunk.

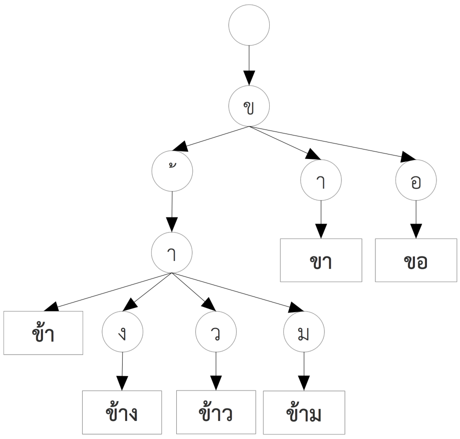


Figure 4: An example of trie

4 Experiments and discussion

In this section, we first perform a statistical analysis on a Twitter corpus to observe language usage characteristics. An experiment is then performed to evaluate the proposed tokenization and normalization solution.

4.1 Statistical analysis

To understand the characteristics of the language usage among Thai users on Twitter, we analyze a Twitter corpus consisting of *2,388,649* posts in Thai language. The total number of all words in the corpus is *25,683,296*. The number of unique tokens in the corpus is equal to *81,136*.

We run the *DCB-Norm* algorithm on the corpus and collect all repeated character chunks. Table 4 summarizes statistics of number of grams in repeated word-ending characters. Figure 5 shows the plot of the gram statistics. It can be observed that the occurrence of repeated characters is gradually decreased with the number of grams. From the corpus, we observe that some of the

# repeated grams	# posts
1	39,986
2	27,315
3	18,069
4	9,820
5	5,270
6 or more	14,560
Total	**115,020**

Table 4: Number of grams in repeated characters

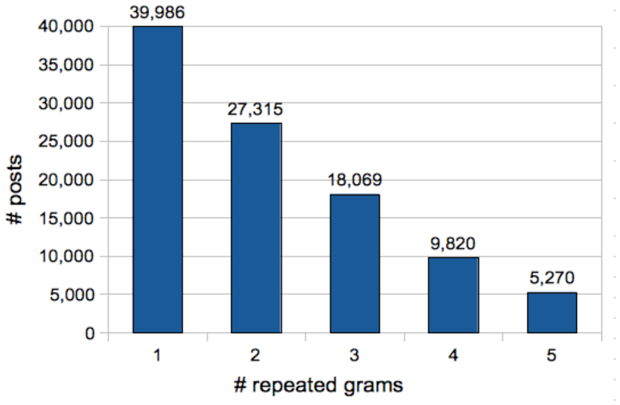


Figure 5: Number of grams in repeated characters

แล้ว (10291)	กรี๊ด (2774)	แว้ว (2061)
เนี่ย (1921)	โอ้ว (1810)	โว้ย (1387)
อืม (1142)	สาด (1124)	เว้ย (935)
เล้ย (896)	แร้ว (875)	ฮิ้ว (790)
แย้ว (789)	ยอด (787)	เลย (786)
โอย (751)	ข้าว (734)	เรย (720)
โอ๊ย (706)	หิว (682)	โอ้ย (666)
ด้วย (615)	ว้อย (613)	ว้าว (602)
เฮ้ย (574)	นอน (563)	แสรด (533)

Table 5: Top ranked terms with repeated characters

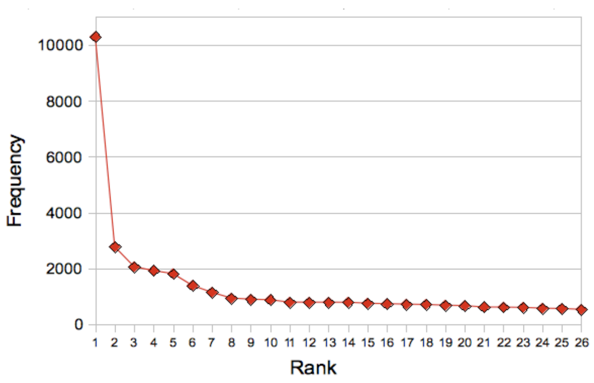


Figure 6: Top ranked terms with repeated characters

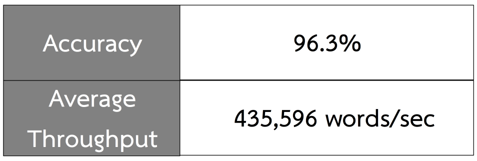
posts containing more than 10 repeated character grams. Another observation is the repeated characters mostly occur at the end of the words as we expected.

Next, we analyze the high frequent terms which contains the repeated characters. Table 5 shows top ranked terms with repeated characters. The total number of posts which contain some repeated characters is 115,020 which is equal to 4.8% of corpus size. Figure 6 shows the plot of top ranked terms. From the figure, we observe the Zipf (or long-tail) distribution over the terms with repeated characters. Most of the top terms are very informal and often used in conversational language. Some of the terms are used to express the feeling and emotion of the users.

4.2 Experiments and discussion

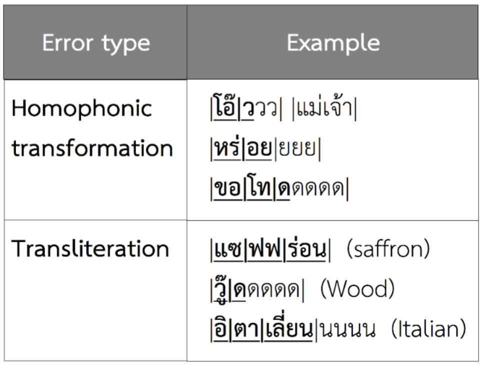
To evaluate the performance of the proposed approach, we perform an experiment by using a set of 1,000 randomly selected Twitter posts written in Thai language. Each post contains some repeated characters and is manually assigned with correct word boundary markers. For the proposed DCB-Norm algorithm, we use a lexicon from LEX*i*TRON[2] which contains *35,328* general terms. We also include another lexicon consisting of *1,341* terms frequently found in Twitter corpus. Words obtained from Twitter lexicon include chat, slangs and transliterated words from other languages.

The performance evaluation is carried out on a notebook with a 2 GHz Intel Core 2 Duo CPU, 4 GB RAM running under Mac OS X. We evaluate the algorithm in terms of accuracy, i.e., the number of correctly tokenized texts over the total number of test texts. We also evaluate the running time efficiency. The results are summarized in Table 6. The overall accuracy is equal to *96.3%*. To analyze the errors, we manually look at the incorrectly tokenized results. We observe that in the case of all words in the text are in the dictionary, the words are recognized and the repeated characters are correctly removed. However, the problem is mostly due to out-of-vocabulary (OOV) which causes the incorrect assignment of word boundary markers. As a result, the words with repeated characters at the end could be not normalized correctly. Two error types associated with OOV problem is homophonic transformation and transliteration. Table 7 shows some examples from the error analysis. The simplest solution to the OOV problem is to manually collect newly created terms from the corpus.



Accuracy	96.3%
Average Throughput	435,596 words/sec

Table 6: Evaluation results



<table>
<tr><th>Error type</th><th>Example</th></tr>
<tr><td>Homophonic transformation</td><td>|โอ๊|ววว| |แม่เจ้า|
|หร่|อย|ยยย|
|ขอ|โท|ดดดดด|</td></tr>
<tr><td>Transliteration</td><td>|แซ|ฟฟ|ร่อน| (saffron)
|วู๊|ดดดดด| (Wood)
|อิ|ตา|เลี่ยน|นนนน (Italian)</td></tr>
</table>

Table 7: Examples of tokenized errors

5 Conclusion and future work

We proposed a tool called *LexToPlus* for tokenizing and normalizing Thai written texts. LexToPlus is designed to handle the intentional word errors commonly found in social media texts. In this paper, we focus on solving the tokenization errors caused by repeated characters at the end of words. The proposed algorithm DCB-Norm is a dictionary-based parser with a rule-based extension to merge and remove repeated characters. We

[2] LEX*i*TRON, *http://lexitron.nectec.or.th*

performed some statistical analysis on a Twitter corpus consisting of over 2 million posts written in Thai language. One interesting result is the ranking distribution of top terms with repeated characters follows the Zipf or long-tail distribution. We evaluated the proposed algorithm by using a corpus of 1,000 manually tokenized texts. The accuracy is equal to 96.3% with the average throughput of 435,596 words/second.

From the error analysis, we found that the major problem is the out-of-vocabulary (OOV) which comes from homophonic transformation and transliteration. Although the OOV problem can be partially solved by adding new terms into the dictionary. However, it is labor intensive and ineffective in long terms. For future work, we plan to improve the performance of the proposed algorithm by constructing a machine learning model to automatically detect new terms based on the contextual information.

References

Chisato Asaga, Yusuf Mukarramah and Chiemi Watanabe. 2008. ONOMATOPEDIA: onomatopoeia online example dictionary system extracted from data on the web. *Proc. of the 10th Asia-Pacific web conf. on Progress in WWW research and development* , 601-612.

AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for SMS text normalization. *Proc. of the COLING/ACL on Main conference poster sessions*, 33–40.

Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. *Proc. of the 38th Annual Meeting on Association for Computational Linguistics*, 286–293.

Marta R. Costa-Jussa and Rafael E. Banchs. 2013. Automatic normalization of short texts by combining statistical and rule-based techniques. *Language Resources and Evaluation* , 47(1):179–193.

William B. Frakes and Ricardo Baeza-Yates (Eds.). 1992. *Information Retrieval: Data Structures and Algorithms* , Prentice-Hall, Englewood Cliffs, NJ.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: makn sens a #twitter. *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics*, 368–378.

Bo Han, Paul Cook, and Timothy Baldwin. 2013. Lexical normalization for social media text. *ACM Transactions on Intelligent Systems and Technology*, 4(1): 1–27.

Choochart Haruechaiyasak, Sarawoot Kongyoung and Matthew Dailey. 2008. A comparative study on Thai word segmentation approaches. *Proc. of the ECTI-CON 2008*, 1:125-128.

Graeme Hirst and Alexander Budanitsky. 2005. Correcting real-word spelling errors by restoring lexical cohesion. *Natural Language Engineering*, 11:87–111.

Ayumi Kato, Yusuke Fukazawa, Tomomasa Sato and Taketoshi Mori. 2012. Extraction of onomatopoeia used for foods from food reviews and its application to restaurant search. *Proc. of the 21st int. conf. companion on World Wide Web* , 719-728.

Canasai Kruengkrai and Hitoshi Isahara. 2006. A conditional random field framework for thai morphological analysis. *Proc. of the Fifth Int. Conf. on Language Resources and Evaluation (LREC-2006).*

Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. *Proc. of the 50th Annual Meeting of the Association for Computational Linguistics*, 1035-1044.

Surapant Meknavin, Paisarn Charoenpornsawat, and Boonserm Kijsirikul. 1997. Feature-Based Thai Word Segmentation. *Proc. of the Natural Language Processing Pacific Rim Symposium*, 289–296.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese Segmentation and New Word Detection Using Conditional Random Fields. *Proc. of the 20th COLING*, 562–568.

James L. Peterson. 1980. Computer programs for detecting and correcting spelling errors. *Communications of ACM*, 23:676–687.

Richard Sproat, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer Speech and Language*, 15(3):287 333.

Yuzu Uchida, Kenji Araki, and Jun Yoneyama. 2012. Classification of Emotional Onomatopoeias Based on Questionnaire Surveys. *Proc. of the 2012 International Conference on Asian Language Processing*, 1–4.