A Method for Normalizing Non-standard Words in Online Social Network Services: A Case Study on Twitter

Dongjin Choi, Jeongin Kim, and Pankoo Kim(✉)

Department of Computer Engineering, Chosun University, 375 Seoseok-dong, Dong-gu, Gwangju, Republic of Korea
{dongjin.choi84,jungingim}@gmail.com, pkkim@chosun.ac.kr

Abstract. Due to the big developments of Smartphone devices and online social network services, people can share diverse information about what they have been experienced during a day with no constrain to time or location. This fact has changed entire previous online system. We simply insert a query to search engine or OSNSs by using Smartphone devices. Because of this effectiveness, text data in OSNSs is getting bigger including many noisy data especially non-standard words. People are likely to type a text in short format such as abbreviation, acronym, and more when they using Smartphone to send a message to their friends in order to save time and data usages. As a result of these reasons, non-standard words on the web is extremely increasing so it has to be normalize into standard words in order to enhance performance of Natural Language Processing. When we analyze plain text data to extract semantic meaning, this nosy data has been ignore even though it has valuable information. In order to overcome this problem, we address a method for normalizing non-standard words in OSNSs, particularly for Twitter text data. We analyzed more than fifty million tweets which was collected by Stanford University and normalized non-standard words into standard English words by using diverse coefficient method such as dice, jacard, ochiai, sorgenfrei, and more. We finally conclude this paper by comparing those coefficient methods with our proposed one.

Keywords: Words normalization · Online social network services · Twitter

1 Introduction

There are many kinds of contents described in diverse forms such as text, audio, images, and more on the World Wide Web. Particularly, text data is the most common type of data and the volume of this data is extremely huge. Due to this text data was written by human, many researchers have been studying to discover semantic meanings from human written text data by using diverse kinds of approaches such as statistical, machine learning, knowledge

P.C. Vinh et al. (Eds.): ICCASA 2013, LNICST 128, pp. 359–368, 2014.
DOI: 10.1007/978-3-319-05939-6_35,

based approaches, and so on [1–3]. Although, scientists have been applied highly advanced approaches for a long time, it is still challenging that the performance of Natural Language Processing (NLP) is not always increasing due to the fact that human written data has many noisy texts. For instance, miss spelled word 'univercity,' abbreviation 'st.' acronym 'WSD' are normally considered as a unimportant data for extracting semantics even though those words has valuable information. Moreover, special characters such as #, @, &, and more are normally removed in a preprocessing step in order to reduce the size of test data and enhance the performance ratio. However, these special characters do indicate specific actions in Twitter system. There is a policy in Twitter that people can share information by sending text-based message restricted to only 140 characters, known as tweets. As a result of this policy, non-standard words on the web is extremely increasing so it has to be normalize into standard words in order to machine can analyze non-standard words. Besides, there is a high possibility that people are likely to type a text in short format such as abbreviation, acronym, or no-standard English form when they using Smartphone to send a message to their friends in order to save time and data usages. This issue brings huge obstacle when scientists conduct an experiment by using NLP techniques. In order to overcome this problem, we introduce a method to normalize non-standard words in online social network services particularly for Twitter data by using several coefficient approaches. The reminder of the paper is organized as follows: Sect. 2 describes what Twitter is and related works; Sect. 3 explains a method for normalizing non-standard words in Twitter based on several coefficient approaches; Sect. 4 gives experimental results; and finally Sect. 5 conclude this paper with future works.

2 Related Works

Twitter is one of the most common online social network services (OSNSs) platform which let people share information, interests, and knowledge among others by sending text-based message restricted to only 140 characters, known as a tweet [4]. Because of this limitation of text size and inconvenience of Smartphone text input system, people type a text message as short as possible in order to save time and usage. As a result, Twitter is full of non-standard words which machine cannot detect as important terms. Besides, there are few interesting functions which operated by special characters in Twitter. The following Table 1 shows examples of tweets written by certain online user A.

According to Twitter policy, there are interesting functions which is described by special characters such as @, #, and RT. The @ sign is used to call out (mention) usernames in Tweets and it becomes a link to a user profile. The # indicates hashtags which is used to mark keywords or topics to categorize messages. RT represents retweet which is a re-posting someone else tweets to spread news or share valuable information with others [5]. Therefore, these non-standard words have not to be removed in pre-processing step. Non-standard words have been interested by many researchers for long time. Before we introduce other

Table 1. Examples of tweet messages in Twitter

Text messages in Twitter
RT @fanfusionblog: RT this message to win $ 50 Gift Card! 100th to do so wins!
@AllSelenaGomez Blah, I'm tired, good. You?
#dumbquestions (calling the house phone) WHERE ARE YOU?!
@MrsBieber69 love you too, and #ofcourse #betheresoon

Table 2. Examples of non-standard words

Taxonomy	Example
Contraction	Im, cant, wont, havent, ...
Abbreviation	uni., dept., ref., max., ...
Acronym	SCUBA, ROM, NLP, NER, FBI, ...
Mixed	WS99, x220, MS-Dos, ...
Funny spelling	cooool, hoooooot, lol, :), b4, ...
Misspelling	geogaphy, univercity, knowlege, ...

researches approaches to normalize non-standard words, we describe examples of non-standard words shown in Table 2.

A contraction word is a shortened form of word created by using apostrophe such as {*let us: lets, I am: Im, cannot: cant, have not: haven*}. This contraction is defined in English grammar so it is not difficult to normalize into standard words. An abbreviation is a shortened form of word or phrase to omit when the length of a word is too long such as ad for advertisement. Abbreviation is also defined in many English dictionaries so it can be normalized by using dictionary and rule based approaches [6,7]. An acronym is an abbreviation formed which is composed of initial components of multiple words such as {*CEO: Chief Executive Officer, SCUBA: Self-Contained Underwater Breathing Apparatus, FAQ: Frequently Asked Question, BBC: British Broadcasting Corporation*}. The problem is that acronyms can have several kinds of different expansion words, in other words it has a word sense disambiguation problem. For example, acronym NER indicates not only Named Entity Recognition in computer science area but also North Eastern Railway which was an English railway company. According to Wikipedia[1], there are 14 different kinds of entities indicating acronym NER. This is huge obstacle when we faced this polysemous acronym in test data. In order to overcome this obstacle, Hwang [8] proposed a method for normalize terminologies by using Wikipedia labels to enhance the performance. This research based on the assumption that labels in Wikipedia is the standard terms for terminologies. However, they did not cover the acronyms which have multiple expansion words. Non-standard words such as a mixed form would not be fully detected when we only applied rule-based approaches. Because, there are no particular patterns to represent mixed formed words. Funny spelling words are new type of words to

[1] http://www.en.wikipedia.org/wiki/NER

emphasize adjective by repeating certain characters or emoticons. These words are created due to the fact that people want to express their emotional status far beyond normal English words. Misspelled words are commonly appeared in online text messages, email, short message service (SMS), blog, Twitter, Facebook, and more. Misspelled words can be normalized by using n-gram based approaches [9,10]. As we can see in this section, English words are not always expressed by perfect standard format. Especially, online text messages are full of non-standard words. Hence, we propose a method for normalizing non-standard words appeared in OSNSs especially for Twitter text messages.

3 Twitter Data Set

Twitter text messages which we are going to deal with were collected by Stanford University [11] contains entire text messages on November in 2009 approximately 8.27 GB by more than 5.5 million users. In order to find active users not spammers, we focused on users who exposed someones birthday by using simple linguistic rule [12]. So we can obtain 24,922 candidate active users. This data set consists of three kinds of information which are time, user URL, and tweet message shown in Table 3.

Table 3. Examples of the Twitter data set

Type	Information
T	2009-11-01 00:43:19
U	http://twitter.com/ivoryshorty
W	@iamdorkster you the effin' best ...
T	2009-11-01 01:10:46
U	http://twitter.com/ivoryshorty
W	@documentedmusic very dope...I like your flow ...
T	2009-11-01 02:49:53
U	http://twitter.com/ivoryshorty
W	@PinkRoyaltie Aww how cute! Yeah ours was yesterday ...

As we can see in Table 3, the tweet messages are full of non-standard words which are needed to be normalized. We randomly choose five thousands users from Twitter data set and analyzed how many non-standard words are by using PyEnchant[2] module which is a spellchecking library for Python. The total number of tweets for 5,000 users is 365,967 which contain 5,748,586 words. According to PyEnchant spellchecking library, there are more than 40 percent of non-standard words in this tweet text messages. This will bring big obstacle if we analyze natural language processing by using this Twitter text messages. This is the main reason why we want to normalize non-standard words in Twitter.

[2] http://www.pythonhosted.org/pyenchant/

4 Normalization for Non-standard Words in Twitter

This section describes a method for normalizing non-standard words into standard words in Twitter text messages by using several coefficient methods. As we introduced in the previous section, we prepared test data set which contains non-standard words. These non-standard words are consisted of contractions, abbreviations, acronyms, mixed words, funny spelling words, and misspelling words. We put those words into PyEnchant module to find standard words of them. The following Table 4 shows the results of this step.

Table 4. Results of suggestions for non-standard words by using PyEnchant

Taxonomy	NSW	Suggestions
Contraction	havent	haven, haven't, havens, ha vent, haven t'
Contraction	youre	yourself, your, you're, you've
Mixed	MS-Dos	None
Funny spelling	freee	free, freeze, frees, freer, freed, free e
Funny spelling	need2know	needlework, needlewomen, needlewoman,needlepoint
Misspelling	definately	definitely, definably, determinately, definable, definitively

Although, PyEnchant is powerful module to check English spelling, it does not always give a perfect suggestion for non-standard words. PyEnchant module cannot find expansion words for abbreviations, acronyms, and mixed words. However, it does recommend candidate standard words for contractions, funny spelling words, and misspelling words. The problem is that we cannot guarantee whether suggested words for funny spelling words are correct or not. Therefore, we want to find the most appropriate words by using Dice, Jaccard, Ochiai, and proposed coefficient approaches. The non-standard words we want to focus on are contraction, funny spelling, and misspelling words in this paper.

Dice coefficient is a statistic approach for comparing the similarity of two samples developed by *Lee Raymond Dice* [13]. Let us assume that we have two samples '*havent*' and '*haven't*'. Bigrams of these two words can be represented by as follows: $bigram_{havent} = \{ha, av, ve, en, nt\}$ and $bigram_{haven't} = \{ha, av, ve, en, n', 't\}$. Dice coefficient of these two words can be calculated by following the Eq. 1.

$$Dice\_coeff(w_i, w_j) = \frac{2 \times |bigram_{w_i} \cap bigram_{w_j}|}{|bigram_{w_i}| + |bigram_{w_j}|} \tag{1}$$

where, $|bigram_{w_i}|$ and $|bigram_{w_j}|$ are the number of total bigram of given words $|w_i|$ and $|w_j|$. $|bigram_{w_i} \cap bigram_{w_j}|$ denotes the number of bigrams which appeared in $|w_i|$ and $|w_j|$ at the same time.

Jaccard coefficient which was developed by *Paul Jaccard* is statistic used for measuring similarity and diversity of samples followed by the Eq. 2 [13].

$$Jaccard\_coeff(w_i, w_j) = \frac{|bigram_{w_i} \cap bigram_{w_j}|}{|bigram_{w_i} \cup bigram_{w_j}|} \tag{2}$$

Ochiai coefficient or also known as Ochiai-Barkman coefficient, or Otsuka-Ochiai coefficient was considered as a superior coefficient measurement in research [13] which can be calculated by the following Eq. 3.

$$Ochiai\_coeff(w_i, w_j) = \frac{|bigram_{w_i} \cap bigram_{w_j}|}{\sqrt{|bigram_{w_i}| \times |bigram_{w_j}|}} \tag{3}$$

Our proposed coefficient measurement is performed well when two given words has the same number of characters such as words {*'Seoul' and 'Seuol'*} followed by the Eq. 4.

$$Proposed\_coeff(w_i, w_j) = \frac{|bigram_{w_i} \cap bigram_{w_j}|}{avg(|bigram_{w_i}| + |bigram_{w_j}|)} \tag{4}$$

We hereby measure the similarities between non-standard word and suggested words by PyEnchant module in order to address what the weakness of this

Table 5. Similarity results between non-standard words and suggested words

NSW	Suggestions	Dice	Jaccard	Ochiai	Proposed
freeee	freezer	0.667	0.375	0.548	0.545
	freemen	0.667	0.375	0.548	0.545
	freeze	**0.75**	**0.428**	**0.6**	**0.6**
	freeness	0.6	0.333	0.507	0.5
	free	1.0	0.6	0.774	0.75
definately	**definitely**	**0.778**	**0.636**	**0.778**	0.889
	definably	0.706	0.545	0.707	**0.941**
	definable	0.588	0.417	0.589	0.823
	subordinately	0.571	0.4	0.577	0.762
Thinking	**Thinking**	**0.909**	0.625	0.771	0.769
	Thinkable	0.615	0.4	0.577	0.714
	Thinkably	0.615	0.4	0.577	0.714
	Thinker	0.727	0.5	0.667	**0.833**
	Think	0.889	**0.667**	**0.816**	0.8
havent	**haven't**	0.727	0.571	0.730	**0.909**
	Haven	**0.889**	**0.8**	**0.894**	0.889
	haver	0.667	0.5	0.670	0.889
bday	Hobday	**0.75**	**0.6**	**0.774**	0.75
	Bayda	0.571	0.4	0.577	**0.857**
	birthday	0.4	0.25	0.436	0.6
	daybed	0.5	0.33	0.516	0.75
thnks	methinks	0.545	0.375	0.567	0.727
	think	0.5	0.333	0.5	0.75
	thank	0.5	0.333	0.5	0.75
	thunk	0.5	0.333	0.5	0.75
txt	**text**	0.4	0.25	0.408	**0.8**
	twixt	0.333	0.2	0.353	0.667
	TX	**0.667**	**0.5**	**0.707**	0.667

system and how can we applied our proposed coefficient measurement. Table 5 indicates the results of similarities between non-standard words and suggested words by using four kinds of coefficient methods.

According to the results of Table 5, the proposed method does not always give the best standard words for NSWs. However, it does give the best words if the type of non-standard words is the contraction such as *'havent'*. As we can see in the Table 5, the candidate words for *'havent'* are {*havent, Haven, haven, haver*}. If we applied the Dice, Jaccard, and Ochiai coefficient methods, we cannot find the right words for *'havent'*. However, we could find the most likely reasonable word if we applied the proposed coefficient method.

The non-standard word *'freeee'* is the funny spelling words for a word *'free'*. However, as we can see in the Table 5, PyEnchant suggested four kinds of words {*'freezer,' 'freemen,' 'freeze,' 'freeness'*} except *'free'*. If PyEnchant can recommend a word *'free'* the similarities between words *'freeee'* and *'free'* might be the highest values by using four kinds of coefficients. Therefore, we have concluded that this module is not suitable for normalizing funny spelling words. In

Table 6. Similarity results between funny words and suggested words

Funny words	Suggestions	Dice	Jaccard	Ochiai	Proposed
happppppy	happily	0.6	0.3	0.462	0.615
	happening	0.5	0.25	0.400	0.4
	happiness	0.5	0.25	0.400	0.4
	happy	**1.0**	**0.571**	**0.755**	**0.727**
freeee	freezer	0.667	0.375	0.548	0.545
	freemen	0.667	0.375	0.548	0.545
	freeze	**0.75**	**0.428**	**0.6**	**0.6**
	freeness	0.6	0.333	0.507	0.5
juuuust	justness	0.545	0.3	0.462	0.461
	justest	**0.666**	**0.333**	**0.5**	**0.5**
	justing	0.6	**0.333**	**0.5**	**0.5**
	Justice	0.6	**0.333**	**0.5**	**0.5**
truuueee	trusteeing	0.428	0.230	0.377	0.5
	trusteeship	0.4	0.214	0.358	0.470
	Truckee	**0.545**	**0.3**	**0.462**	**0.615**
	trustee	**0.545**	**0.3**	**0.462**	**0.615**
goooooole	**Google**	0.666	0.333	0.507	0.5
baaaad	Baal	0.666	0.333	0.516	0.5
weeeeird	weekender	0.307	0.153	0.267	0.266
	weirdness	0.615	0.363	0.534	0.533
	weedkiller	0.285	0.142	0.251	0.375
	weirdie	**0.727**	**0.444**	**0.617**	**0.615**
hellllooo	hellebore	0.461	0.25	0.400	0.533
	Hellespont	0.428	0.230	0.377	0.5
	hellhole	0.5	0.272	0.428	0.571
	hellion	**0.545**	**0.3**	**0.462**	**0.615**

Table 7. Similarity results between normalized funny words and suggested words

Normalized words	Suggestions	Dice	Jaccard	Ochiai	Proposed
happy	**happy**	**1.0**	**1.0**	**1.0**	**1.0**
free	**free**	**1.0**	**1.0**	**1.0**	**1.0**
	frees	0.857	0.75	0.866	0.857
	freer	0.857	0.75	0.866	0.857
	freed	0.857	0.75	0.866	0.857
juust	**just**	**0.857**	**0.75**	**0.866**	**0.857**
	Justen	0.666	0.5	0.670	0.666
	juster	0.666	0.5	0.670	0.666
	justed	0.666	0.5	0.670	0.666
	Justis	0.666	0.5	0.670	0.666
truuee	trustee	0.5454	0.375	0.547	0.727
	Truckee	0.5454	0.375	0.547	0.727
	truelove	0.5	0.333	0.507	0.666
	trueness	0.5	0.333	0.507	0.666
	true	**0.75**	**0.6**	**0.774**	**0.75**
goole	**Google**	**0.666**	**0.5**	**0.607**	**0.666**
	goober	0.444	0.285	0.447	0.444
	gooier	0.444	0.285	0.447	0.444
	Goober	0.444	0.285	0.447	0.444
baad	Baal	0.666	0.5	0.666	0.666
	baa	**0.8**	**0.666**	**0.816**	**0.8**
	ballad	0.5	0.333	0.516	0.5
	Baden	0.571	0.4	0.577	0.571
	bad	**0.8**	**0.666**	**0.816**	**0.8**
weeird	**weird**	**0.888**	**0.8**	**0.894**	**0.888**
	weirdie	0.727	0.571	0.730	0.727
	weirdo	0.8	0.666	0.8	0.8
	weeing	0.6	0.428	0.6	0.6
	weirdness	0.615	0.444	0.632	0.615
helloo	**hello**	**1.0**	**1.0**	**1.0**	**0.75**
	weirdness	0.888	0.8	0.894	0.666

order to find the most appropriate word for *'freeee'*, we defined a rule based on the linguistic patterns of the funny spelling words. In case of the funny spelling words, certain character is normally repeated for emphasizing a word. For example, {*happppppy, freeee, juuuust, truuueee, gooooole, baaaad, weeeeird, hellllooo*} these words are the funny spelling words when people used for giving emotional status in text messages. A funny spelling word can be represented by following Eq. 5 where, w_i denotes a funny spelling word and c_j denotes an English character.

$$w_i = (c_j + c_{j+1} + c_{j+2} + ... + c_n) \tag{5}$$

If c_j and c_{j+1} are the same, it is not a funny word. However, if c_j, c_{j+1}, ... c_{j+n} are the same, we can simply normalize this word into standard word due to the fact that the maximum number of the same character in an English word is two.

When we simply put funny spelled words without the normalizing step into our system, the results will not be reasonable as show in Table 6. As we can see in Table 6, we only can find two correct words (happy and Google) for funny spelled words. PyEnchant suggested words based on the length of word and their characters so the suggested words for funny spelled words are not always adequate. Therefore, we need to normalize funny spelled words by using Eq. 5 in order to reduce their complexities. As a result, we can obtain more precise candidate words for the funny spelled words described in Table 7. We cannot find a standard word for *'juuuust'* shown in Table 6 however, we are able to find *'just'* by using our proposed approach. Therefore, we can normalize the funny spelled words which commonly considered as a noisy data into the standard English word successfully. If we do not normalize the funny spelled word such as *'baaaad'*, computer is not able to find the standard word *'bad'*.

5 Conclusion and Future Works

This paper proposed a method for normalizing non-standard words to standard English words in online social network services especially for Twitter text messages. Although, Twitter provides great convenience to user for sharing their interest, experience, knowledge and more, there are full of non-standard words in Twitter due to the restricted input policy and users have to text a message by using smartphone devices. Non-standard words bring huge obstacle when we analyze human written language in order to find semantic meanings from given texts or documents. There is a powerful NLP module based on Python named PyEnchant which is a spellchecking library. It suggests words to users when input words are not in standard English form based on the length of words and combination of words characters. The problem is that we cannot guarantee whether those suggested words are the precise one or not. Therefore, we introduced a method for finding the most appropriate word from suggested words by using Dice, Jaccard, Ochiai, and proposed coefficient similarities. We can conclude that our proposed method is strongly able to distinguish contraction and funny spelled type of NSW compared with other methods. However, the proposed approach is mainly depending on the PyEnchant. If PyEnchant does not suggest candidate words for NSW, we cannot find the standard type of words at all. Hence, we need to develop more strongly system to find the candidate word for NSW without PyEnchant's help. This is the next goal of our research.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2013R1A1A2A10011667) and financially supported by the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea (NRF) through the Human Resource Training Project for Regional Innovation.

References

1. Hwang, M., Choi, C., Kim, P.: Automatic enrichment of semantic relation network and its application to word sense disambiguation. IEEE Trans. Knowl. Data Eng. **23**(6), 845–858 (2011)
2. Steyvers, M., Tenenbaum, J.B.: The large-scale structure of semantic networks: statistical analyses and a model of semantic growth. Cogn. Sci. **29**, 41–78 (2005)
3. Choudhury, M., Saraf, R., Jain, V., Mukherjee, A., Sarkar, S., Basu, A.: Investigation and modeling of the structure of texting language. Int. J. Doc. Anal. Recogn. **10**(3), 157–174 (2007)
4. http://www.en.wikipedia.org/wiki/Twitter
5. http://www.support.twitter.com/articles/166337-the-twitter-glossary
6. Cook, P., Stevenson, S.: An unsupervised model for text message normalization. In: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity, pp. 421–432 (2009)
7. Han, B., Cook, P., Baldwin, T.: Automatically constructing a normalisation dictionary for microblogs. In: Conference on Empirical Methods in Natural Language Processing, pp. 421–432 (2012)
8. Hwang, M., Jeong, D., Kim, J., Song, S., Jung, H., Shin, J., Kim, P.: A term normalization method for efficient knowledge acquisition through text processing. Multimedia Tools Appl. **65**(1), 75–91 (2013)
9. Henriquez, C.A., Hernandez, A.: A ngrambased statistical machine translation approach for text normalization on chatspeak style communications. In: Proceedings of CAW2.0, pp. 1–5 (2009)
10. Sproat, R., Black, A.W., Chen, S., Kumar, S., Ostendorf, M., Richards, C.: Normalization of non-standard words. Comput. Speech Lang. **15**(3), 287–333 (2001)
11. Yang, J., Leskovec, J.: Patterns of temporal variation in online media. In: ACM International Conference on Web Search and Data Mining, pp. 177–186 (2011)
12. Choi, D., You, I., Kim, P.: Syntactic analysis for monitoring personal information leakage on social network services: a case study on twitter. In: Mustofa, K., Neuhold, E.J., Tjoa, A.M., Weippl, E., You, I. (eds.) ICT-EurAsia 2013. LNCS, vol. 7804, pp. 253–260. Springer, Heidelberg (2013)
13. Jackson, D.A., Sombers, K.M., Harvey, H.H.: Similarity coefficients: measures of co-occurrence and association or simply measures of occurrence? Am. Nat. **133**(3), 436–453 (1989)