Development and Evaluation of a System for Normalizing Internet Slangs in Social Media Texts

Adedoja A. Adedamola, Abiodun Modupe, and Olumuyiwa J. Dehinbo

***Abstract*— Social Media sites have changed the way people communicate. They are now the world's largest virtual communities. People use social media to make friends, communicate with each other and express their preferences and opinions about various things. As such, every day, massive number of pieces of textual information is gathered into Social Media, and such information can be leveraged for many uses. However, the limited character length for these texts has caused people to communicate unconventionally with a mixture of formal and slang words. This has created problems on two levels, The first is that social media text is often unsuitable as data for Natural Language Processing tasks such as Machine Translation, Information Retrieval and Opinion Mining, due to the its unconventionality and the irregularity of the language featured. The second is that non-native speakers of English, older Internet users and non-members of the "in-group" often find such texts difficult to understand. The normalization of these slang words is by generating plain text from the slang words. This study therefore develops a system to normalize social media texts that can be deployed as either a preprocessing step for Natural Language Processing tasks or to bridge the generational-gap when handling social media text. The system is then tested and its performance evaluated.**

***Index Terms*— normalization, social media, twitter, NLP**

I. Introduction

SOCIAL media sites, like Twitter, Facebook, Sina or Weibo (The Chinese equivalent of Twitter), have become very popular in the last 10 years [6]. They allow people to make friends, communicate with each other and express their preferences and opinions about various things. As such, every day, massive number of pieces of textual information is gathered into Social Media, and this textual information has a tremendous value if they can be processed and structured accordingly. They often contain extremely current information about world events and are reshaping the way information is published and analyzed [4]. However, the limited character length in social media e.g. twitter where users are allowed just 140 characters per tweet have caused people to communicate unconventionally with a mixture of formal and slang words (e.g How R U?) which is very unlike the style of communication found in more traditional ways of communication like e-mails or regular letters.

An example of the unconventional ways of communication is lexical variants, when standard words or phrases are written in a different form. This could be something as simple as minor spelling errors (e.g.jst is a variant of just), to abbreviations (e.g.tmi, which stands for too much information and wanna which means want to), or also jargon (e.g.bday, which is an African-American slang word for birthday). Some are also developed from phonetic attributes of words, such 4eva (forever) and 2mao (tomorrow) [6]. Twitter for example has become one of the most important social media site since its inception in 2006. It is a micro blogging service that allows users to post messages up to 140 characters in length. Once a message or tweet is posted, any twitter user in the world can see it, repost/retweet it, and reply to it. Users can search for messages based on topic or person of interest. Users can also "follow" other users which will cause all of the messages posted by a user that is being followed to be displayed on the follower's Twitter timeline. Twitter generates a lot of data, with over 200 million active users who collectively tweet over 500 million messages per day or roughly 5,800 tweets per second [14] which is way beyond what human beings can handle even with crowdsourcing. The ability to glean information off data from social media, especially from twitter is proving to be very important especially for many natural language processing (NLP) tasks, such as sentiment analysis, information extraction, summarization, information retrieval, text-to-speech etc. [7]. The remainder of this paper is organized as follows: Section II describes the issues that are intended to be addressed by the research project. Section III identifies the research questions posed by this research project. Then, Section IV enumerates the research objectives of the study. Section V briefly presents previous works done on the subject of normalization. The system design and methodology to be employed in this research project is explored in Section VI. And the architecture employed by this research study is explained in detail in section VII. Section VIII presents the results and evaluation of the experiments performed on the normalization system. Section IX includes list of ways in which the study can be extended in the future. Finally, Section X concludes this paper with a brief summary of the

Manuscript received June 11, 2015; revised July 30, 2015. Adedoja Adedamola is with the Department of Computer Science, Tshwane University of Technology, Pretoria, South Africa (Phone: +27123829681; e-mail: damdey@gmail.com).

Abiodun Modupe is with Department of Mathematical Science, University of Johannnesburg, South Africa (Phone: +27-8034059344, e-mail: abiodunmodupe@gmail.com).

Johnson Dehinbo is with the Department of Computer Science, Tshwane University of Technology, Soshanguve, North of Pretoria, South Africa (Phone: +27123829219; e-mail: jdehinbo@yahoo.com).

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

work done so far.

II. THE CONTEXT OF THE RESEARCH PROBLEM

The limited character length typical in social media sites like twitter have caused people to communicate unconventionally with a mixture of formal and slang words. Such has created problems on two levels.

Firstly, traditional NLP tools often face problems, because they are developed with Conventional domains(e.g. news, reports) in mind, which often make strong assumptions about the type of the language we are processing is of orthographic homogeneity (i.e., that there is just one way to spell you as against you/U) [6]. Slang words have the ability to interrupt and falsify Natural Language Processing tasks done on social media text accordingly [3]. Data gathered off social media, presents a different variety in the type of content in messages. Unlike more professional/traditional domains such as news and where the authors write in a professional and standardized format, in social media, users write in a much more casual environment. This does not mean that there is no formal data in microblogs, but Informal data exist and constitute a sizable portion of the data in this domain, and means that some form of pre-processing is required before NLP tasks can be carried out on the data [6]. Current tools, while practical on news articles and similar types of well written documents, perform quite poorly for Part-of-Speech (POS) tagging and Named Entity Recognition (NER) when applied to tweets. The accuracy of tools falls from 97% accuracy for news articles to about 80% for tweets [6].

Secondly, non-native speakers of the English Language, older Internet users and non-members of the "in-group" often find such means of communication difficult to understand.

Following the problems stated above, this research project aims to develop a system that normalizes internet slangs and then test for the applications usability and efficiency. The overall research question is:
How can we develop a system to normalize internet slangs in Social media text using a research compatible approach?

A. *Sub Questions*

- How can we effectively detect slang words in a stream of text?
- How can we effectively translate detected slang words to their lexical equivalent?
- How accurate is the normalized text?

B. *The Research Objectives and Sub Objectives*

To answer the research questions enumerated above, we must clearly formulate the objectives of the research project. The objectives are therefore given below:

- To develop a web application capable of accepting a stream of text and detecting the slang words contained in the text
- To develop a web application capable of translating slang words to its lexical equivalent.
- To test for the lexical accuracy of the translated text.

III. LITERATURE REVIEW

This research project focuses on the general area of normalizing non-standard words in the English vocabulary. There are many other sub-areas like spelling correction, sense disambiguation, text to speech synthesis and text conditioning under the concern of normalizing non-standard words [15]. However, this research project is interested in Normalizing slang words from social media. A lot of research work has gone into the normalization process, from the earlier times of normalizing short messages service (SMS) messages to modern time social media. However, no matter how different the normalization processes may be from each other, they all follow basically the same workflow process.

As seen in Figure 1, pre-processing is usually done first. It is usually some forms of tokenization, removal of stop words etc. This is then followed by the detection of Slang/Out-Of-Vocabulary (OOV) Words. Then the system normalizes the OOV words to their Lexical equivalent and it concatenates it back to the normalized form of the initial post.

Eranga Mapa [3] worked on Slang normalization in Social Media. An approach based on spell correction and dictionary lookup was employed to resolve tweets containing slang words to the formal version. Python was used for the development of the system by leveraging the NLTK library. A Regex based cleaner was used to eliminate unnecessary entities like URLs, emoticons, hash tags and at signs from the tweet to be normalized. Using spaces and ignoring punctuation symbols, tokenization was used on the tweets to break the stream of text into tokens. The tokens are then tagged with POS tags and then are moved to a comparator where a comparison of token with a non-hit list, the pyEnchant dictionary and the Names dictionary takes place. A spell checker is now used to check for spelling errors. An analyzer takes over and now checks for possible slang candidates for slang words. The analyzer then accesses N-gram model and select the most suitable slang candidate depending on context. Han & Baldwin [4] proposed a system where the lexical normalization of tweets is used as a preprocessing step for NLP tasks like POS tagging. The strategy employed for normalization involves three steps: (1) the generation of a confusion set, which are normalization candidates are generated for a given word; (2) the identification of ill-formed words and (3) the selection of candidates, with a standard form for tokens that have been identified as ill formed. Of all Out-Of-Vocabulary words that are said to be ill formed, the best candidate is selected from the confusion set as the basis of normalization.

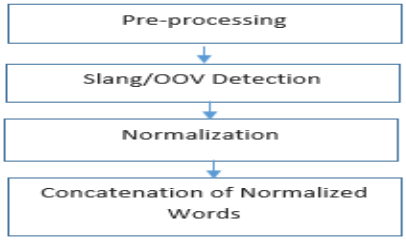

Fig. 1. Generic Normalization Model.

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Munoz-Garcıa et al. [10] used Freeling [11] for microposts tokenization. Its specific tokenization rules and its user map module were adapted for dealing with smileys and particular elements typically used in Twitter, such as hash-tags, RTs, and user IDs. Then for the identification of Twitter meta-language elements (i.e. Hash-tags, user IDs, RTs and URLs) regular expressions was used (e.g., if a token starts by the symbol \#", then it is a hash-tag). Then using a dictionary words are then classified as OOV or IV. OOV words that are classified as slang words are normalized and those OOV words that are classified as spelling errors are then corrected. The resulting normalized form of each token is concatenated, and the micro-post is amended to its normalized form.

Some researches go the machine learning route like a system which worked on the identification and transcription of slang language in twitter by crowd sourcing [9]. With the use of a twitter corpus, slang candidates were identified. Then a comparison of the twitter corpus against the English Wikipedia corpus is used to filter out-of-vocabulary (OOV) words. The words are now manually categorized by crowdsourcing as abbreviations, slangs, different language, proper names, or interjections etc. For the categorization to be automated, the study then trained a machine learning algorithm using these manually classified OOV terms. With the aid of the MaxEnt Classifier with context tweets, a fair amount of accuracy for classification task with high probabilistic scores was attained.

IV. SYSTEM DESIGN AND METHODOLOGY

A. Research Strategy

The research project follows a positivist research design model but will involve some elements of interpretive descriptive studies. The positivist research design employed involves methodologies like prototyping that will be used in the development phase and experimentation used in the evaluation phase.

B. Methodology

A Suitable methodology for this research project is the Design Science methodology. Author [2] recommends design science as a suitable methodology for research involving software development. This is in line with [16:19] because design science involves the creation of new knowledge through the design of innovative artifacts and analysis/evaluation of the performance of such artifacts. As observed by [2], a typical design science research effort as illustrated in figure 2 follows the phases/steps below:

Awareness of a problem: Previous work done in the research area is essential to the awareness of a problem. This results in the thorough understanding of the problem and the putting together of a formal or informal new research effort.

Suggestion: An understanding of the problem is going to the lead to a design that contains the new functionality envisioned. This could be in the form of the design of a prototype.

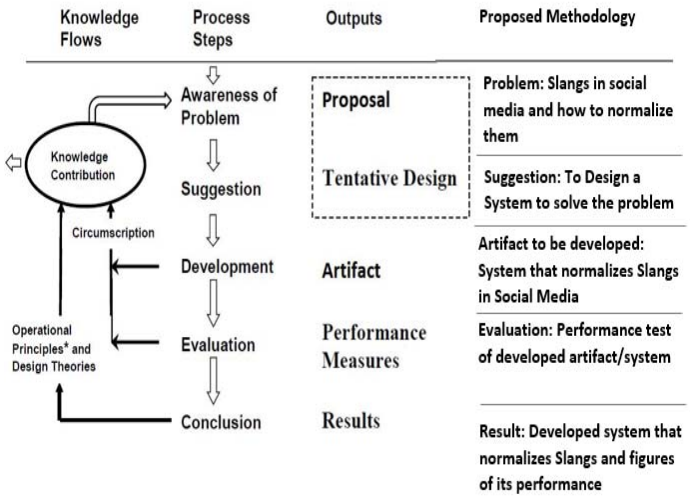

Fig. 2. Design Science Process model adopted

Development: This stage entails creative development and implementation of the proposed design. The development of the artifact takes place in this stage. It could be in the form of algorithm construction, expert system development using a high-level tool, etc. In the event of setbacks or errors, circumscription involves looping back into suggestion phase.

Evaluation: For every developed artifact, evaluation is necessary. It could be either quantitative or qualitative. The results of the evaluation and other information that was made available while constructing and implementing the software or artifact are then fed back to another round of "suggestion" through circumscription.

Conclusion: This phase is the final stage of the research project, where the results of the evaluation performed on the artifact developed are adjudged to be "good enough" despite possible minor deviations. Such deviations may be recommended as areas for further research.

V. SYSTEM ARCHITECTURE

A. Tokenizer

This component of the normalizer receives the string of text to be normalized from a textbox and breaks this string into words/tokens, twitter meta-language elements (e.g., hash-tags, user IDs), emoticons, URLs etc. The output (i.e. the list of tokens) is sent to the Token Classifier component.

This is done with the aid of a regex pattern that accepts a string and outputs out the tokens:

/([#@]?\\w+[']?\\w*|\\pP+|\\p{Sc}+|\\S~u)/u

The regex pattern tokenizer tokenizes using the php preg_match_all function which matches all occurrences of patterns in string. It accepts a string and pattern as input and outputs outs the tokens in array. The tokenizer regex is given the following rules:

- match any single character present
- match any word character
- match any kind of punctuation character
- match any currency sign
- match characters "@", "#", and with " ' "

So, the output from this tokenizer process is sent to the token classifier component.

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

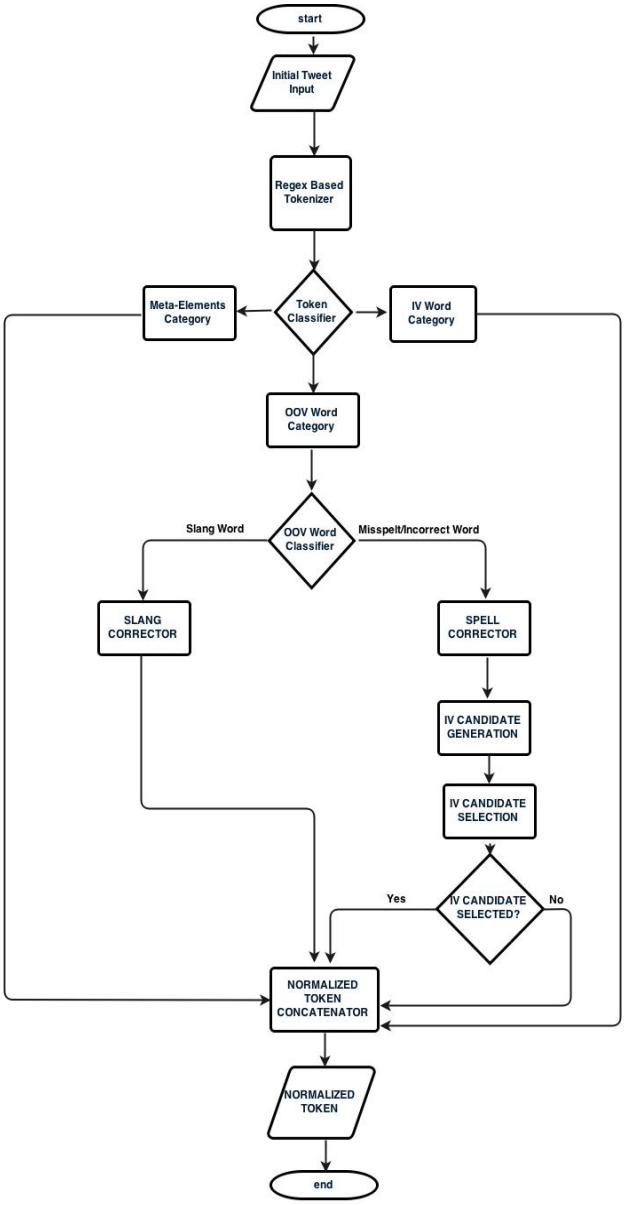

Fig. 3. System Flow Chart

B. Token Classifier

After, the tokenizer has broken down the string of text into tokens, the tokens generated are sent here to the token classifier. It now classifies each of these tokens into one of the following categories:

• **Meta-elements Category**: Such elements are detected by matching regular expressions against the token (e.g., if a token starts by the symbol \#", then it is a hash-tag). Each token classified in this category are saved in an array alongside their original index number so as to be rearranged later, and they are sent to the meta-elements component of the normalizer. The meta-elements include the followings:

- Hash-tags: Examples are #Yo, #Happy, and the regular expression used is:
  /#([A-Za-z0-9\/\.]*)/
- Mentions and user IDs: Examples are @lebo @jees1, and the regular expression used is:
  /@([A-Za-z0-9\/\.]*)/
- Retweets (RTs): The regular expression used is:
  /RT([A-Za-z0-9\/\.]*)/
- Punctuations: Here regular expressions are used to check if the token are punctuations e.g. the period(.), the comma (,) etc. The regular expression used is:
  /\W+$/

• **In Vocabulary category (IV)** –In Vocabulary Word are words in the standard English dictionary: To make this work, tokens are checked a dictionary to see if it's an IV (in-vocabulary) word, and tokens found in the dictionary. The GNU aspell dictionary is the best option, Aspell uses a combinatorial approach of both edit distance and soundex techniques. Basically, it finds all words that have a sounds-like within one or two edit distances from the original word sounds-like [1]. The aspell dictionary accepts a token and then returns a list of possible substitutes. However, the last time a windows port for aspell was released was December 22, 2002. So the alternative is the Enchant library, which aims to provide uniformity and conformity on top of all spelling libraries, and implement certain features that may be lacking in any individual provider library. Enchant supports the following backends:

- Aspell/Pspell (intends to replace Ispell) [13].

An enchant function is used to check if each token/word is contained in the dictionary, and enchant_dict_check checks whether a word is correctly spelled or not:

If the word is correctly spelled, the function returns TRUE, otherwise returns FALSE [13]. All words/tokens that return TRUE are classified as correct and then saved in an array alongside their original index number so as to be rearranged later and they are sent to the IV Word Component.

• **OOV Word Category** – Out Of Vocabulary Word-Words not in the Standard English dictionary: These are going to be misspelt, slangs words etc., basically all words that are not in the dictionary. Words/tokens to be in this category are words/tokens that return FALSE in the above process. They are then saved in an array alongside their original index number so as to be rearranged later and they are sent to the OOV Word Classifier Component.

C. OOV Word Classifier Component

This component is to receive the tokens that were previously classified as OOV Word (i.e. words not in the Standard English dictionary) by the Token Classifier Component. The component is to determine whether the OOV word/token is to be classified a slang word or if it was simply misspelt. The following processes occur here:

• First, the tokens are looked up in an array of slang words to check if they are contained in the array. The search disregards both case and accents. Tokens that return true and are found to be contained in the array of slang words are classified to be slang words and are sent to the slang corrector component.

• The tokens left from the process above are classified to be misspelt words and are sent to the spell corrector component.

D. Slang Corrector Component

Tokens sent to this component are tokens that have gone through the OOV word classifier component and were classified as slangs. These token are now passed through the slang array and this time, the slang words are substituted for

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

their regular equivalent. The words substituted are now classified as correct.

E. Spell Corrector Component

Tokens sent to this component are tokens that have gone through the OOV Word classifier component and were classified as misspelt. The aim of this component is to correct misspelt words. It contains the following processes:

1) IV Candidate Generation

For every misspelt word, there is a number of words that can be used to substitute it. To generate the IV candidates, which is a list of possible word substitutes, the PHP enchant library is used, which as explained earlier utilizes the Aspell/Pspell library using a combinatorial approach of both edit distance and soundex techniques to find suitable matches for misspelt words. This is done with the use of the php enchant_dict_suggest function –which returns an array of suggestions of the misspelt word.

array enchant_dict_suggest (resource $dict , string $word)

where:

- Dict - Dictionary resource
- Word - The word to check [13].

The output is an array of possible word substitutes for the OOV Word – also called the IV Candidates.

2) IV Candidate Selection

Here, a candidate will be selected from a list of possible candidates/matches, and this will be done with the use of a number of functions:

- The Levenshtein Function

Used to Calculate Levenshtein distance between two strings [13]:

int Levenshtein (str1, str2).

where str1 and str2 are the strings being evaluated for Levenshtein distance.

The levenshtein() function takes two parameters, these are the two strings to be compared. If the two strings are the same then the distance is zero. The higher this value is, the more distance there is between two strings. Examples are:

```
echo levenshtein('word','word'); // 0
echo levenshtein('stone','magnet'); // 4
echo levenshtein('wibble','wobble'); // 1
echo levenshtein('test','toast'); // 2
```

The similar the words, the lesser the levenshtein distance. For each word/token, the function is now applied to IV candidates and the misspelt word. Words with levenshtein score greater than 1 are very dissimilar and as such are classified as unknown and are sent to the normalized token concatenator. For those with scores of 1, if the candidates are still more than one, further processing takes place.

- Longest Common Substring (LCS)

The LCS is the longest string (or strings) that is a substring (or are substrings) of two or more strings. The more similar two words are, the more the strings they will have in common. For example, the longest common substring of the strings "ababc", "babca","abcba" is string "abc" of length 3.

This is applied to the remaining IV candidates and compared. The IV candidate with the longest common string as the misspelt word is classified as correct. And if it still cannot choose a candidate at this point, the candidate is labelled as unknown and sent to the normalized token concatenator that way.

F. Normalized Token Concatenator

This component gets all tokens from

- The metaelement component
- The IV Word Component
- The slang corrector component
- The spell corrector component
- Tokens classified as unknown

This component concatenates all the processed tokens according to their IDs, thereby outputting a normalized version of the initial string put into the system.

VI. EXPERIMENTAL RESULTS AND EVALUATION

A. Data Set Used

Experiments were performed on a datasets of microblogs which contained 10 tweets with 135 tokens. The purpose of this dataset is testing the framework in the detection and translation (normalization) of Slang words and testing the performance. The dataset was collected by randomly sampling tweets that includes slang words gotten from Twitter streaming API.

B. Experimental Results Evaluation

The metric to be used to evaluate the results is the BiLingual Evaluation Understudy (BLEU) score. The BLEU algorithm was developed by [12]. The BLEU score is used for the evaluation of translation accuracy from one language to another. It is used to rank systems according to their performance in translating texts from one language to another. What the algorithm does is to look for n-gram coincidences between a candidate text (translation produced by the system) and a set of reference texts (the translation by a human being). The BLEU score needs a gold standard, which basically is the translation done by a human being. This file is compared against the translated version and is assigned a score between 0 and 1 [5]. This score shows the similarity between the candidate text and reference text. Basically, the closer the value is to 1, the more similar the texts are.

To calculate the BLEU score, this study used iBLEU, which is an interactive web version of BLEU. iBLEU was developed to allow users to visually examine BLEU scores of candidate translations. It allows for the comparison of two different translations in an interactive manner. iBLEU is a pure Javascript implementation of the BLEU metric, and it is based on the NIST mteval script [8]. The structure of a regular tweet with slang words is different from the structure of a normalized tweet. This tool gives correct indication of the performance of the normalized system.

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE I:
BLEU SCORE OF TWEETS BEFORE & AFTER NORMALIZATIONS.

	BLEU SCORE
BEFORE NORMALIZATION	0.46
AFTER NORMALIZATION	0.83

TABLE II:
BLEU SCORE COMPARISON OF DIFFERENT NORMALIZATIONS

	BLEU SCORE
NORMALIZATION SYSTEM	0.83
tranl8it	0.48
lingo2word	0.52

Table I shows the BLEU scores of tweets before and after normalization. The normalization process had a significant effect on BLEU scores, increasing the score significantly. The normalization process adopted by this research project was also compared with other systems performing similar functions (tranl8it which is available online at http://transl8it.com/ and lingo2word which is available at http://www.lingo2word.com/translate.php). Table II contains their BLEU scores. This shows off the efficiency of the proposed system, as it outperforms similar systems. Below is an example of the translations generated by the normalization system as compared to other systems.

Regular Tweet: The things people do for some retweets lol smh

Normalized Form: The things people do for some retweets laughing out loud shaking my head

Lingo2Word: The things people do for some retweets lots of laughs signal message handling

tranl8it: The things people do for some retweets lots of luck smh

C. *Experimental Results Conclusion*

Experiments were performed on microblogs data which is the rich source for analysis of slangs, acronyms and emoticons. From the sample tweets above, it is obvious that the Token Classifier Component works as intended and can distinguish between OOV tokens/IV tokens/MetaElements. The system is successful in translating the Slang OOV tokens into their English equivalent. The normalized system produces a result with a 37% increase in the BLEU score as compared with the un-normalized data.

However, there are some problems with the status of contractions (e.g. I'm or Don't) as either words to be normalized or not. The aspell dictionary which the system depends on for spell correction does not recognize contractions as out of vocabulary words. And this significantly reduces the BLEU score.

Also, the ability of the system is limited by its slang repository, as it cannot detect a slang that is not present in its repository. So, as such the new system does not possess the ability to learn. Further, there are problems with the spell corrector component in the normalization system. The component gets more than one suggestion/candidate for each wrong spelling, making it difficult for the system to choose which the right spelling is. The checks put in place by the system to make sure it chooses the right candidate (LCS and Levenshtein distance) sometimes do fail.

VII. Conclusion

In this study, a system that normalizes internet slang in microblog text through direct substitution of internet slangs with their lexical English equivalent was developed. After investigating the different methods employed by other researches while normalizing words, this research project developed a system that normalizes slang words.

An experiment was then conducted on a dataset of slang words thereby showing that the methods described in this study achieve state-of-the-art lexical normalization on internet slags words. And the performance of the normalization system was also compared with other previously developed normalization software, thereby suggesting that our results are definitely competitive. Possible improvements in the future could make it better e.g. applying it to other languages.

References

[1] K. Atkinson, *Aspell User Manual* [Online]. Available: http://aspell.net/ [Accessed April 4 2015].

[2] J. Dehinbo, "Teaching Students on How Software Development Project can be turned into a Research Project", *Proceedings of the Society for Information Technology & Teacher Education Conference (SITE 2014)*, Jacksonville, Florida. USA. 17-21 March 2014. Online. Available from http://www.editlib.org/p/131099

[3] L. W. Eranga-Mapa, C. Chathuranga, S. Dassanayake, N. De Silva, U. Kohomban, D. Maldeniya 2012. "Text Normalization in Social Media by using Spell Correction and Dictionary Based Approach". *Systems learning*, Vol. 1. pp. 1-6.

[4] B. Han, and T. Baldwin, "Lexical normalisation of short text messages: Makn sens a# twitter". *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*-Vol. 1, 2011. Association for Computational Linguistics, pp. 368-378.

[5] M. Kaufmann, and J. Kalita, "Syntactic normalization of twitter messages". *International conference on natural language processing*, Kharagpur, India, 2010.

[6] W. Ling, and L. C. Ist, "Machine Translation in Microblogs". System cybernetics, Vol. 1. *2013.*

[7] D. Lopresti, S. Roy, K. Schulz, and L. V. Subramaniam, *Special issue on noisy text analytics*. International Journal on Document Analysis and Recognition, Vol. 12, 2009, pp.139-140.

[8] N. Madnani, "iBLEU: Interactively debugging and scoring statistical machine translation systems", *Semantic Computing (ICSC),* 2011 Fifth IEEE International Conference on, 2011. IEEE, pp. 213-214. Stanford University.

[9] B. Milde, "*Crowdsourcing slang identification and transcription in twitter language*." Gesellschaft für Informatik eV. 2013. pp. 51.

[10] O. Munoz-Garcıa, S. Vazquez, and N. Bel, "Exploiting web-based collective knowledge for micropost normalization", System cybernetics, Vol. 1. *2013.*

[11] L. Padró, and E. Stanilovsky, "Freeling 3.0: Towards wider multilinguality", System in action, Vol. 1. *2013*

[12] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, "BLEU: a method for automatic evaluation of machine translation", *Proceedings of the 40th annual meeting on association for computational linguistics*, 2002. Association for Computational Linguistics, pp. 311-318.

[13] PHP. "*PHP Function Reference",* [Online]. Available: http://php.net/manual/en/ref.enchant.php [Accessed May 1 2015].

[14] V. Prabhu, T. Lee, S. Loeb, J. H. Holmes, H. T., Gold, H. Lepor, D. F. Penson, and D. V. Makarov, "Twitter response to the United States Preventive Services Task Force recommendations against screening with prostate specific antigen". BJU international, 2013.

[15] R. Sproat, A. W. Black, S. Chen, S. Kumar, M. Ostendorf, and C. Richards, "Normalization of non-standard words", *Computer Speech & Language*, Vol. 15, pp. 287-333.

[16] V. K. Vaishnavi, and W. Kuechler, *Design Science Research Methods and Patterns*, 2008, Boca Raton, USA: Auerbach Publications.

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)