Text Normalization in Social Media by using Spell Correction and Dictionary Based Approach

Eranga Mapa[#1], Lasitha Wattaladeniya[#2], Chiran Chathuranga[#3], Samith Dassanayake[#4], Nisansa de Silva[#5], Upali Kohomban[*6], Danaja Maldeniya[*7]

[#]*Department of Computer Science and Engineering, University of Moratuwa*
Moratuwa, Sri Lanka

[1]erangamapa@gmail.com
[2]wattale@gmail.com
[3]chiranrg@gmail.com
[4]hisamith@gmail.com
[5]nisansads@uom.lk

[*]Codegen International *(Pvt) Ltd*
29,Braybrooke Street
Colombo 2, Sri Lanka

[6]upali@codegen.co.uk
[7]danajamkdt@gmail.com

Abstract— **Daily, massive number of pieces of textual information is gathered in to Social Media. They comprise a challenging style as they are formed with slang words. This has become an obstacle for processing texts in Social Media. In this paper we address this issue by introducing a pre-processing pipeline for social media text. In this solution we are focused on English texts from famous micro blogging site, Twitter [1]. We are referring a set of common slang words which we gathered by incorporating various sources. Apart from that we are resolving derivations of slang words by following spell correction based approach.**

Keywords— **Text Normalization, NLP, Social Media, Spell Checking**

I. INTRODUCTION

The world is going through the era of Social Media where Facebook and Twitter dominate. People use social media to make friends, communicate with each other and express their preferences and opinions. Nowadays Social Media has become a paradise for business and marketing. With this awakening of social media, a huge number of pieces of textual information is added into it. This textual information has a tremendous value if we can process them and structure them accordingly. But with the existence of slang words in them, it makes hard to process texts in Social Media with available tools.

Slang words have the ability to interrupt and falsify Natural Language Processing tasks done on social media text. To illustrate that ability, consider the tweet which we extracted from our data set. “at de moment he cnt just put me in da better zone thoughhhh. happy bday mic, ur a legend”. At this moment when you are going through this sentence, you will recognize some terms which doesn’t belong to decent English vocabulary. But while going through these sentences, then and there your brain will resolve the slang word to a meaningful word or phrase. When you see “cnt” and its neighbouring words “he” and “just”, you know that it is “can’t”. That is because you are not naïve with slang terms. You brain is trained with previous experiences. But when it comes to Natural Language Processing tools, they are trained and adopted to work properly with formal language. Mapping slang words to formal words can be very sensitive at some cases. A wrong mapping can result in alternations of the meaning or it may destroy semantics under the applied context. If you consider the sub phrase “ur a legend” in above example tweet, ‘ur’ can be considered as ‘your’ or ‘you are’. You can understand that its “you’re a legend” and not “your a legend”. But a direct mapping form a language tool would not. Hence it depends on the context which the word is used. Area of text normalization is not much focused as a research area and few solutions have come out. Besides, most of them have taken a manual approach when resolving slang words. With this paper, we propose a method which aggregates several strategies in order ensure a much fair accuracy for output.

II. BACKGROUND

In our implementation, we have a combination of manual and automated methods to map slang words to their meaningful forms. We are using various features of Natural Language Toolkit (NLTK) which is implemented in python for our solution. Tweets are consisting of many inappropriate items. They have at mentions like ‘@john’ and hash tags like ‘#bangkok’. Then they have URLs and emotions like ‘:)’ and ‘:(’. Initially we are sending tweets through a filter which cleans tweets by eliminating above mentioned items in them. Then we have a well compiled slang word dictionary. In addition to that dictionary, we have a tailored spell checker engine for slang words. Mappings will be cross evaluated by using both spell checker and slang dictionary before arriving at a conclusion. We also have a separate list to handle new slang words.

III. RELATED WORK

Normalization of non-standard words can be considered as a general area where our matter belongs to. There are many other sub areas like sense disambiguation, text conditioning, text to speech synthesis and spell correction under the concern of normalizing non-standard words [2]. We are particularly interested in Normalizing of slang words from social media. There are few researches that have already been carried out in that area. Knowledge that we gained from them will be the basis for this solution. In addition, we are considering the area of spell checking to improve our solution. It takes our solution beyond traditional slang mapping. Spell checking, on the other hand has a wide coverage of researches been carried out according to [3].

SMS text normalizing is a similar problem which has risen before normalizing Social Media texts. In early stages, solutions employed a dictionary substitution approach [4]. There were some popular web sites which provide a service to translate SMS language to proper English and vice versa [5]. (Karthik et al.) [4] discuss about a machine translation approach for SMS text normalization. Their solution is based on a corpus with 3000 SMS messages. After cleaning each tweet, they built a native dictionary by referring the corpus and filtering non vocabulary words from it. When there are two mappings for a word, they have used a random mapping. It's poor approach because an incorrect mapping may change the meaning of the message. They have employed Statistical Machine Translation by building a language model with Sri Language Modelling toolkit [6]. By using that language model, they have determined word alignments for proper words which appear in between slang words.

Beyond SMS text normalization, some other work has been carried out focused on slang normalizing in social media. In [7], collection of slang candidates was formed by using a twitter corpus. They have compared twitter corpus against English Wikipedia corpus to filter out of vocabulary (OOV) terms. They have manually categorized those terms using crowdsourcing [8] as abbreviations, slang, different language, proper names, interjections and other. For automated categorization, they have trained a machine learning algorithm with these manually classified OOV terms. By using MaxEnt Classifier with context tweets, they have obtained a fair amount of accuracy for classification task with high probabilistic scores. Another research has been carried out in [9] using the popular online slang word community, slangdictonary.com [10]. There can be words which appear as slang even though they exist in vocabulary. In [9], solution is also focused on such words and to providing definitions, semantic meaning, and synonyms for them. They have used a spider to scrape and extract terms from [10]. This approach resulted with more than 600,000 terms and their definitions from slangdictonary.com. Then they have used number of votes for each meaning to implement a filter. After applying, filter gave them a manageable number of terms.

IV. Pre-processing & Slang Detection

Our slang correction solution is specifically for resolving slangs appearing in social media with English language. Thus, first and foremost we need to filter English texts from Tweeter. With Tweeter public stream [11], they provide the language of each user who posted a particular tweet. By using that feature, we collected a set of English tweets. But by going through those tweets, we found some non-English tweets as well. Even though users have registered in Twitter in English, they will occasionally use other languages to tweet. Hence we wanted a way to identify the language of a particular tweet by looking at its content. During our research, we found out various solutions for language detection. In [12], they describe about a solution which calculates and compare profiles of N-gram frequencies to detect the language. Their solution has the ability to detect about 69 natural languages. But our requirement is to find weather a given tweet is written in English or not. Therefore we used a simple solution described in [13]. It uses the English Stop Words corpus from NLTK. It will count number of stop words in a text and depending on that value, it will decide whether it's English or not. After applying this for Tweeter public stream, we obtained prosperous results and decided to use this to filter English only tweets.

Before feeding into our pipeline, Twitter texts need to be pre-processed. There are various types of entities in a tweet which needs to be cleared. Following is an example tweet.

"RT @LanaParrilla: Will you be there? http://t.co/sey0eq9MG9 #thegracies YES!!! :) I will be!!"

In above tweet, you can identify the URLs, hash tags, at mentions, emotions and some punctuation symbols mixed with words. With those items in the scene, it is hard to identify slang terms. Hence we implemented a cleaning component based on python regex engine to eliminate above mentioned entities. Then we tokenized tweets using NLTK word punkt tokenizer.

With some words, people use extra characters to accentuate them. People use word 'love' as 'loveeeeee' and words like 'please' as 'plzzzzzzz'. In English language, it's rare to find words with same character three times repeated consecutively. Hence we reduced those words with more than 2 repetitive characters to 2. Then we search in both dictionary and slang list for a hit. If it does not exist, we make it 1 and search again. Again if it's not there, we forwarded the word to next step.

Next we had the problem of distinguishing between slang words and formal words. Initially we were focused on identifying non vocabulary terms with the aid of Part of Speech (POS) tagging in NLTK. If NLTK can't determine the POS tag for a particular word, it will tag that word as 'None'. They became slang candidates. For our experiment, we prepared a corpus of 1000 tweets from twitter public stream. First we POS tagged those 1000 tweets manually. Then we trained NLTK POS tagger with Brown corpus. After that we obtained POS tagging results for all 1000 tweets. After careful inspection, we found out two things. In some cases slang terms are tagged incorrectly. They were supposed to get tagged as "None". But at the end some of them were tagged as NNP (Proper Noun). We also found that system judged few names as slang terms. Following table represents the results from our experiment.

TABLE I
Confusion Matrix

		Predicted Class	
		Slangs	Names
Actual Class	Slangs	771	131
	Names	469	184

Total number of terms = 1555

Overall error percentage = 38%

Probability of false negative = 0.41

Probability of false positive = 0.37

With the 0.37 of false positive probability, NLTK POS tagger will predict many names as a slang terms. It also has a 0.41 value for false negative. Therefore slang prediction error is high. Thereafter we trained the tagger with Treebank corpus. But end results gave a similar effect. Then we decided to use dictionary comparison. We used English dictionary from PyEnchant [16] which is a spell checking library for Python.

But by only comparing against a dictionary, we can't distinguish between slangs and names. Consequently we will have a mix of slangs. To overcome that, we decided to compile a list of common names.

If you consider tweets, there are various types of names they accommodate. Mainly, people names, location names and brand names. For people names, we adopted the names corpus provided by NLTK. It has more than 8000 names categorized as male, female and pet names. Then we found a solution described in [17] which has a nice list of brand names. We implemented a spider and crawled it in [17] to extract this list. But in twitter, people don't always use brand names as they are. If you consider the brands like 'Coca Cola' and 'Mcdonalds', they use them as 'coke' and 'mac'. Therefore we manually went through the brand list that we have and added different derivations of them which we suspects that people will use. For location names we used [18]. In [18] they have not provided a list of locations that we can directly add to our dictionary. They have a location hierarchy where locations linked to each other. Therefore we implemented spider that can crawl recursively in [18] and collected location names. Finally we combined all above mentioned name lists and created a name dictionary. Then we used it to filter out names. Ultimately, we decided to use a combination of POS tagging and dictionary comparison to achieve a fair amount of accuracy. We highly considered the words which got tagged as 'None' in POS tagging process. Also we considered the words which got tagged as NNP (proper nouns). That is because according to the results we got from previous experiment, there is a possibility for a word which got tagged as NNP to be a slang word.

V. Slang Words Frequency Analysis

In the world of social networks, people tend to use slang words in various ways. Some of them uses slang words with different other meanings, for an example "cut it off" slang is used in some contexts to mean "stop". Sometimes people use words without vowels or by removing some set of insignificant letters from the word, identifying the correct word from these terms has to be done with the past experience and knowledge of the context. Sometimes users tend to write words as how they pronounce, not the actual word. Some people uses abbreviations such as "lol" to mean something like "Laugh Out Loud". Some people use repeated characters to emphasize the meaning. For an example we can take word "loooooong". There is another way of using slang words in social media. That is when new trends comes in, users start to make new slangs out of it. For an example the TV series such as "How I Met Your Mother" are referred to as "HIMYM" by some of social media users. To understand the meanings of those words uses should have an understanding about the context. As mentioned earlier there are different types of slang words. Thus, understanding the frequency of different types of slang words used in Social Media is very important in this research.

As one of the steps of understanding the degree of how many OOV words are used in twitter type of social networks, we did a frequency analysis. In that we used a twitter corpus with one million tweets. What we did in the analysis was, we first cleared the tweets by removing unwanted URLs, hash tags and white spaces. Python and Natural Language Toolkit were used in the above task. After cleaning the data, we started to run the script on those pre-processed data to calculate the frequency of OOV words. What the script basically did was, it checked each word with python PyEnchant spellchecking dctionary (dictionary which we used was "en_US") and if word doesn't contain in the dictionary, it was added as OOV term to a python dictionary. Since there were one million tweets to process and script ran nearly 48 hours. Followings are the OOV words with highest frequencies out of the one million tweets; by looking at the results we can observe that those are the very commonly used slang terms.

TABLE 2
SLANG WORD FREQUENCIES

Word	**Frequency**
im	68090
dont	39218
lol	33885
youre	11680
haha	9104
aint	7829
lmao	6946
omg	6386
didnt	6022

In the results set there were more than twelve thousand out of vocabulary words, more than half of them were garbage words with frequency less than 50. The statistics showed that the words with some kind of spelling deviations also has some weight in the results, we can't just ignore them. Hence all the different types of slang word terms have to be considered when pre-processing the twitter dataset.

Apart from the well-known meaningful slang words, there are other types of words tending to appear at the bottom of the list with some less frequency. The deviations of the same word also appear with similar number of frequency. Following are the list of those words.

TABLE 3
SLANG DEVIATION FREQUENCIES

Word	**Frequency**
aww	970
hmmm	595
ahh	560
duhhh	28
shhh	30
ewwww	27
yaya	25
grrrr	23
ehhh	20
grr	18

These types of words are called interjections. They are used to express feeling and they are not grammatically related to the rest of the sentence. These words are also a type of slang words. But we don't need to consider them when filtering slang words. We can just ignore them. But if we are planning to implement sentiment analysis using those text, these interjections plays a vital role.

VI. Slang Candidates

Nowadays people use various slang words across the world. We have much common slang used across internet. Then we have various types of slangs like American Slang and Australian Slang which are considered as native. We can consider the slangs used in internet as a mix of these. In Social Media, daily there will be a large number of new slangs added. For our purpose we need to find slang words along with their proper meaning. But there are no solid definitions for slang words. Most of the slang definitions available are made up by people who uses them in their day today communication.

For our research, we were in the need of finding a proper source to compile a list of slang words and their meanings. Then we came across Urban Dictionary [10], the most popular and informative slang word dictionary. According to [9] urban dictionary is a community which updates frequently with new and trending slang words. Almost all the words in Urban Dictionary have more than single meaning for a word. As of march 2013, Urban Dictionary possess more than 7 million TO definitions [14]. They have opened a voting system for users in order to rank the meanings according to popularity.

To compile a list of slang words, we needed to access the Urban Dictionary API. But they haven't provided an API. Urban dictionary has a view to show list of words alphabetically. So we implemented a spider to extract words from Urban Dictionary. After running spider for two hours, we found 1300251 words. It's a quite large number to handle with our mapping process. Since meanings are not formatted into a structure, it's hard to extract the correct meaning that can be replaced with the place of the slang. It not only contains slang words, but also the words with proper spelling and with slang sense depending on their usage. For our solution, we are concerned on converting slang words to their meaningful form. We discovered some other sources which have a small list compared to Urban Dictionary. Among them we found [15] which we considered as a suitable web source for our task. We also found that they have well defined meanings with a proper structure for their wordlist. Their structure is well supportive for scraping using a spider. Again we wrote a spider and extracted their words and meanings. With that we were able to collect 1047 words.

As described in section V, we created a frequency list. That list also contained 11278 entries with person names and other names. So we used a frequency threshold value of 60 to remove names and obtain a solid slang word list. With that we formed a list of 1324 most frequent slang words. Then we compared both lists and merged them to get a common list. With the comparison, we removed different slang words which ultimately maps to same formal word. To illustrate consider the forms '2moro', '2mrw' and 'tomrw'. All these forms map to the same word tomorrow. Among all the forms, we got the most frequent form '2mrw'. Slang words coming after slang detection will be first compared with the list that we created by merging. If there is a hit, we are replacing it with what we have in our direct list. If not, we will then pass them to a normal spell checker first. With that we identified incorrectly spelled words in tweets.

VII. Handling Non-Hits

Going forward with the research we faced another problem. That is to automatically include the newly originating slang words to the system. When we get a newly originated slang word, it will not get a hit from the Urban Dictionary. But when we calculate the minimum edit distance of that term, we'll get some results with some high edit distance values. After analysing those edit distance values we can come to a conclusion that if the minimum edit distance is greater than a threshold, the unidentified word as a newly originated slang word. We put that new slang word to a new list called non-hit list.

Think about a word that is passed through both normal spell checker and modified spell checker described in section VIII. If it ended up without getting resolved, we update that word into the non-hit list. If the word already exists in that list, we update its frequency. With it, we can identify new trending slang words by using a threshold frequency. In twitter, there can be some entities that will get popular in a certain period and those will be referred in tweets as slang words. To illustrate we can consider the singer 'Justin Bieber'. When he got popular, tweets referred him as 'jb'. Slang 'jb' will gain a higher frequency in our list. Likewise we can make aware the system about new trending slangs. By creating a human interface, we can allow users to enter meanings of those trending slangs. When a user enters a meaning for a possible slang word (word with a frequency higher than the threshold) in non-hit list, that word entry will be removed from it and added to the main slang word list along with its meaning. But in order to pick trending words, they need to have significant frequency values. For that we can run the system for a quite long time period and pick trending words. Therefore we can do this periodically to update the system with upcoming slang words. If a word belongs to non-hit list, there is no meaning discovered for that word yet. If that same word appears again in a tweet, we have to pass it again through our pipeline. But it's costly. Instead we decided to compare it against our non-hit list as soon as we detect it as a slang word. If we find it in non-hit list, we increments its frequency and discontinues processing with it.

VIII. Spell Checker Approach

In this section we are contending about a spell checker divergent from a normal spell checker. With various researches carried out for spell checking over the years, we decided to adopt one of them for our solution. With that focus, we improved our solution to resolve slang words to formal words not only by looking at the characters, but also considering contexts [21]. We are using edit distance to figure out character confusions. To identify how suitable is a particular candidate for replacing, we are using context based spell checking.

Minimum edit distance is a technique used in identifying the different between two words. Literary it's the minimum number of edit operations needed to transform one string to another. Basically these edit operations are "Deletion" , "Insertion" and "Substitution". When calculating the value we can use marking schema for each operation. For an example, for both insertion , deletion, cost is 1 and for substitution cost is 2. Likewise we can calculate a value for transforming each word to another word. This value is useful in spell correction applications. In spellchecking application we can calculate the minimum edit distance of misspelled word. Using those values we can guess the correct word of misspelled word as the word with minimum edit distance value. This simple technique already covers 80% of all spell errors [21].

In our research we used minimum edit distance to identify wrongly interpreted slang words. Sometimes uses make mistakes when they type correct English words, same as that they also make mistake in writing slang words. The result of a misspelled slang word will be less frequent slang word of the parent word. Text from social media contains formal words, slang words and misspelled slang words. For an example consider the word 'tomorrow', the most frequently used slang word of it is 'tmrw', therefore we can consider any misspelling of the root word 'tomorrow' as a misspelling of its most frequently used slang word 'tmrw'. Thus, when a user mistakenly types a word, we can find the most possible correct word of it using minimum edit distance.

In our research we did an experiment to identify how good this approach is. For that we referred Peter Norvig simple spell correction algorithm [20]. We modified that algorithm to cater our purposes. Given a slang word, we are trying to identify the most relevant slang word where the given slang word is a derivation of suggesting slang word. In other words, we are providing the best possible parent slang word for given slang words. Consider the word 'whatever'. With our frequency analysis, we found 'watevr', 'wotevr', 'watev' and 'watevs' as a set of slang words used for 'whatever'. Among them 'watevr' as the most frequent slang word used. Therefore we call it as the parent slang word. All the other derivations mentioned are child slang words. Following are the calculations we considered for our algorithm.

P(PSW) – Probability that a parent slang word (PSW) that will appear on a Tweet.

P(DSW) – Probability that a detected slang word (DSW) that will appear on a Tweet.

P(DSW | PSW) – Probability that slang word DSW appears in a Tweet where author meant slang word PSW.

P(PSW | DSW) - Probability that parent slang word being PSW when detected slang word DSW appears in text.

To find the best parent slang for a detected slang word, we need to find the maximum value of P(PSW | DSW).

By Baye's Theorem

$$\mathrm{P(PSW \mid DSW)} = \frac{P(\mathrm{DSW} \mid \mathrm{PSW})\, P(\mathrm{PSW})}{\mathrm{P(DSW)}}$$

$$\mathrm{argmax\ P(PSW \mid DSW)} = \mathrm{argmax}\ \frac{P(\mathrm{DSW} \mid \mathrm{PSW})\, P(\mathrm{PSW})}{\mathrm{P(DSW)}}$$

P(DSW) is the same value for all probabilities of S. Therefore we can ignore P(DSW). Now we are left with

$$\mathrm{argmax\ P(PSW \mid DSW)} \propto \mathrm{argmax\ P(DSW \mid PSW)\ P(PSW)}$$

If we calculate and find argmax P(DSW | PSW) P(PSW) , PSW in it is the correct parent slang for DSW. For all the parent slang words PSW, we calculate argmax P(PSW | DSW) and get the PSW as the parent which maximises it. To take this into action, we used the text corpus used in [20]. We created a dictionary with all possible words that can have a slang word. Then we added there frequencies to the dictionary. Consider the word 'the'. We found that the word 'the' appears 66327 times in the corpus. Which means slang 'da' of the proper word 'the' will also appear 66327 times in a complete slang corpus. In this scenario, we are only concerned about mapping a slang word to its parent slang. So we only need to train the model for slangs. Model does not need to have the knowledge on the word 'the'. It should only have to know 'da' which is the parent slang for the word 'the' according to our slang frequency list. Our next challenge was to incorporate minimum edit distance for this. If you consider the slang words 'wotevr', 'watev' and 'watevs', most of them are not more than two edit units away with parent term 'watevr'. So we decided to use minimum edit distance threshold as two. Then we prepared a set of parent slang words which have two or less than two minimum edit distances compared to detected slang word. Then by using the elements in this set, we got the highest P(PSW) value among the frequencies available for the elements in the dictionary. That is our parent slang for the detected slang. Before conclude this experiment, we tested the accuracy of this. For that we used 76 different derivations of various slang terms and tagged them with their parent slangs. After that we passed those items to our spell checker and compared the results. Out of 76 we got 47 words correctly tagged with their parent. That means 62% accuracy. Following table shows some slangs and number of derivations used.

TABLE 4
SLANG DERIVATIONS USED FOR TESTING

Parent slang	Derivations
2moro (tomorrow)	5
somthin (something)	6
watevr (whatever)	5
srsly (seriously)	3
sht (shit)	6

When considering resolving slang words, contexts will also matter. As an example consider the tweet "at least im not a prvt lyk someone here". In this sentence 'prvt' is a slang term which can be mapped to both 'private' and 'pervert'. Both have minimum edit distance value of 3. By looking at the sentence we can say that the correct mapping is 'pervert'. Therefore to pick the most suitable word, we also need to give the context knowledge to our system. In order to achieve this, we thought of using an N-gram model to map slang word with the correct context. That approach is described in section IX.

IX. CONTEXT BASED REPLACEMENT

As explained at the end of section VII, there can be situations where what the author of a tweet meant may be different from what we get using minimum edit distance. The reason is there are different character sequences which can be transformed in to the same word with same edit distance value. To resolve those kinds of situations, we have to understand the context of the writing. When suggesting for spell errors, understanding the context is not implemented in most of the spell correction tools.

We found that there are different approaches to identify words according to the context of the sentence. Two of the best approaches are n-gram modelling combined with minimum edit distance and LEXAS [22] algorithm.

N-Gram model is a statistical technique used to calculate the probability of the next word given sequence of words. For an example,

"large green ___________"
tree? mountain? frog? Car?

"swallowed the large green ________"
pill? broccoli?

Given words are w_1, w_2, w_3, w_4, …, w_{n-1}, we should be able to calculate the w_n. That is a problem of conditional probability. Therefore to get results with fair accuracy implementing a N-gram model requires a lot of corpus data. In real word usages of N-gram modelling, they have used thri-gram modelling, four-gram modelling where N is quite small.

Larger N : more information about the context of the specific instance (greater discrimination)

Smaller N : more instances in training data, better statistical estimates (more reliability)

The main idea of our research is to find correct term of misspelled words. This problem has already been solved in formal English language context. But in the context of social networks, we have some additional complexity to deal with. As we have stated in Section VI, we will first check each misspelled word in Urban Dictionary and will replace it with its original word. For an example consider the sentence,

"fnd a good art glari"

Here we can identify that "fnd" and "glari" are out of vocabulary terms. First we have to search the meaning of these terms in an Urban Dictionary. "fnd" will get a hit since it's the most frequently used slang term of 'Find'. Even though there are other possible formal words such as 'Friend' for the slang term 'fnd' we replace it with the word 'Find' by analysing the context. We can identify the most frequently used slang terms of the formal words from the results in section V. The term "glari" will not get any hit. The next step is to find what does actually mean by the word "glari" using a n-gram model. In our experiments we used maximum of 4-grams to achieve the results. For an example to resolve the given example, we take the correct 3 words before the term "glari" and feed it to the four-gram model with list of possible candidates for the next word. The candidate words list is calculated by using the minimum edit distance of 'glari' with respect to the slang words list and then mapping those to it's formal word. Finally the n-gram model will output the list of probabilities of those candidate words. We can identify the word with highest probability as the correctly spelled word.

If there are no correct 4 words before the slang term, we have to consider three-gram modelling. Likewise we have to consider different n-gram models depending on the available number of correct words. We can achieve a good understanding about the context when we use a higher n-gram model (ex : four-gram model). In our research we played more attention on training the n-gram corpus. The problem with standard n-gram corpora is that their content is not specific to social media context. Having a context specific corpus is important because it will directly affect SWAP to the results. In order to build a context specific corpus, we collected tweets which have only the most frequently used slang words and formal English words, over a 1week period. Then we replaced those slang words with its formal English word to build the corpus.

X. System Overview

In this section we are describing how a tweet with slang words will get resolved to its formal version. We implemented this system using python. For language processing tasks, we used NLTK library. When we feed a tweet, initially it will get cleaned by Regex base cleaner. It will eliminate all the redundant entities like URLs, hash tags and at mentions. Then tweet will be moved to tokenizer and POS tagger. It will tokenize words by using spaces and ignoring punctuation symbols. Thereafter it will tag resulted tokens with POS tags. After that those tagged words will get moved to comparator. Firstly, comparator will compare them in non-hit list. Then it will get relevant POS tagged words and compare with pyEnchant dictionary. Finally, it will compare with Names dictionary. As next step, words will be moved to spell checker where they will be checked for spelling errors. After that both words and tweet will enter to SHUD BE THE analyzer. Then analyzer, with the aid of modified spell checker and slang list, will check what the possible parent slang candidates for detected slang are. Finally analyzer will access N-gram model and select the most suitable meaning among the meanings that we have for identified parent slang candidates. Likewise analyzer will do this for all detected slang words and give the most possible meaning depending on context. Overview of the system is given in Fig. 1.

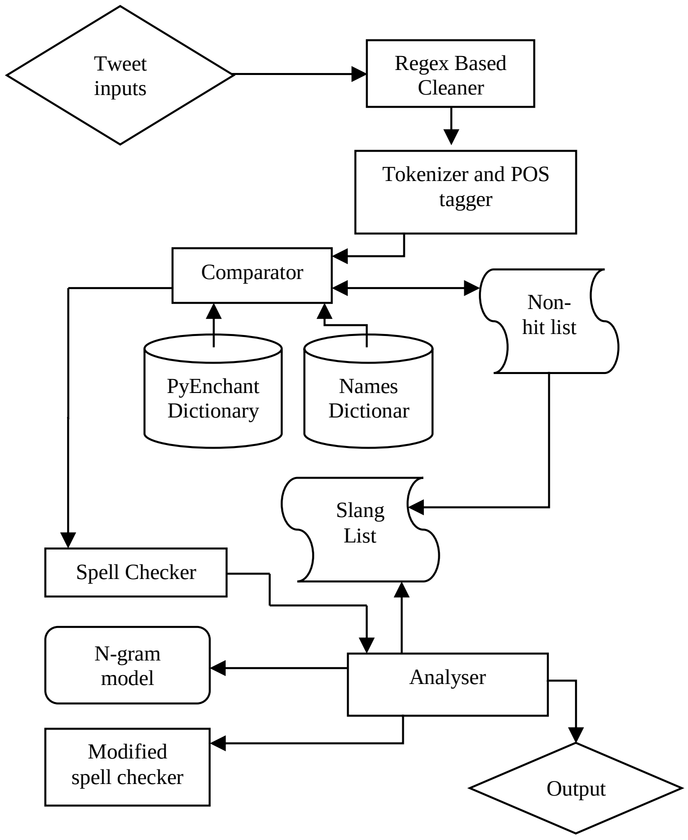

Fig. 1 System overview diagram

XI. Future Work

LEXAS algorithm is also a good alternative for n-gram model. We experimented the tool "It Makes Sense" (IMS) [23] which has been implemented including the LEXAS algorithm. To use the tool, first we had to enter a set of words that will guess what the misspelled word may be, as the input. The guessed word list is calculated as same as mentioned in the above section, by using minimum edit distance. After that the output of the algorithm is a list of probability values which represent the possibilities of each word being the resolving of the slang. The accuracy of the algorithm can be increased using more training data. Training data includes various uses of English words, mainly sentences. The training has done in a way that for each sense of a word, there is a set of sentences in which that the word has been used. By adding more sentences to this training set we can improve the accuracy. We can improve IMS by customizing it to automatically add the resolved sentences to its training data set. By collecting sets of tweets for each slang word, we can train IMS for our application. By doing so, the accuracy of our system will get increased. Going forward, by collecting more and more tweets for slang words, we can further train our system. In IMS, LEXAS algorithm is used with Wordnet [24] to identify difference senses of the same word. For our research we don't need the sense attribute of the word, but the probability of occurrences. In future, as our next improvement, we are thinking of integrating LEXAS algorithm to our system. With that we can train our system with slang word specific tweets and improve the accuracy. Currently we haven't addressed the issue of having multiple formal words for a slang word. Right now we will straight away replace the slang with the meaning from the output we get. But when there are many formal words, we can select the best one with context. This is another minor improvement for our system.

XII. Conclusion

With this paper, we have addressed one of the major problems that we face when processing raw social media text. Instead of using conventional direct mapping approach to resolve slang words, we have successfully proposed and implemented an approach which comprises a combination of automated and direct mapping. We have addressed various aspects and issues that can arise during this mapping process and have proposed effective solutions for them. In addition we have concluded that there is an added advantage of adopting a spell checker with context based spell correction for this mapping process. Results from the experiments that we have conducted have supported our reasoning and proven that this system will give a fair amount of accuracy for the task of normalizing social media texts with slang. Finally we are confident that further improvements discussed in section XI will enhance the results of our system.

XIII. Acknowledgment

We gratefully appreciate the support of project supervisors who guided when solving the problem addressed here which is a sub problem of our final year project. We also like to convey our sincere gratitude to Department of Computer Science and Engineering of University of Moratuwa for always encouraging us to be involved in research activities.

References

[1] Benjamin Milde. Twitter. [Online]. https://twitter.com/about

[2] Shankar Kumar, Mari Ostendorf, and Christopher Richards, "Normalization of non-standard words," Computer Speech and Language, vol. 15, pp. 287-333, Jan 2001.

[3] Tommi A Pirinen and Miikka Silfverberg. (2012) Improving Finite-State SpellChecker Suggestions with Part of Speech N-Grams. English. [Online].http://www.helsinki.fi/~tapirine/publications/Pirinen-2012-cicling.pdf

[4] Karthik Raghunathan and Stefan Krawczyk. (2009) Investigating SMS Text Normalization using Statistical Machine Translation. English. [Online]. http://nlp.stanford.edu/courses/cs224n/2009/fp/27.pdf

[5] (2013) Translate. [Online]. http://transl8it.com/

[6] (2011) SRI International. [Online]. http://www.speech.sri.com/projects

[7] Benjamin Milde. Crowdsourcing slang identification and transcription in twitter language. English. [Online]. http://www.ukp.tu-darmstadt.de/fileadmin/user_upload/Group_UKP/teaching/TA2012/BM_twitter_slang.pdf

[8] Wikipedia contributors. (2013, June) Crowdsourcing. [Online]. http://en.wikipedia.org/wiki/Crowdsourcing

[9] Bradley A. Swerdfeger. Assessing the Viability of the Urban Dictionary as a Resource for Slang. English. [Online]. http://www.bswerd.com/AssessingViabilitySlang.pdf

[10] Urban Dictonary. [Online].

[11] Twitter Public Stream. [Online]. https://dev.twitter.com/docs/streaming-apis/streams/public

[12] William B. Cavnar and John M. Trenkle, "N-Gram-Based Text Categorization," in *3rd Annual Symposium on Document Analysis and Information Retrieval*, 1994, pp. 161-175.

[13] Imri Goldberg. (2012, June) algorithm.co.il. [Online]. http://www.algorithm.co.il/blogs/programming/python/cheap-language-detection-nltk/

[14] Wikipedia contributors. (2012, March 2). Urban Dictionary. [Online]. http://en.wikipedia.org/wiki/Urban_Dictionary

[15] No Slang. [Online]. http://www.noslang.com/

[16] Pyenchant. [Online]. http://pythonhosted.org/pyenchant/

[17] Name Development. [Online]. http://www.namedevelopment.com/trend-names.html

[18] Falling Grain. [Online]. http://www.fallingrain.com/world/

[19] Andrew Golding and Yves Schabes, "Combining Trigram-based and feature-based methods for context-sensitive spelling correction," in *ACL '96 Proceedings of the 34th annual meeting on Association for Computational Linguistics* , Pennsylvania, 2002, pp. 71-78.

[20] Peter Norvig. Norvig. [Online]. http://norvig.com/spell-correct.html

[21] Damerau, F.J.: A techniqu for computer detection and correction of spelling errors. Common. ACM (7) (1964)

[22] Hwee Tou Ng and Hian Beng Lee, "Integrating multiple knowledge sources to disambiguate word sense: an exemplar-based approach," in ACL '96 Proceedings of the 34th annual meeting on Association for Computational Linguistics , Stroudsburg, 1996, pp. 40-47.

[23] NUS Natural Language Processing Group. [Online]. http://www.comp.nus.edu.sg/~nlp/software.html

[24] Wordnet. [Online]. http://www.wordnet.princeton.edu