Graph node rank based important keyword detection from Twitter

Mukesh Kumar and Palak Rehan
Department of Computer Science and Engineering, University Institute of Engineering and Technology, Panjab University, Chandigarh, India

Received 9 April 2018
Revised 9 August 2018
Accepted 13 August 2018

Abstract

Social media networks like Twitter, Facebook, WhatsApp etc. are most commonly used medium for sharing news, opinions and to stay in touch with peers. Messages on twitter are limited to 140 characters. This led users to create their own novel syntax in tweets to express more in lesser words. Free writing style, use of URLs, markup syntax, inappropriate punctuations, ungrammatical structures, abbreviations etc. makes it harder to mine useful information from them. For each tweet, we can get an explicit time stamp, the name of the user, the social network the user belongs to, or even the GPS coordinates if the tweet is created with a GPS-enabled mobile device. With these features, Twitter is, in nature, a good resource for detecting and analyzing the real time events happening around the world. By using the speed and coverage of Twitter, we can detect events, a sequence of important keywords being talked, in a timely manner which can be used in different applications like natural calamity relief support, earthquake relief support, product launches, suspicious activity detection etc. The keyword detection process from Twitter can be seen as a two step process: detection of keyword in the raw text form (words as posted by the users) and keyword normalization process (reforming the users' unstructured words in the complete meaningful English language words). In this paper a keyword detection technique based upon the graph, spanning tree and Page Rank algorithm is proposed. A text normalization technique based upon hybrid approach using Levenshtein distance, demetaphone algorithm and dictionary mapping is proposed to work upon the unstructured keywords as produced by the proposed keyword detector. The proposed normalization technique is validated using the standard lexnorm 1.2 dataset. The proposed system is used to detect the keywords from Twiter text being posted at real time. The detected and normalized keywords are further validated from the search engine results at later time for detection of events.

Keywords Graphs, Normalization, Social media, Spanning trees

Paper type Original Article

1. Introduction

Social media services provide access to enormous data but tendency to express in colloquial and breviate form hampers its utility in Natural Language Processing (NLP), Information Retrieval (IR), data mining and Machine Translation (MT) applications. Recently, Twitter, a popular micro-blogging service, has become a new information channel for users to receive

Applied Computing and Informatics
Vol. 17 No. 2, 2021
pp. 194-209
Emerald Publishing Limited
e-ISSN: 2210-8327
p-ISSN: 2634-1964
DOI 10.1016/j.aci.2018.08.002

and to exchange information. Every day, nearly 170 million tweets are created and redistributed by millions of active users. Twitter has several unique advantages that distinguish it from news websites, blogs, or other information channels. With the brevity guaranteed by a 140-character-message limit and the popularity of Twitter's mobile applications, users tweet and re-tweet instantly. First, tweets are created in real-time. For example, we could detect a tweet related to a shooting crime just after 10 min of shot get fired, while the news report would approximately appear 3 h later. Second, tweets have a broad coverage over events. On Twitter, millions of general users, as well as verified accounts such as news agents, organizations and public figures, are constantly publishing new tweets. Every user can report news that is happening around him or her. Thus, tweets cover nearly every aspect of daily life, from national breaking news (e.g., earthquakes), local events (e.g., car accidents), to personal feelings. Third, tweets are not isolated; they are associated with rich information by E-mail or by any other method. But the tweets are not posted in any standard format. This lack of standardization hampers NLP and MT tasks and renders huge volume of social media data useless. Therefore, there is a need to reform such text forms into standard forms. This can be achieved by normalization which is a preprocessing step for any application that handles social media text. Process of converting ill formed words into their canonical form is known as normalization. Text normalization is challenging due to the colloquial nature of tweets. For example: repeating characters such as "gooood" (can refer to god or good), presence of phonetic errors (nite → night), use of acronym (ikr → I know really) are some of the commonly seen traits in social media text.

2. Related work

Graph based keyword extraction techniques can be both supervised and unsupervised, context dependent and context independent. In this research work, many context-independent unsupervised graph based keyword extraction techniques have been explored. KeyWorld is an automatic indexing system which has been proposed by Matsou et al. [19] which extracts candidate keywords by measuring their influence on small-world properties. It captures characteristic path lengths and extended characterstic path lengths. This algorithm has been inspired by small-world phenomenon and keyGraph algorithm proposed by Ohsawa et al. [23]. Thereafter, Erkan et al. [9] proposed LexRank which is insensitive to noise in text and calculates importance of sentence (or word) using eigenvector centrality. Mihalcea et al. [20] proposed graph based TextRank model which has been originated from the concept of PageRank. The author further improved TextRank further for text summarization. In 2007, Palshikar [24] proposed hybrid and statistics based approach for keyword extraction using co-occurrence frequency measure. The author described eccentricity based keyword identification, other centrality measure based keyword extraction and proximity based keyword identification. Litvak et al. [18] proposed HITS based algorithm for keyword extraction. In 2009, for event detection and tracking in social streams, Sayyadi [29] used keyGraph algorithm which was proposed earlier by Ohsawa et al. [23]. Later, in 2011, the author introduced DegExt, a graph-based language independent key-phrase extractor. The author used degree centrality for keyword extraction. In 2013, Boudin et al. [34] compared various centrality measures for graph based Keyphrase extraction from short documents. Abilhoa et al. [1] proposed Twitter Keyword Graph (TKG) algorithm to extract keywords from Twitter data.

Normalization: Previous work attempted noisy channel model as one of the text normalization technique. Brill and Moore [8] characterized the noisy channel model based on string edits for handling the spelling errors. Toutanova and Moore [17] improved above model by embedding information regarding pronunciation. Choudhury et al. [22] proposed a supervised approach based on Hidden Markov Model (HMM) for SMS text by considering

graphemic/phonetic abbreviations and unintentional typos. Cook and stevenson [25] expanded error model by introducing probabilistic models for different erroneous forms according to sampled error distribution. This work tackled three common types: stylistic variation, prefix clipping and subsequence abbreviations. Yang and Eisenstein [33] presented a unified log linear unsupervised statistical model for text normalization using maximum likelihood framework and novel sequential montocarlo training algorithm.

Some of the previous work was based on Statistical Machine Translation (SMT) approach for normalization. SMT deals with context-sensitive text by treating noisy forms as the source language and the standard form as the target language. Aw et al. [3] proposed an approach for Short Messaging Service (SMS) text normalization using phrase-level SMT and bootstrapped phrase alignment techniques. The main drawback of SMT approach is that it needs lot of training data and it cannot accurately represent error types without contextual information.

Some researchers also treated text normalization problem as a speech recognition problem. Kobus et al. [7] convert input tokens into phonetic forms and then applied phonetic dictionary lookup to restore them into words. Beaufort et al. [26] employed finite state methods for normalizing French SMS. Kaufmann and Kalita [16] proposed a machine translation approach for syntactic normalization (instead of lexical normalization). Literature also contained dictionary based normalization approaches. Saloot et al. [27] used dictionary approach with OOV and standard form pairs as its entries. But these approaches are highly dependent on dictionary size.

Normalization of social network text is a challenging task. Han and Baldwin [5] developed an approach using classifier for identifying non-standard words and then generate candidates based on morphophonemic similarity. Liu et al. [11] developed a technique for tweet normalization by introducing character-level Conditional Random Field (CRF) sequence labeler on the edit sequences (computed for OOV words). Along with it, unigram language model and phoneme and syllable features were taken into consideration.

Gouws et al. [30] developed an approach based on string and distributional similarity along with dictionary look-up method to deal with ill-formed words [4]. Introduced similar technique based on distributional similarity and string similarity. Selection of correct forms was performed on pair-wise basis. (Hany [12] proposed an approach based on random walks on a contextual similarity bipartite graph constructed from n-gram sequences on large unlabelled text corpus. Mohammad Arshi et al. [21] proposed a tweet normalization approach. Firstly, candidates were generated by targeting lexical, phonemic and morphophonemic similarities. Then candidate selection was performed via three different probability scores (positional indexing, dependency-based frequency features and language model).

More recent approaches handle the text normalization using CRFs and neural networks. Min et al. [32] proposed a system where Long-Short Term Memory (LSTM) recurrent neural networks using character sequences and Part-Of-speech (POS) tags, had been used for predicting word-level edits. Leeman-Munk et al. [28] applied two forward field neural networks to predict normalized token for an ill-formed word. Wagner and Foster [14] proposed a generalized perceptron method to generate word edit operations with character n-grams, recurrent neural network (RNN) language model hidden activation features. Then, character language model selected the final normalization candidate. Yang and Kim [10] used an CRF based approach. CRF using both brown clusters and word embeddings were trained using canonical correlation analysis as features.

Lochter [15] proposed an ensemble system to automatically detect opinions in SMS which combine text normalization and semantic indexing techniques. Almeida [31] developed text processing approach based upon lexicographic and semantic dictionaries for semantic analysis and context detection. This technique can normalize terms as well as can create new attributes so as to change and expand original text samples in order to improve performance (redundancies and inconsistencies).

Kim et al. [13] proposed a technique for correcting misspelled words in twitter text using character n-gram method to deal with spelling information and the word n-gram method to tackle dependency of co-occurrence words. Abiodun Modupe [2] developed a semi-supervised probabilistic approach for normalizing informal short text messages. Language model probability had been used to enhance the relationships between formal and informal word. Then string similarity was employed with a linear model to include features for both word-level transformations and local context similarity.

3. The proposed work

The architecture of the proposed system is given in Figure 1. It consists of the two blocks. The function of upper block is to extract the keywords in unstructured form. (i.e. in the form of ill-formed words). The lower block role is to normalize the keywords which are unstructured in nature, to the normal and understandable form.

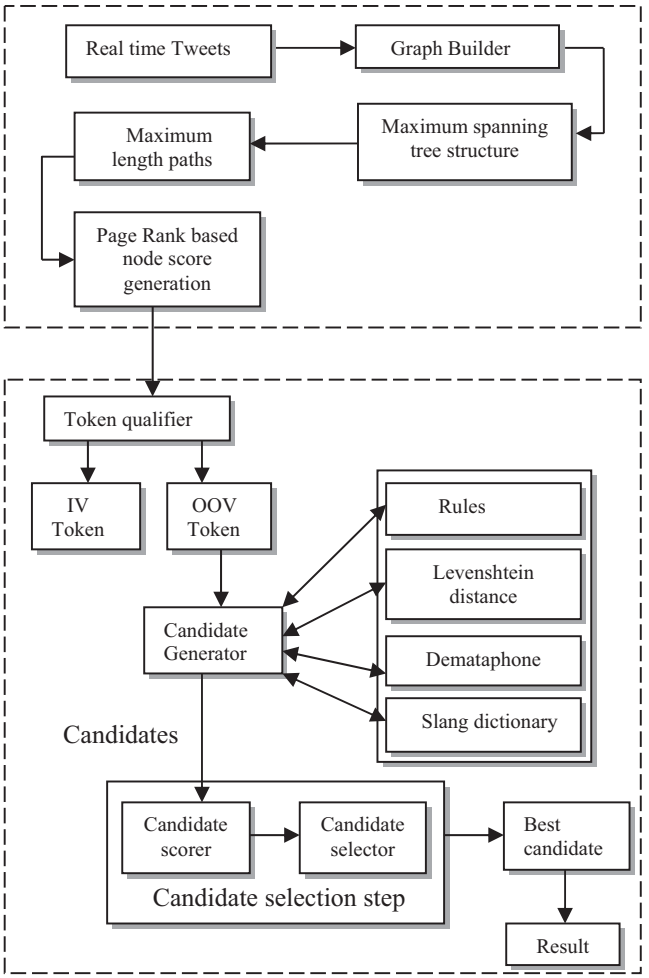

Figure 1.
The Proposed System Architecture.

Real time tweets are extracted from the Twitter and are used to create a directed weighted graph. The nodes of the directed weighted graph are the words constituting the individual tweet and there is a directed edge between two nodes if one node (word) precedes the other in the tweet. The directed graph G is given by

$$G = (V, E) \tag{1}$$

where V is the set of vertices and E is the set of edges.

Let W be the set of all tweet words then

$$V = \{\nu_1, \nu_2, \nu_3 \ldots\ldots\ldots\ldots\ldots \nu_n\} \tag{2}$$

$$\forall \nu_i \in W$$

and if there is a word sequence (tweet) in the form of

$$T_i = \{t_1 t_2 t_3 \ldots\ldots\ldots\ldots\ldots t_m\} \tag{3}$$

where $t_i \in V$

then there will be a directed edge from e_i to e_j such that

$$(e_i, e_j) \in E$$

$$\text{if } e_i \text{ and } e_j \in V$$

$$\textit{and } e_i \textit{ immediately preceeds } e_j \textit{ in some } T_i \tag{4}$$

The weight of the edge is to be increased each time the edge is repeated in any tweet.

Once a directed weighted graph is constructed, it is passed for the generation of maximum spanning tree generator. The maximum spanning tree generation is performed as per the Algorithm 1.

Algorithm 1: Maximum Spanning Tree (G)

```
{
1. result = { };
2. Sort the set E in decreasing order of edge wieght;
3. while(|result|! = n − 1 edges)
     do
     choose the next edge from the sorted list;
     if the selected edge is not generating any cycle
      then insert the edge in the Result;
4. Return result;

}
```

The weighted directed graph is passed to the Algorithm 1, which generates the maximum spanning tree using the Kruskal's minimum spanning tree generation concept. The maximum spanning tree so generated can be looked as the set of multiple word sequences which are supposed to be the most talked by users as they are having the maximum path lengths. The words contained in the maximum path may be considered as the important

keywords in unstructured form. The Page rank algorithm is used to assign importance score to the keywords and the keywords having scored more than a predefined threshold are extracted as the most important keywords which may further lead to event detection. The unstructured keywords are transformed by using the below-mentioned normalization process.

In the normalization step the tokenization of input is performed in order to extract strings. Text refining is applied on extracted strings and thereafter categorization into In-vocabulary (IV) and Out -of –vocabulary (OOV) lexicons is performed. Candidate Generation stage generates list of possible correct words for an input OOV word. In the end, Candidate Selection stage selects a best possible candidate from all generated candidates.

Token Qualifier segregates input into two heaps: OOV and IV words. OOV tokens detected by token qualifier, will be processed by the candidate generator which will generate possible normalized candidates via three different techniques: Levenshtein distance, demetaphone algorithm and dictionary mappings. Candidate selector module will work on candidate list generated by the candidate generator and will generate best possible candidate for each OOV. The Token qualifier functions as according to Algorithm 2.

Algorithm2: **Token_Qualifier**(**lexicon**)
{
1. $out\_of\_vocabulary = set()$, $In\_vocabulary = set()$
2. *Input the lexicon*
3. *If* (*lexicon* $\in$ *english dictionary word*):
   Add lexicon to In_vocabulary
   else :Add lexicon to out_of_vocabulary
4. *Return out_of_vocabulary and In_vocabulary sets*
}

OOV words of dataset (output of Algorithm 2) act as input to Algorithm 3. According to research, correct formed English words having repeating characters have maximum of two character repetitions. Thus, repetition of more than two characters in a string is trimmed off to two characters (helloooo → hello, gooood → good). Regular expressions are applied to OOV strings with alphanumeric text. Some of transformations with example are given below:

4 → fore (B4 → bfore), 2 → to (2night → tonight), 9 → ine (F9 → fine) etc.

After applying trimming and regular expressions, OOV words that are going to be processed further are obtained.

Algorithm3: **Rules**(**OOV**)
{
1. *Modified_OOV* = [ ]
2. *Input the OOV lexicon* //*OOV means out of vocabulary*
3. *If* (*OOV lexicon contains only alphabets*):
   If more than two repeated characters are present:
   Prune such characters from OOV lexicon and add newly obtained lexicon into Modified_OOV list
4. *Else if*(*OOV lexicon contains alphanumericals*):
   apply following conversions if applicable and store resultant in Modified_OOV list :
   $4 \rightarrow fore$, $2 \rightarrow to$, $9 \rightarrow ine$ ….. etc
5. *Else*: *Add OOV to Modified_OOV list*
6. *Return Modified_OOV*
}

First technique to generate candidates for OOV word is through Levenshtein distance (also known as edit distance). Edit distance is the number of applied insertions, deletions, alterations in order to transform one string into another. It is used to handle spelling errors. Edit distance >2 results in generation of large number of candidates most of which are inappropriate and at same time are complex to process. So we prefer edit distance with ≤2. Algorithm 4 takes input of Algorithm 3 and generates strings having edit distance ≤2 with respect to input OOV. In order to have precise and limited generated candidate list, string similarity measures are applied on candidate list generated via edit distance (≤2).

Algorithm4: **Levenshtein_candidates**(**Modified_OOV**)
{
1. *Levenshtein_set* = [ ], *near_by_editcandidates* = [ ]
2. *Input the Modified_OOV // output of Algorithm 3*
3. *Generate strings with edit distance ≤ 2 from input OOV.*
4. *If generated string(from step 3)english vocabulary:*
   Add generated strings to Levenshtein_set only
5. *Apply string similarity (fuzzy_ratio)between input and each of corresponding strings in Levenshtein_set :*
   5.1 *Select those pairs having maximum similarity ratio.*
   5.2 *Add above pairs in near_by_editcandidates*
6. *Return near_by_editcandidates* }

In order to handle errors due to phonemes (words that sound same), demetaphone algorithm is used. Words like nite and night are phonemes of each other. In order to have limited, precise candidate set and to reduce processing complexity, string similarity measures are applied on phonemes generated by demetaphone Algorithm 5.

Algorithm5: **Demetaphone_candidates**(**Modified_OOV**)
{
1. *Demetaphone_set* = [ ], *Relevant_demetaphone* = [ ]
2. *Input the Modified_OOV // output of algorithm 3*
3. *Generate demetaphone code for each input and english vocabulary word pair.*
4. *Add pairs having same code to Demetaphone_set.*
5. *In order to have only relevant and limited pairs, apply string similarity measures* ($fuzzy_{ratio}$) *to each pair in Demetaphone_set.*
   5.1 *Select only those pairs that have maximum similarity ratio.*
   5.2 *Add above pairs to Relevant_demetaphone set.*
6. *Return Relevant_demetaphone.*
}

Now a days, internet slangs like lol → laughing out loud and abbreviations (Cuz → because) are common in social media text. So at last we generate candidates using dictionary mapping. Output of Algorithm 6 is shown in Table 1.

Algorithm 6: **Slang_candidates**(**Modified_OOV**)
{
1. *Slang_output* = [ ]
2. *Input the Modified_OOV // output of algorithm* 3
3. *Check input in Slang dictionary*
//Slang dictionary is prepared by collecting internet abbreviations from www.noslang.com on 4*nov*, 2016
3.1 *If input is found in dictionary* :
output corresponding mapping to Slang_output
4. *Return Slang_output*
}

Candidate list generated by all three techniques (output of Algorithm 4, 5 and 6) act as input to Candidate scorer (Algorithm 7). Equal probability to each candidate in list corresponding to a OOV lexicon, is assigned. Aggregate probabilities of all those candidates which are present in more than one list is calculated by performing summation on their probabilities. This will act as score. Prepare an aggregate list by combining candidate lists of all three candidate generation techniques.

Algorithm7: $\boldsymbol{Candidate_{Scorer}}$
(**near_by_editcandidates**, **Relevant_demetaphone**, **Slang_output**)
{
1. $Edit_{score} = [\,], demetaphone_{score} = [\,], slang_{score} = [\,], combine_{score} = [\,],\ aggregate_{candidates} = [\,]$
2. *Assign equal probability to each candidates in near_by_editcandidates and store probabilities in Edit_score set.*
3. *Assign equal probability to each candidates in* $Relevant_{demetaphone}$ *and in* $Slang_{output}$ *and store them in* $demetaphone_{score}$ *and slang_score respectively.*
4. *Aggregate probabilities for common candidates that are present in all above threesetsCombine score corresponding to a* $Modified_{OOV}$ *token is computed as*:
$Edit_{score|Modified_{OOV}]} + demetaphone_{score|Modified_{OOV}]} + slang_{score|Modified_{OOV}]}$
5. *Prepare* $aggregate_{candidates}$ *set by combining candidates from*
6. *Return* $combine_{score}, aggregate_{candidates}$
}

Aggregate candidate list and score list prepared by Algorithm 7 will act as input to Algorithm 8. Select that candidate from aggregate candidate list (for an OOV lexicon) corresponding to which maximum scores are present in score list. In case, there are more than one candidate with same scores then apply part of speech tagging (POS). During POS tag,

OOV token	Generated candidates
'btw'	'by the way'
'lol'	'laughing out loud'
'yall'	'you all'
'b4'	Before

Table 1. Candidates generated via dictionary approach.

assign scores according to importance of context like noun is given highest weight followed by verb and then adjective. This will return a single best candidate for each incorrect word.

Algorithm 8: $Candidate_{selector}(aggregate_{candidates}, combine_{score})$
{
1. $Best\_candidate = [\,]$
2. $Read\ candidates\ from\ aggregate\_candidates.$
3. $Select\ those\ candidates\ from\ aggregate_{candidates}$
   $corresponding\ \ to\ which maximum\ probability\ is\ present$
   $in\ combine\_score\ set.$
4. $If(only\ one\ candidate\ is\ outputed\ from\ step\ 3)$:
   $Store\ it\ in\ best\_candidate\ set.$
5. $Else$:
   5.1 $Apply\ POS(part\ of\ speech)\ tag.$
   5.2 $Assign\ maximum\ score(say\ 1) to\ noun, followed\ by$
   $verb(0.5) and\ then\ adjective(0.25)$
   5.3 $Select\ candidate\ with\ maximum\ score\ \ //$
   $obtained\ after\ scoring\ of\ 5.2$
   5.4 $Store\ result\ in\ Best\_candidate$
6. $Return\ Best\_candidate$
}

Proposed modular approach, Algorithm 9, works on raw tweets. Preprocessing is done by removing unwanted strings (punctuations, hastags etc). Token qualifier is then called to detect OOV and IV words.

Algorithm 9: $Modular\_approach(tweet)$
{
1. $Lexicon = Preprocessor\_module(tweet)$
2. $In\_vocab, Out\_of\_vocab = Token\_Qualifier(lexicon)$
3. $Modified\_OOV = Rules(Out\_of\_vocab)$
4. $Near\_by\_editcandidates\ = Levenshtein\_candidates(Modified\_OOV)$
5. $Relevant\_demetaphone = Demetaphone\_candidates(Modified\_OOV)$
6. $Slang\_output\ =\ Slang\_candidates(Modified\_OOV)$
7. $aggregate_{candidates}, combine_{score} =$
   $Candidate\_Scorer(near\_by\_editcandidates, Relevant\_demetaphone,$
   $Slang\_output)$
8. $Best\_candidate\ Candidate\_selector(aggregate_{candidates}, combine_{score})$
9. $Result\ Best\_candidate$
}

Rules are applied to the output of the token qualifier to generate the OOV tokens which will be used for further processing. Candidates are generated via Levenshtein distance (Algorithm 4), demetaphone algorithm (Algorithm 5) and dictionary approach (Algorithm 6). In order to select best possible normalized word corresponding to a OOV word, candidate scorer (Algorithm 7) and candidate selector (Algorithm 8) are employed. The words coming after normalization may contain the stop words, which were removed further.

4. Experimental set up and results

The proposed work is a combination of two parts: keword detection and normalization. Both the parts are implemented and validated.

The proposed normalization approach has been implemented on LexNorm 1.2 dataset which was an updated version of dataset for lexical normalization described in Han et al. [4]. This dataset contains English messages sampled from Twitter API (from August to October 2010). The dataset is annotated considering one-to-one as well as one-to-many token mappings (like ttyl → talk to you later).

Processed OOV tokens (on which normalization task is performed) are divided into three broad categories: letter, letter and number and others. Letter refers to those OOV tokens that contain only alphabetic text. Spelling errors (tomoroe tomorrow), phonetic errors (u → you), stylistic variations (NOE → know) and clippings (cuz → because) are some of the traits of this category. Letter and number refers to alphanumeric text. It contains phonetic errors mostly (b4 → before). Others refers to internet slangs (like lol → laughing out loud) and abbreviations (hw → homework).

96.2% of processed OOV tokens contain spell errors, phonetic substitutions, stylish text variations and prefix clippings. 2.04% tokens are ill formed exclusively due to phonetic errors and rest have short forms mainly due to slangs and abbreviations.

Normalization results are evaluated on the basis of Precision, Recall, F-score and BLEU score. Let $T_{dataset}$ be all tokens from dataset and let OOV_t be the list of all detected OOV in dataset $\in T_{dataset}$.gen^t_{oov} be the generated candidates for an oov $\in OOV_t$. sel^t_{oov} be the best normalized candidate selected by system for an oovtoken, oov $\in OOV_t$. cor^t_{oov} be the tagged correction for an oov $\in OOV_t$. $norm^t_{oov}$ be the set of normalized oov tokens $\in OOV_t$ normalized by system

$$Precision(P) = \frac{\sum_{t \in T_{dataset}} \left| \left\{ sel^t_{oov} : sel^t_{oov} = cor^r_{oov}, sel^t_{oov} \in gen^t_{oov},\ oov \in OOV_t \right\} \right|}{\sum_{t \in T_{dataset}} \left| \left\{ norm^t_{oov} : norm^t_{oov},\ oov \in OOV_t \right\} \right|} \quad (5)$$

$$Recall(R) = \frac{\sum_{t \in T_{dataset}} \left| \left\{ sel^t_{oov} : sel^t_{oov} = cor^r_{oov}, sel^t_{oov} \in gen^t_{oov},\ oov \in OOV_t \right\} \right|}{\sum_{t \in T_{dataset}} \left| \left\{ oov :\ oov \in OOV_t \right\} \right|} \quad (6)$$

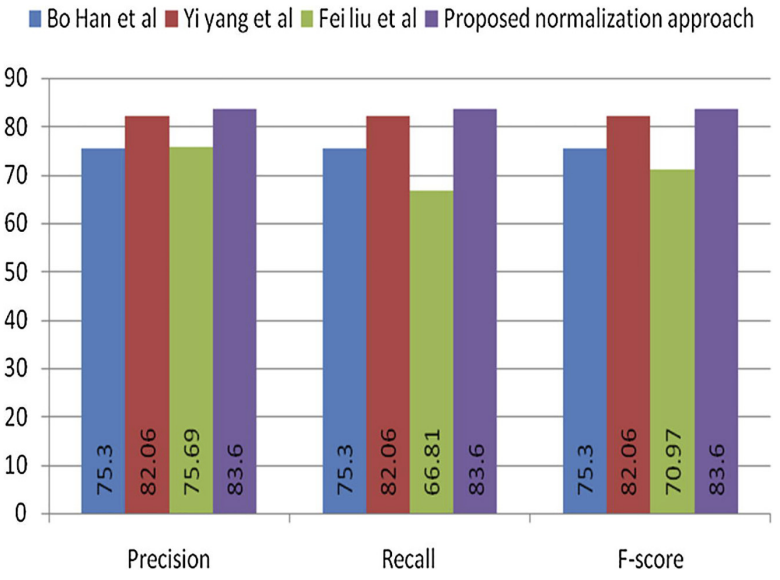

Figure 2.
Comparative Results on LexNorm 1.2.

$$F - \text{score} = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

BLEU (BiLingual Evaluation Understudy) score evaluates translation accuracy from one language to another. Translation done by human is considered as gold standard. This standard is compared with the machine translated version and then score is assigned between 0 and 1. If the machine translated file is exactly same as human translated file then a score of 1 is assigned and zero score indicates that these two files are very much different. We calculate BLEU score between normalized OOV tokens and their corresponding tagged correction.

Proposed normalization approach has achieved precision of 83.6%,recall of 83.6%, F-score of 83.6% and BLEU scores of 91.1%.

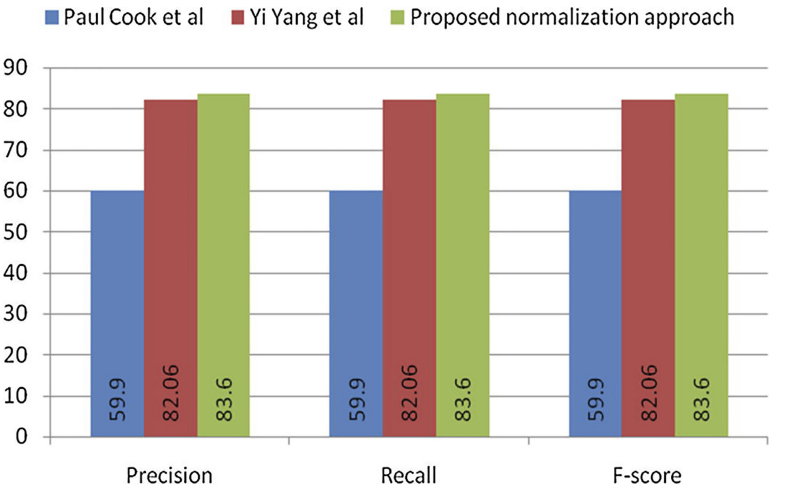

Figure 3.
Comparative results with Unsupervised Methods.

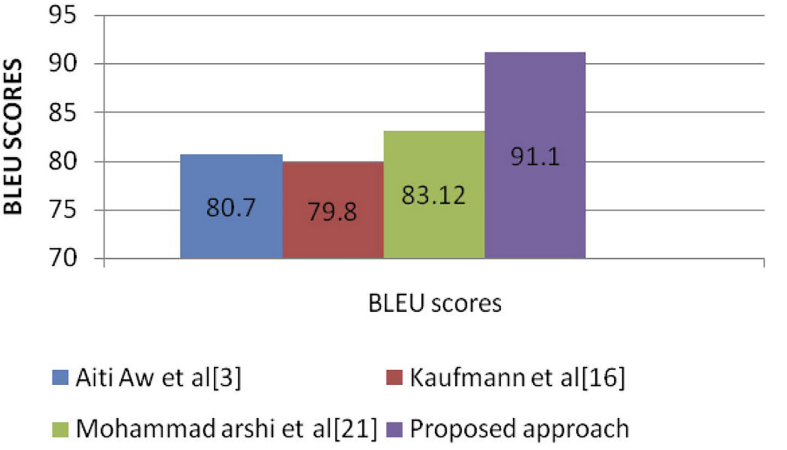

Figure 4.
Comparative results with supervised Methods.

Twitter Filters	Extracted keywords	Results of proposed normalization approach In Vocabulary tokens	Out-of-Vocabulary (OVV) tokens	Proper nouns detected	Normalized OVV words
Football	McPhail, fotbal, latest, loveisland, oficialy	Latest	fotbal, loveisland, oficialy	McPhail	fotbal: football, oficialy: officially, loveisland: lavishly
Tennis	now?, Wimbledon, Zarina, building, club, disrespectful, featuring, gifting, gruesome, history, ht?, legendary, soufert, u, vela, workout	building, club, disrespectful, featuring, gifting, gruesome, history, legendary, vela, workout, now	ht, soufert, u	Wimbledon, Zarina	soufert: subvert, u: you, ht: hit
Cricket	Werneth, captains, Indian, cricket, blesed, healthy, funiest, atribute, wonderful, suport, kasthuri, a, nz, prediction, bet, Sehwag	captains, Indian, cricket, healthy, wonderful, a, prediction, bet	blesed, funiest, atribute, suport, nz	Werneth, kasthuri, Sehwag	nz: New Zealand, blesed: blessed, funiest: funniest, atribute: attribute, suport: support
Baseball	Congresman, Guardians of the Galaxy Vol, how, Sandra, Bigest, future, played, swears, unplayable	how, future, played, swears, unplayable	bigest	Congresman, Guardians of the Galaxy Vol, Sandra	bigest: biggest
Basketball	Ano, the, Cancer, can, By, autograph, Alexander, convent, badminton, gamers, And, beyond, AAU, ATW, in	the, cancer, can, by, autograph, convent, badminton, gamers, and, beyond, in	AAU, ATW	Alexander, Ano	AAU: Amateur Athletic Union, ATW: Around The World

Table 2.
Normalized Keywords from the Proposed Approach.

Experimental results are compared on three aspects. First, we compare the proposed modular approach with the previous techniques which use LexNorm 1.2 as dataset. Second, results of the proposed approach are compared with the existing unsupervised and supervised approaches. At last, intrinsic evaluation is explored for individual techniques employed in proposed modular approach. These comparative results are shown in Figures 2–4.

As shown in Figure 2, Modular approach outperforms the existing techniques which use the same dataset, in terms of precision, recall and F-score. Text normalization task of Yang and Eisenstein [33] has slightly low results (performance of 82.06%) followed by Bo Han et al. [4] having overall performance of 75.3% and Liu et al. [11] having recall of 66.81.

Figure 3 shows that the proposed normalization approach yields better accuracy as compared to existing unsupervised methods. Modular approach has 1.54% better results than log linear model for unsupervised text normalization Yang and Eisenstein [33]. Moreover, an unsupervised model for text normalization proposed by Cook et al. [25] also has low performance (57.9% accuracy) than the proposed approach (having 83.6% performance).

Comparative study with supervised methods of Figure 4 shows that the proposed modular approach performs better than the existing supervised text normalization technique (91.1 BLEU scores). Mohammad Arshi et al. [21], tweets normalization approach using maximum entropy achieved an 83.12 BLEU scores followed by Phrase-based statistical model for SMS text normalization with BLEU scores of 80.7% (Aiti [3] and then syntactic normalization of twitter messages (by Kaufmann and Kalita [16] which has achieved 79.8% BLEU scores.

Figures 2–4 validate the normalization part of the proposed work.

The proposed keyword detection approach is used to detect the important keywords in unstructured form from the live feed of Twitter on 10-07-2017 for one thousand number of tweets in continuation after applying different Twitter filters. The results generated by the proposed keyword detection approach are shown in Table 2.

Football	
Normalized Keywords	McPhail, football, latest, lavishly, officially
Event from the Web	'Football, thank you for everything': Former Ireland midfielder Stephen . . .www.the42.ie/stephen-mcphail-retirment-3040811-Oct2016/
Tennis	
Normalized Keywords	now?, Wimbledon, Zarina, building, club, disrespectful, featuring, gifting, gruesome, history, hit?, legendary, subvert, you, vela, workout
Event from the Web	Wimbledon 2017: ArinaArinaRodionova loses to ZarinaDiyas|Herald Sun www.heraldsun.com/. . ./wimbledon. . .zarina. . ./a178a9ae2e804fc5cb551b0fdd67a952
Cricket	
Normalized Keywords	werneth, captains, Indian, cricket, blessed, healthy, funniest, attribute, wonderful, support, Kastuari, a, New Zealand, prediction, bet, Sehwag
Event from the Web	Live India vs New Zealand: Live cricket scores, updates- India timesofindia.indiatimes.com › Sports
Baseball	
Normalized Keywords	Congresman, Guardians of the Galaxy Vol, how, Sandra, biggest, future, played, swears, unplayable
Event from the Web	Iceland fans: we will still cheer on England- if you beat us www.telegraph.co.uk › News
Basketball	
Normalized Keywords	Ano, the, cancer, can, by, autograph, Alexander, convent, badminton, gamers, and, beyond, Amateur Athletic Union, Around The World
Event from the Web	BCA Indonesia open 2016|Badminton R16 M2-MD | Lee/Yoo vs Gid.. https://www.youtube.com/watch?v=OjxqtwOsa88

Table 3.
Keywords-Event Mappings for the Normalized Keywords Extracted by the Proposed Approach.

As seen from the table the unstructured keywords falling under OOV words generated by the proposed approach do not possess any meaning but are important, hence need normalization. The proposed normalization is applied on these OOV words and the result is shown in the last column of Table 2, along with other kind of extracted keywords like in vocabulary words and proper nouns.

The normalized words so obtained were searched in combination with a particular filter and the results obtained in the form of events are presented in the Table 3. A careful inspection of Table 3 suggests that the search results obtained in response to different normalized keywords is the actual event happened in relation to the filter applied and hence justifies the proposed approach as significant toward an efficient event detection mechanism.

5. Conclusion

In this paper, a keyword detection technique based upon the directed graph, maximum spanning tree and Page Rank algorithm is proposed. A text normalization technique based upon Levenshtein distance, demetaphone algorithm and dictionary mapping is proposed to work upon the unstructured keywords as produced by the proposed keyword detector. The proposed normalization technique is validated using the standard LexNorm 1.2 dataset. The proposed system is used to detect the keywords from Twitter text being posted at real time. The detected and normalized keywords are further validated from the search engine results at later time for detection of events.

References

[1] W.D. Abilhoa, L.N. de Castro, A keyword extraction method from twitter messages represented as graphs, Appl. Math. Comput. 240 (2014) 308–325.

[2] Abiodun Modupe, Turgay Celik, Vukosi Marivate, Melvin Diale, Semisupervised probabilistics approach for normalising informal short text messages, Information Communication Technology and Society (ICTAS), Conference on. IEEE, 2017.

[3] AiTi Aw, Min Zhang, Juan Xiao, Jian Su, A phrase-based statistical model for zSMS text normalization, in: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, 2006, pp. 33–40.

[4] Bo Han, Paul Cook, Timothy Baldwin, Automatically constructing a normalization dictionary for microblogs, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012), Jeju Island, Korea, 2012, pp. 421–432.

[5] Bo Han, Timothy Baldwin, Lexical normalisation of short text messages: maknsens a #twitter, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies- Volume 1, Portland,Oregon. Association for Computational Linguistics, 2011, pp. 368–378.

[7] Catherine Kobus, Franois Yvon, Graldine Damnati, Transcrire les SMS comme on reconnat la parole, in: Actes de la Confrence sur le TraitementAutomatique des Langues (TALN 08), 2008, pp. 128–138.

[8] Eric Brill, Robert C. Moore, An improved error model for noisy channel spelling correction, in: ACL 00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, Hong Kong, 2000, pp. 286–293.

[9] G. Erkan, D.R. Radev, LexRank: Graph-based lexical centrality as salience in text summarization, J. Artif. Intell. Res. 22 (2004) 457–479.

[10] Eun-Suk Yang, Yu-Seop Kim, Hallym: Named entity recognition on twitter, Proceedings of WNUT, 2015.

[11] Fei Liu, Fuliang Weng, Xiao Jiang, A broad coverage normalization system for social media language, in: Proceedings of ACL, 2012, pp. 1035–1044.

[12] Hany Hassan, Arul Menezes, Social text normalization using contextual graph random walks, Proceedings of the 51st International Conference, pages 1577–1586, Sofia, Bulgario, 2013.

[13] Jeongin Kim, Eunji Lee, Hong Taekeun, Pankoo Kim, "Correcting Misspelled Words in Twitter Text." big data technologies and applications, 7th International Conference, BDTA 2016, Seoul, South Korea, November 17–18, 2016, Proceedings, Springer, 2017.

[14] Joachim Wagner, Jennifer Foster, Dcuadapt: Learning edit operations for microblog normalisation with the generalised perceptron, Proceedings of WNUT, Beijing, China, 2015.

[15] Johannes V. Lochter, Rafael F. Zanetti, Dominik Reller, Tiago A. Almeidaa, Short text opinion detection using ensemble of classifiers and semantic indexing, Expert Syst. Appl. 62 (2016) 243–249.

[16] Joseph Kaufmann, Jugal Kalita, Syntactic normalization of Twitter messages, International Conference on Natural Language Processing, Kharagpur, India, 2010.

[17] Kristina Toutanova, Robert C. Moore, Pronunciation modeling for improved spelling correction, in: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL 02, Philadelphia, USA, 2002, pp. 144–151.

[18] M. Litvak, M. Last, Graph-based keyword extraction for single-document summarization, in: Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization, Association for Computational Linguistics, 2008, pp. 17–24.

[19] Y. Matsuo, Y. Ohsawa, M. Ishizuka, Keyworld: extracting keywords from document s small world, in: International Conference on Discovery Science, Springer, Berlin Heidelberg, 2001, pp. 271–281.

[20] R. Mihalcea, P. Tarau, in: TextRank: Bringing order into texts, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 104–401.

[21] Mohammad Arshi Saloot, Norisma Idris, Liyana Shuib, Ram Gopal Raj, AiTi Aw, Toward tweets normalization using maximum entropy, in: Proceedings of the ACL 2015 Workshop on Noisy User-generated Text, July 31, 2015, Association for Computational Linguistics, Beijing, China, 2015, pp. 19–27.

[22] Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, Anupam Basu, Investigation and modeling of the structure of texting language, Int. J. Doc. Anal. Recogn. 10 (2007) 157–174.

[23] Y. Ohsawa, N.E. Benson, M. Yachida, KeyGraph: automatic indexing by cooccurrence graph based on building construction metaphor, in: Research and Technology Advances in Digital Libraries, 1998. ADL 98. Proceedings. IEEE International Forum on, IEEE, 1998, pp. 12–18.

[24] G.K. Palshikar, Keyword extraction from a single document using centrality measures, in: International Conference on Pattern Recognition and Machine Intelligence, Springer, Berlin Heidelberg, 2007, pp. 503–510.

[25] Paul Cook, Suzanne Stevenson, An unsupervised model for text message normalization, in: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity, Association for Computational Linguistics, Boulder, USA, 2009, pp. 71–78.

[26] Richard Beaufort, Sophie Roekhaut, Louise-Amelie Cougnon, Cedrick Fairon, A hybrid rule/model-based finite-state framework for normalizing SMS messages, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 2010, pp. 770–779.

[27] M.A. Saloot, N. Idris, A. Aw, Noisy text normalization using an enhanced language model, in: Proceedings of the International Conference on Artificial Intelligence and Pattern Recognition, SDIWC, Kuala Lumpur, Malaysia, 2014, pp. 111–122.

[28] Samuel Leeman-Munk, James Lester, James Cox, Ncsu_sas_sam: Deep encoding and reconstruction for normalization of noisy text, Proceedings of WNUT, Beijing, China, 2015.

[29] H. Sayyadi, M. Hurst, A. Maykov, Event detection and tracking in social streams, ICWSM, 2009.

[30] Stephan Gouws, Dirk Hovy, Donald Metzler, Unsupervised mining of lexical variants from noisy text, in: Proceedings of the First workshop on Unsupervised Learning in NLP, Edinburgh, Scotland, 2011, pp. 82–90.

[31] Tiago A. Almeidaa, Tiago P. Silvaa, Igor Santos, Jose M. Gomez Hidalgoc, Text, normalization and semantic indexing to enhance Instant Messaging and SMS spam filtering, Knowl.-Based Syst. 108 (2016) 25–32.

[32] Wookhee Min, Bradford Mott, James Lester, James Cox, Ncsu_sas_wookhee: A deep contextual long-short term memory model for text normalization, Proceedings of WNUT, Beijing, China, 2015.

[33] Yi Yang, Jacob Eisenstein, A log-linear model for unsupervised text normalization, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013), Seattle, USA, October, 2013, pp. 61–72.

[34] Florian Boudin, Emmanuel Morin, Keyphrase extraction for n-best reranking in multi-sentence compression, in: Proceedings of NAACL-HLT 2013, Atlanta, Georgia, 9–14 June, 2013, pp. 298–305.

Corresponding author
Mukesh Kumar can be contacted at: mukesh_rai9@yahoo.com