Social Network Analysis and Mining (2018) 8:58
https://doi.org/10.1007/s13278-018-0536-8

ORIGINAL ARTICLE

Keyword extraction from micro-blogs using collective weight

Monali Bordoloi[1] · **Saroj Kr. Biswas**[1]

Received: 3 November 2017 / Revised: 5 September 2018 / Accepted: 6 September 2018 / Published online: 12 September 2018

Abstract
The growth of social networking has increased the scope of expression on a public platform. Twitter alone, being one of the most trending social networking sites, generates a huge amount of text every minute. Twitter content analysis and summarization benefits many applications such as information retrieval, automatic indexing, automatic classification, automatic clustering, automatic filtering, etc. One of the most important tasks in analyzing tweets is automatic keyword extraction. Many existing graph-based keyword extraction approaches determine keywords purely based on centrality measure. However, various features such as frequency, centrality, position, and strength of the neighbours of the keyword also affect the importance of a keyword in tweets. Therefore, this paper proposes a novel unsupervised graph-based keyword extraction method called keywords from collective weights (KCW) which determines the importance of a keyword by collectively considering various influencing features. The KCW is based on node-edge rank centrality with node weight depending on various features. The model is validated with five data sets: Uri Attack, Harry Potter, IPL, Donald Trump and IPhone5. The result of KCW is compared with three existing models. It is observed from the experimental results that the proposed method is far better than the others. The performances are shown in terms of precision, recall, and *F* measure.

Keywords Text mining · Keyword extraction · Graph-based model · Centrality measure · Sentiment analysis

1 Introduction

Keywords are described as a series of one or more words which provide a compact representation of a documents' content (Berry and Kogan 2010; Lahiri et al. 2014; Boudin 2013; Grineva et al. 2009). However, assigning keywords to documents manually is very costly, time consuming, and tedious task, in addition to this, the web is a very rich source of information which has been progressively expanding, and hence, the number of digital documents available has been progressively expanding. Consequently, automatic keyword extraction from text of social networking site has attracted the interest of researchers over the last few years. Automatic keyword extraction is an important task and hence finds its applicability in many research directions such as text mining (TM), information retrieval (IR), and natural language processing (NLP) as it enables us to represent text documents in a condensed way. The compact representation of documents can be helpful in several applications, such as automatic indexing, automatic summarization, automatic classification, automatic clustering, automatic topic detection and tracking, and automatic filtering.

With the increase of social networking, people started to share more information through different kinds of social media. Social networks are established by social interactions like co-authoring, counseling, supervising, helping academic committees, sharing views, etc. Micro-blogs have been recently attracting people to express their opinions and socialize with others. One of the most popular micro-blogging sites is twitter. Here, people put their opinions about various topics like politics, brands, products, and celebrities. (Savita and Gore 2016). This growth desires study and analysis of contents of tweets in hope of summarizing the huge collection of posts, which is called sentiment analysis. Hence, sentiment analysis has been an important and significant topic for data mining (Hemalatha and Saradhi Varma 2013). Sentiment classification is extensively helpful and useful in business intelligence applications, recommender systems, and political and administrative decisions. One of the most important tasks in sentiment analysis is keyword

✉ Saroj Kr. Biswas
bissarojkum@yahoo.com

Monali Bordoloi
monali.bordoloi@gmail.com

[1] NIT Silchar, Silchar, India

extraction. If keywords of a text are extracted properly, subject of the text can be studied and analyzed comprehensively and good decision can be made on the text.

Texts are commonly represented using the well-known vector space model (VSM); however, it results in sparse matrices which is to be dealt computationally. When target application involves twitter contents, as compared to traditional text collections, this problem becomes even worse. Because of many factors such as short length of texts, diversity in twitter contents, casualness, grammatical errors, catchwords, slangs, and the speed with which real-time content is produced; an effective technique is obligatory (Ediger et al. 2010) to extract useful keywords. Graph-based technique to extract important keywords, from a set of tweets, is appropriate in such situation and has gained popularity in the recent times.

Bellaachia and Al-Dhelaan (2012) proposed a graph-based method to extract keywords from twitter data, which uses node weight with TextRank, hereby resulting in a node-edge-weighting approach called NE rank (node and edge rank). Term frequency–inverse document frequency (TF–IDF) is used as the node weight. However, keywords in tweets do not solely depend on TF–IDF. Abilhoa and Castro (2014) proposed a graph-based technique to extract keywords from twitter data, which uses closeness and eccentricity centralities to determine node weight and degree centrality as the tie breaker. Closeness and eccentricity centralities do not work well for disconnected graphs. However, in most of the cases, the graph made from tweets becomes a disconnected graph due to the diversity of the tweet contents. Therefore, an effective graph-based keyword extraction method is required which can overcome most of the drawbacks of graph-based model including the ones cited above. This paper proposes such a graph-based keyword extraction method called keywords from collective weights (KCW). KCW analyzes the whole corpus of tweets and uses different features of a node like frequency, centrality, position, and strength of neighbours, to determine the weight of the nodes and thus extracts the important set of keywords, based on ranks obtained using NE rank and degree centrality. Here, the NE rank depends on the node weight obtained using the different features.

The remaining part of the research article is organized as follows. Section 2 presents literature survey which describes the related previous works. Section 3 discusses the proposed model in detail. An illustrative example is presented in Sect. 4 to understand the proposed model clearly. Results with discussion are presented in Sect. 5, and Sect. 6 draws some conclusions about the research work.

2 Literature survey

The keyword extraction techniques can be divided into four categories, namely, linguistic approach, machine learning approach, statistical approach and other approaches (Zahang et al. 2008). The linguistic properties of the words, sentences and documents are used in the linguistic approaches, the most commonly examined linguistic properties being lexical, syntactic, semantic, and discourse analysis (Hulth 2003; Nguyen and Kan 2007; Cohen-Kerner 2003). Supervised or unsupervised learning approach for keyword extraction is considered in machine learning approaches. In supervised machine learning approach, a model is trained on a set of known keywords and then it is used to find the keywords for unknown documents (Witten et al. 1999; Zhang et al. 2006; Medelyan and Witten 2006). Statistical approach comprises of language and domain independent simple methods which do not require the training data but uses the statistics of the words from document such as *n*-gram statistics, word frequency, TF–IDF, word co-occurrences, PAT Tree etc. to identify keywords (Chen and Lin 2010). Other approaches for keyword extraction, in general, is a combination of all approaches mentioned above.

Graph-based keyword extraction is a statistical approach for identifying the keywords. The literature provides many recent graph-based methods for keyword extraction. Litvak et al. (2011) used graph-based syntactic representation of text and web documents to propose an unsupervised cross-lingual key phrase extractor, known as DegExt. The absence of any constraints on the number of nodes in their model leads to exponentially larger graphs for larger data sets. Bellaachia and Al-Dhelaan (2012) proposed a novel graph-based keyword ranking method, called NE rank which considers word weights in addition to edge weights when calculating the ranking. NE rank forms a major part of the keywords extraction process in the proposed model. Bougouin et al. (2013) proposed an unsupervised method that aims to extract key phrases from the most important topics of a document, called TopicRank. Noun phrases belonging to a particular topic are clustered to form the vertices of the graph and then each topic is ranked using TextRank model. With the ranked clusters, keyphrases are selected. However, the current method proposed by Bougouin et al. (2013) does not provide the best solution for keyphrase selection. Zhao et al. (2011) proposed a three step algorithm that consists of keyword ranking, candidate keyphrase generation, and keyphrase ranking, for extraction of important keyphrases meant for a particular topic.

The edge-weighting scheme used for the ranking of the keywords simply uses the frequency of co-occurrence of two words in a tweet assigned to a topic. Abilhoa and Castro (2014) proposed a keyword extraction method from tweet collections that represent texts as graphs and applies centrality measures-degree, closeness, and eccentricity, for finding the relevant vertices (keywords). The performance evaluation of the proposed model is mainly inspired by this model and hence has been used as one of the baseline models for comparison purpose. Lahiri et al. (2014) extracted keywords using different centrality measures such as degree, strength, neighbourhood size—order 1, coreness, pagerank, etc. on word and noun phrase collocation networks and analyzed their performance on four benchmark data sets. Lahiri et al. (2014) observed that degree centrality measure is much simpler to use and performs well than most of the existing methods while extracting keywords and keyphrases. Beliga et al. (2015) proposed a keyword extraction method using node selectivity while using different centrality measures. Kwon et al. (2015) proposed a model for term weighting and representative keyword extraction based on graphs. Wang et al. (23) introduced average term frequency (ATF) and document frequency (DF) as an improvisation over TextRank to calculate the node weight for extracting domain-specific keyphrases. Khan et al. (2016) proposed a novel graph-based re-ranking approach, called term ranker which extracts single-word and multi-word terms using a statistical approach, identifies groups of semantically similar terms, estimates term similarity based on term embedding, and uses graph refinement and node centrality ranking to extract the top k terms. Ravinuthala et al. (2016) proposed a directed graph representation technique in which weighted edges are drawn between the words based on the theme of the document. They use both system generated keywords and manually generated keywords for performance evaluation. Nagarajan et al. (2016) presented a keyword extraction algorithm, where words of the documents are represented as nodes, the relation between the words of the documents is represented as edges, documents are represented as graphs, and keywords are extracted using degree and closeness centrality measures. In the proposed model, degree centrality is a major contributor in the determination of the top k keywords. Song et al. (2017) proposed a method which considers three major factors, namely, temporal history of the preceding utterances, topic relevance, and the participants for keyword extraction using TextRank and some graph operations. The utterances spoken by the current speaker should be considered as more important than those spoken by other participants.

3 Proposed KCW model

The KCW model considers frequency, centrality, position, and strength of neighbours of a node to calculate importance of the node. The implementation of the model is divided into four phases: pre-processing, textual graph representation, node weight assignment, and keyword extraction. The flowchart of the proposed model is shown in Fig. 1. All the phases are discussed in detail below.

3.1 Phase 1: pre-processing

Twitter is a micro-blog, where people from all over the world interact through messages, called tweets which are restricted to 140 characters. Tweets are generally very noisy for any text mining task as they contain a number of symbols that do not have any useful information and make further processing ineffective. Therefore, this model performs effective pre-processing to remove meaningless symbols, characters or words from the tweets so as to extract keywords more effectively. The steps for pre-processing are as follows:

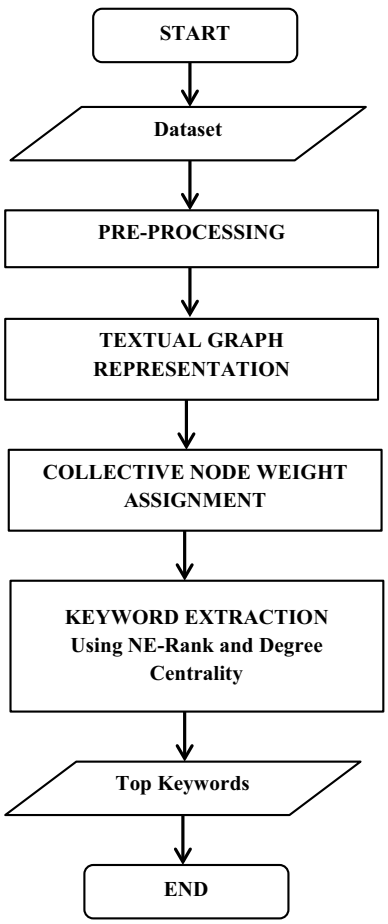

Fig. 1 Flowchart of the proposed model

(i) Remove username and retweet symbol: Most of the tweets begin with a username with the symbol '@' preceding it. Sometimes, when another user agrees, supports and shares a tweet of a user, the original tweet is said to be re-tweeted and contains the symbol RT. These usernames and retweet symbol do not contribute much to keyword extraction and act as noise and thus are removed.

(ii) Remove hashtags: The Hashtag, i.e., # before a word such as *#KarnatakaWithCongress*, is removed to get '*KarnatakaWithCongress*'.

(iii) Remove URLs: The model focuses only on the textual part of the original tweet which users provided as sufficient information source to evaluate the content and context of their coping expressions, and to extract the important keywords. In addition, the URL links mainly represent image or video files which are not the focus of the proposed model. Therefore, URLs are considered as noise, and thus are removed.

(iv) Stop word removal: A standard list of stop words such as about, be, etc. that does not contribute much in the overall meaning is created and these stop words are then removed from the set.

(v) Tokenization: Tokens are the basic constituents of a tweet/text. Each term in a tweet is treated as a token. Let T be the set of tweets which is represented as $T = \{T_1, T_2, T_3 \ldots T_i | \ i$ is the number of tweets$\}$. Each T_i in T is pre-processed and its terms are treated as tokens. Let t be the set of tokens represented as $t = \{t_1, t_2, t_3 \ldots t_k\}$. t includes tokens from all the tweets of T, where the number of tokens in the set T is k.

(vi) Removal of unimportant tokens: There are many tokens which are comparatively less important and cannot be keywords. To identify and remove these tokens, a mechanism is established, so that these tokens do not compete in the keyword extraction phase, making it more efficient. The tokens which occur less than the average occurrence frequency (AOF) are removed. AOF is determined by Eq. (1) as given below:

$$\text{AOF} = \frac{\sum \text{Frequency of each token}}{\text{Number of tokens}}. \quad (1)$$

For a given token i, if frequency $(i) <$ AOF, delete token i.

(vii) Additional white spaces left after the removal of the stop words and unimportant tokens are removed.

3.2 Phase 2: textual graph representation

Let $G = (V, E)$ be a graph, where V is the set of vertices and E is the set of edges. The textual graph is then represented as described below.

(i) Vertex assignment: Each token is used to create one vertex each. The set of vertices (V) is created from the set of tokens in the vertex assignment.

(ii) Edging: If two tokens co-occur within the same window (i.e., a tweet), then there is an edge between them. A directed edge $E_{i,j}$ is generated for each token "i" and its immediate successor "j" based on the same sequence in which they appear in the original tweets/texts. The generated graph revolves around a single topic and thus holds many common tokens that are associated with more than one tweet.

An adjacency matrix for the textual graph is created which represents weights of edges. The weights represent the strength of the relationship between two tokens. Wu et al. (2011) presented a weighted sematic graph, where the use of directed edges representing the co-occurrence relationship between two terms has shown outstanding results. Edge generation based on co-occurrence relationship between terms has encouraged many other researches (Bordag et al. 2003; Jin and Srihari 2007; Rousseau and Vazirgiannis 2013). Thus, to find the weight of each edge, frequency of the nodes, i.e., vertices and their co-occurrence frequency in the overall data set are used. The weight of an edge between vertices/nodes/terms t_i and t_j is determined by Eq. (2) (Sonawane et al. 2014):

$$W_c(i,j) = \frac{\text{freq}(i,j)}{\text{freq}(i) + \text{freq}(j) - \text{freq}(i,j)}, \quad (2)$$

where freq(i, j) is the number of times node i and j co-occur, and freq(i) and freq(j) are the occurrence frequencies of nodes i and j, respectively.

The algorithm of constructing graph is as follows.

Input: The set of extracted tokens $t_i(i=1,\ldots,m)$ obtained after the pre-processing.
Output: A graphic structure.
Algorithm:
1. Initialize the node set N, edge set E and Weight set W_c set to be empty. Set the window size equal to 1 tweet.
2. Loop
For i = 1 to n //From the first tweet to the last
{
// Process the node in the graph
If (t_i is not in N), then create a new node representing t_i, and add it into set N.
// Process the edge in the graph
For each node t_k in front of t_i within the WindowSize-1, if the corresponding terms t_i and t_k appear together in the current WindowSize, then construct a directed edge from t_i to t_k.
Count the times for t_i and t_k appearing in the dataset respectively, as well as the times freq(t_i,t_k) for t_i and t_k appearing together;
Set Weight (t_i, t_k) = W_c(i,k) (using equation 2)
}

3.3 Phase 3: node weight assignment

In keyword extraction using graph-based model, the weight of a node plays a vital role. Proper node weight evaluation leads to effective and representative keywords determination for tweets/text. Many factors affect the importance of a node. The KCW model considers five different important features to calculate the node weights as discussed below:

(i) Position of a node: Hotho et al. (2005) suggested that the position of a term is an important criterion while extracting keywords. Twitter data sets contain short texts, and thus, the probability of the first or last word in a tweet to become keyword is higher. Therefore, added weight is given to them, which is calculated as follows.

Let, freq(i) be the frequency of the token i. Then:	
a.	i. If token i is the first word then, $F[i] = \frac{n_f}{freq(i)}$ where, F[i] is the weight of i and n_f is the number of times i is the first word. ii. Else F[i] =0
b.	i. If token i is the last word then, $L[i] = \frac{n_l}{freq(i)}$ where, L[i] is the weight of i and n_l is the number of times i is the last word. ii. Else L[i] =0

(ii) Term frequency: Important keywords tend to occur more frequently in a document. Thus, the frequency of the keywords forms an essential parameter in the node weight determination. Term frequency is defined as the number of times a given term occurs in a document. Here, the term frequency is the number of times, and the term occurs in the whole set of tweets.

(iii) Selectivity centrality: Selectivity centrality is defined as the average weight on the links of a single node. Centrality measure is an indication of the importance of a node within a graph. Even for disconnected graphs, selectivity centrality works well. The strength of a vertex (v), s (v), is a sum of the weights of all edges incident with the vertex v. The selectivity centrality of vertex v is calculated as a fraction of the vertex strength and vertex degree $d(v)$ and is computed by Eq. (3):

$$\mathrm{SC}(v) = \frac{s(v)}{d(v)} \tag{3}$$

$$s(v) = \sum_{u} W_{vu}. \tag{4}$$

(iv) Distance from central node: The closer a node is to the most central node the more probability it has to be an important keyword. Therefore, importance of a node is calculated as the inverse of its distance from the central node. The central node is found using degree of the nodes. The node with the highest degree is considered as the most central. In case there is a tie in the degree of nodes, frequency of the nodes is used to break the tie. If central node is c, then the importance for a node i is determined by Eq. (5):

$$D_{C(i)} = \frac{1}{d(c,i)}, \tag{5}$$

where $d(c, i)$ is distance of node i from the central node c. This value is normalized to be within the range of 0 and 1. For the central node c, the $D_{C(c)}$ value is set to 1.

(v) Importance of neighbouring nodes: A node is considered more important if its neighbours are also important. The importance of the neighbouring nodes can be calculated as the average strength of all the neighbours, which is calculated by Eq. (6):

$$\mathrm{Neigh}_{\mathrm{Imp}}(i) = \frac{\sum_j \mathrm{Strength}(j)}{N}, \tag{6}$$

where j is any neighbour of i, N is the number of neighbours of i, and $\mathrm{Neigh}_{\mathrm{Imp}}(i)$ is the importance of the neighbouring nodes of i.

Then, the weight of any node i is calculated by Eq. (7) and is normalized to get value between 0 and 1 by Eq. (8). The normalized value of i is the final weight of i:

$$\mathrm{Node}_{\mathrm{weight}(i)} = F(i) + L(i) + \mathrm{TF}(i) + \mathrm{SC}(i) + D_{C(i)} + \mathrm{Neigh}_{\mathrm{Imp}}(i), \tag{7}$$

where $\mathrm{TF}(i)$ and $\mathrm{SC}(i)$ are the term frequency and selectivity centrality of i, respectively:

$$\mathrm{Final}_{\mathrm{weight}(i)} = \frac{\mathrm{Node\_weight}(i) - \mathrm{min\_weight}}{\mathrm{max\_weight} - \mathrm{min\_weight}}, \tag{8}$$

where Node_weight(i) is the weight of node i, min_weight is the minimum weight among all the nodes,

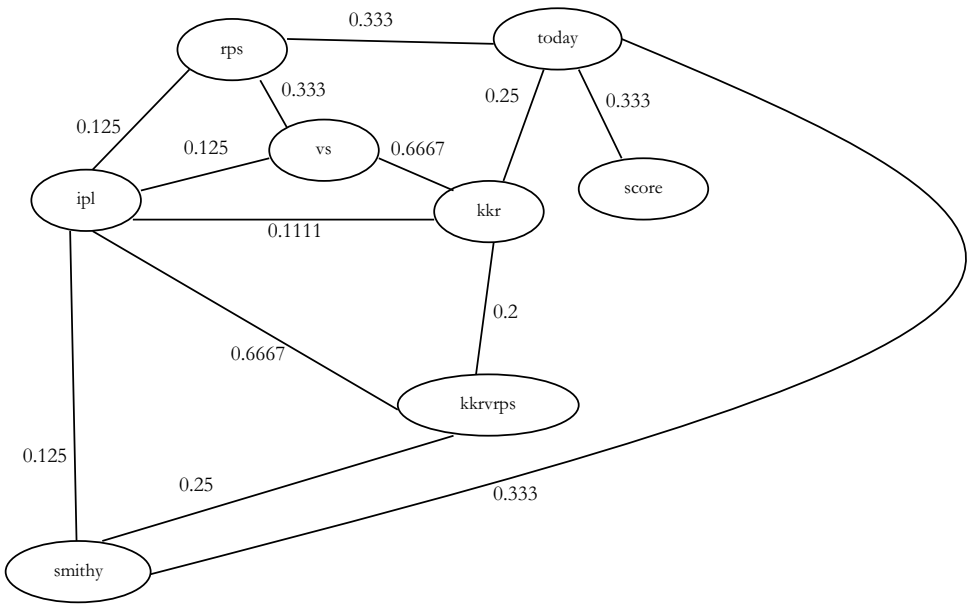

Fig. 2 Textual graph of the illustrative example

Table 1 Weights of nodes of the example

Keywords	Node weight
ipl	1.000
rps	0.185
vs	0.129
kkr	0.368
score	0.208
today	0.000
smithy	0.178
kkrvrps	0.201

Table 2 NE centrality and degree centrality for the example

Nodes	NE rank centrality	Degree centrality
ipl	0.161	0.714
rps	0.029	0.428
vs	0.021	0.428
kkr	0.058	0.571
score	0.034	0.142
today	0.000	0.571
smithy	0.028	0.428
kkrvrps	0.035	0.428

and max_weight is the maximum weight among all the nodes.

3.4 Phase 4: keyword extraction

The process of identifying keywords from a tweet/document that can appropriately represent the subject of the tweet/document is termed keyword extraction. To extract the important keywords, the proposed model uses NE rank and degree centrality.

Table 3 Top keywords

Keywords	Rank
Ipl	1
Kkr	2
Kkrvrps	3
Score	4
Rps	5
Smithy	6
vs	7
Today	8

(i) Calculate the NE rank: NE rank method uses the weight of nodes with TextRank method which results in a node-edge-weighting approach called NE rank (node and edge rank) (Bellaachia and Al-Dhelaan 2012): For a node v_i, the rank/relevance using NE rank is calculated using Eq. (9) given below:

$$R(v_i) = (1-d)\cdot W(v_i) + d\cdot W(v_i)\cdot \sum_{j:v_j\to v_i} \frac{w_{ji}}{\sum_{k:v_j\to v_k} w_{jk}} R(v_j). \quad (9)$$

In Eq. (9), d is the damping factor which denotes the probability of jumping from a node to the next node and is usually set to 0.85; and $(1-d)$ denotes the probability of jumping to a new node. $W(v_i)$ is the weight of the current node v_i. w_{ji} is the weight of the edge from the previous vertex v_j to the current vertex v_i and $\sum_{k:v_j\to v_k} w_{jk}$ is the summation of all edge weights in the previous node v_j. Here, the rank/relevance of node v_j is denoted by $R(v_j)$.

(ii) Calculate the degree centrality: Degree centrality of a node is defined as the number of edges incident on the node.

(iii) Sort the keywords: Do the following for all the terms/nodes.

Let i and j be two terms/nodes such that j occurs immediately after i in the list/text. Then:	
i.	If NE(i) = NE(j)
	If degree(i)<degree(j) Swap (i, j)
ii.	Else if NE(i) < NE(j) Swap (i, j)
iii.	Otherwise No action

Finally, n best ranked terms/nodes are selected as keywords.

4 An illustrative example

Let us use the IPL data set to illustrate the proposed model in detail. Let us consider five tweets as a sample from the data set as shown below.

(i) 'Very excited for todays IPL contest RPS vs KKR, @msdhoni vs @GautamGambhir fight! #IPL'.
(ii) '#poll who score 50 + score today #smithy #dhoni #stokes #Rahane #KKRvRPS #rpsvskkr #cricketlovers #ipl #IPL2017'.
(iii) 'RPS should be happy team today, because KKR have decided to rest NCN. He has been in prime form. #KKRvRPS #IPL @RPSupergiants @KKRiders'.
(iv) 'KKR seek to extend unbeaten run against Pune https://t.co/NdEuZIdxL5 via @cricbuzz @RPSupergiants @KKRiders #IPL'.
(v) '#RPSvKKR Predict What will be the outcome? #ipl #KKRvRPS #ipl #Smithy #Gambhir 21'.

Pre-processing is performed to remove the noise from the tweets. After removing RT symbol, @ symbol, and URLs, the tweets are as follows:

(i) 'Very excited for todays IPL contest RPS vs KKR, vs fight! IPL'.
(ii) 'poll who score 50 + score today smithy dhoni stokes Rahane KKRvRPS rpsvskkr cricketlovers ipl IPL2017'.
(iii) 'RPS should be happy team today, because KKR have decided to rest NCN. He has been in prime form. KKRvRPS IPL'.
(iv) 'KKR seek to extend unbeaten run against Pune via IPL'.
(v) 'RPSvKKR Predict What will be the outcome? ipl KKRvRPS ipl Smithy Gambhir 21'.

Table 4 Keywords extracted by three persons for five data sets

	Data set	Extracted keywords
Reader 1	Uri Attack	**Martyred, uri, attack, terror**, pm, **army**, soldiers, condemns, **surgical, strike**, terrorist, **india, Pakistan**, surgical-strike, pak, Kashmir, **uriattack**
	IPL	**ipl, rpsvkkr, dhoni, rps, ipl2017, kkr, kkrvrps**, pune, **scores**, smith, msdhoni, hit, Kolkata, Tripathi, team, playing, **match**
	Harry Potter	**harry, potter**, time, **oscar**, wins, **Radcliff**, marcus, **JKRowling**, love, books, movies, **hermione**, universal, watch, **Gryffindor**, read, **Ron**, phoenix, **scar**
	Donald Trump	**donald, trump**, russia, administration, **president, obama**, donaltrump, people, hate, Hillary, **republican**, senator, **clinton** soldiers, Oscars, republicans, **Unitedstates**
	IPhone5	**Iphone5. apple, battery**, gb, processor, **heating, price**, function, **specifications, ram, rom, camera**, wifi, product, multi-touch, service
Reader 2	Uri Attack	**uri, attack, army, surgical, strike, uriattack, india, terror, Martyred**, people, **Pakistan**, fawad, ban, artist, jawans, pm
	IPL	**ipl, rpsvkkr, kkr**, tripathi, **dhoni, rps, ipl2017, match**, rahane, uthappa, kohli, cricket, win, **scores**, commentary, **kkrvrps**
	Harry Potter	**harry, potter, hermione, ron** finally, **oscar**, harrypotter, **JKRowling**, series, **scar**, time, books, **Gryffindor**, read, winning, half-blood, prince, **Radcliff**
	Donald Trump	**donald, trump, president, clinton, obama**, house, **unitedstates**, travel, ban, hate, muslims, cost, trumprussia, tax, returns, **republican**
	IPhone5	**Iphone5, ram, rom**, color, **battery, camera, apple, specifications**, gb, size, bluetooth, **heating**, signal, **price**, touch-screen, service
Reader 3	Uri Attack	**uri, attack, army, surgical, strike, uriattack, india, terror**, surgicalstrike, **Martyred**, Kejri, Aap, Ban, Modi, terrorist, issue, Rahul, blame, **Pakistan**
	IPL	**ipl, rpsvkkr, kkr, dhoni, ipl2017**, gambhir, **kkrvrps**, top, hit, rcb, virat, Stokes, **match**, msdhoni, raina, Bravo, **scores, rps**
	Harry Potter	**JKRowling, harry, potter, oscar**, back, wins, first, **Radcliff**, franchise, wands, ron, weasley, Daniel, **scar, Gryffindor**, favourite, **Hermione, Ron**
	Donald Trump	**donald, trump, president, obama, clinton**, tax, Russian, criminals, **republican**, bush, resistance, **unitedstates**, interview, cost, refugees, Donaldjohntrump
	IPhone5	**Iphone5**, color, **camera, battery, apple, specifications, ram, heating**, features, recommendation, amazon, wifi, signal, touchscreen, **price, rom**

After removing stop words and tokenization, the set (t) of tokens is represented as

t = {excited, todays, ipl, contest, rps, vs, kkr, vs, fight, ipl, poll, score, 50, score, today, smithy, dhoni, stokes, rahane, kkrvrps, rpsvskkr, cricketlovers, ipl, ipl2017, rps, happy, team, today, kkr, decided, rest, ncn, prime, form, kkrvrps, ipl, kkr, seek, extend, unbeaten, run, pune, ipl, rpsvkkr, predict, outcome, ipl, kkrvrps, ipl, smithy, gambhir, 21}.

Using frequency of all the tokens, the average occurrence frequency (AOF) is then calculated which is obtained as 1.3659. The tokens whose frequency is less than 1.3659 are removed. Therefore, the set (t) is represented as

t = {ipl, rps, vs, kkr, score, today, smithy, kkrvrps}.

The textual graph is then represented by Fig. 2 and weights of the nodes calculated by Eq. (7) are shown in normalized form in Table 1.

For each of the nodes, the NE rank centrality and degree centrality are calculated as given in Table 2.

Using NE rank centrality with degree as tie breaker, the top keywords are obtained, as shown in Table 3.

5 Results and discussion

Data sets used in this paper are mainly tweets collected from twitter using NodeXL and Google spreadsheets using twitter Achiever. Four data sets, namely, Donald Trump, Harry Potter, IPL, and Uri Attack, are collected from twitter, containing 1000 tweets each and experiments are performed deliberately. For better assessment of the relevance of the results, the study is also conducted using a data set, namely, IPhone5 which is collected from Amazon and Flipkart that contains 1500 IPhone5 reviews.

To evaluate the performance of the proposed model and the other existing methods used for comparison, a set of keywords are defined manually, because there is no standard or correct set of keywords for any given data set. Even humans may not agree fully on the keywords that they extract for a given data set. Therefore, three persons are invited to suggest an unspecified number of keywords from the data sets. The human extractors are guided to extract the keywords based on two norms. First and most importantly, after the summarization of the topic, keywords must be extracted based on the relevance of the keyword along with its importance as a whole for the particular topic. Second, the keywords with higher frequencies are also to be considered as the important keywords. The human

Table 5 Performance of the incremental combinations of parameters in KCW model

Data set	Uri Attack			IPL			Harry Potter			Donald Trump			IPhone5		
Model	Pr	Re	F	Pr	Re	F	Pr	Re	F	Pr	Re	F	Pr	Re	F
KCW using $D_{C(i)}$	100	100	100	90	88.88	89.43	90	77.77	83.44	80	71.43	75.47	80	77.77	78.86
KCW using $D_{C(i)}$ + SC(i)	90	90	90	90	88.88	89.43	80	77.77	78.86	90	85.71	87.80	90	88.88	89.43
KCW using $D_{C(i)}$ + SC(i) + $\text{Neigh}_{\text{Imp}}(i)$	100	100	100	90	88.88	89.43	90	77.77	83.44	70	57.14	62.92	80	77.77	78.86
KCW using $D_{C(i)}$ + SC(i) + $\text{Neigh}_{\text{Imp}}(i)$ + $F(i)$ + $L(i)$	100	100	100	80	88.88	84.21	90	77.77	83.44	90	85.71	87.80	80	88.88	84.21
KCW using $D_{C(i)}$ + SC(i) + $\text{Neigh}_{\text{Imp}}(i)$ + $F(i)$ + $L(i)$ + $TF(i)$	100	100	100	90	88.88	89.43	90	77.77	83.44	90	85.71	87.80	90	88.88	89.43

extractors are instructed to select the words or terms as it is in the data sets. There must not be any alteration of words or terms in case of encountering abbreviations, plural form, joined words, words with the same stem, and words in upper or lower case. They must be selected and extracted as they appear in the data sets as there is no limit in the number of terms that can be extracted as the useful keywords and any word may seem to be important to different individuals for a particular topic. The intersection and union of the sets identified by three people are determined for each data set which ensures zero variations while determining the performances. Table 4 contains keywords extracted by three persons for five data sets and the intersection sets are represented by bold face.

Three different performance measures, namely, precision (Pr), recall (Re), and F measure, are used as evaluation metrics for keyword extraction as given in Eqs. (10)–(12):

$$\mathrm{Pr} = \frac{|\{\mathrm{Relevant}\} \cap \{\mathrm{Retrieved}\}|}{|\{\mathrm{Retrieved}\}|} \tag{10}$$

$$\mathrm{Re} = \frac{|\{\mathrm{Inter\_Relevant}\} \cap \{\mathrm{Retrieved}\}|}{|\{\mathrm{Inter\_Relevant}\}|} \tag{11}$$

$$F \text{ measure} = 2 \times \frac{\mathrm{Pr} \times \mathrm{Re}}{(\mathrm{Pr} + \mathrm{Re})}. \tag{12}$$

To compute Pr, Relevant denotes the number of retrieved keywords which appear in at least one of the human lists/sets. To compute Re, Inter_Relevant denotes the number of retrieved keywords which appear in the intersection set of the three human lists. Precision may be defined as the probability that a keyword is relevant given that it is returned by a system and recall as the probability that a relevant keyword is only returned (Goutte et al. 2005). The intersection set consists of the keywords which are approved by all the three persons to be relevant for a particular topic. Therefore, to maintain zero variation in the consideration of the actual relevant keywords and also to maintain the effectiveness of the system, the intersection is considered only for the recall. However, those keywords which are found to be relevant by at least one of the three persons cannot be neglected and hence are considered while calculating the precision.

To show how different features are sensitive or important to find the overall accuracy of the proposed model, an incremental approach is adopted to make five different combinations. The different features are added one by one in KCW model for experimentation. Performances of five different combinations obtained by adding the features one by one are shown in Table 5 in percentage when the number of retrieved keywords is 10.

Following observations are studied from the experimental results, as shown in Table 5:

(i) Using only distance from central node, $D_{C(i)}$ for the node weight assignment, KCW produces the same accuracy for three data sets, namely, Uri Attack, Harry Potter and IPL, and slightly lower accuracy for the Donald Trump and Iphone5 data set as compared to that when all parameters are used.

(ii) For Harry Potter data set, KCW produces slightly higher accuracy when summation of $D_{C(i)}$ and SC(i)

Table 6 Performances in Uri Attack data set

Model	Edge-weighting mechanism	Pr in %	Re in %	F in %
TKG with closeness centrality (Abilhoa and Castro 2014)	W_l	60	60	60
	W_f	50	30	37.5
	$W_{l/f}$	70	50	58.33
TKG with eccentricity centrality (Abilhoa and Castro 2014)	W_l	90	80	84.7
	W_f	30	10	15
	$W_{l/f}$	50	30	37.5
TKG with Eigen centrality (Abilhoa and Castro 2014)	W_l	30	30	30
	W_f	30	30	30
	$W_{l/f}$	60	30	40
TKG with TextRank centrality (Abilhoa and Castro 2014)	W_l	90	80	84.7
	W_f	90	80	84.7
	$W_{l/f}$	90	70	78.75
TKG with selectivity centrality (Abilhoa and Castro 2014)	W_c	70	50	58.33
KEGBA with degree (Nagarajan et al. 2016)	W_f	100	80	88.89
KEGBA with closeness (Nagarajan et al. 2016)	W_f	70	50	58.33
NE rank using TF–IDF (Bellaachia and Al-Dhelaan 2012)	W_{f2}	60	30	40
Proposed KCW	W_c	100	100	100

Table 7 Performances in IPL data set

Model	Edge-weighting mechanism	Pr in %	Re in %	*F* in %
TKG with closeness centrality (Abilhoa and Castro 2014)	W_1	70	66.66	68.29
	W_f	70	55.55	61.94
	$W_{1/f}$	70	55.55	61.94
TKG with eccentricity centrality (Abilhoa and Castro 2014)	W_1	20	11.11	14.28
	W_f	40	22.22	28.57
	$W_{1/f}$	40	22.22	28.57
TKG with Eigen centrality (Abilhoa and Castro 2014)	W_1	70	66.66	68.29
	W_f	60	55.55	57.69
	$W_{1/f}$	60	55.55	57.69
TKG with TextRank centrality (Abilhoa and Castro 2014)	W_1	80	77.77	78.86
	W_f	80	77.77	78.86
	$W_{1/f}$	80	77.77	78.86
TKG with selectivity centrality (Abilhoa and Castro 2014)	W_c	70	66.66	68.29
KEGBA with degree (Nagarajan et al. 2016)	W_f	80	77.77	78.86
KEGBA with closeness (Nagarajan et al. 2016)	W_f	70	55.55	61.94
NE rank using TF–IDF (Bellaachia and Al-Dhelaan 2012)	W_{f2}	60	55.55	57.69
KCW	W_c	90	88.88	89.43

Table 8 Performances in Harry Potter data set

Model	Edge-weighting mechanism	Pr in %	Re in %	*F* in %
TKG with closeness centrality (Abilhoa and Castro 2014)	W_1	70	66.66	68.29
	W_f	70	55.55	61.94
	$W_{1/f}$	70	66.66	68.29
TKG with eccentricity centrality (Abilhoa and Castro 2014)	W_1	40	33.33	36.36
	W_f	50	33.33	39.99
	$W_{1/f}$	50	33.33	39.99
TKG with Eigen centrality (Abilhoa and Castro 2014)	W_1	60	22.22	32.43
	W_f	50	22.22	30.77
	$W_{1/f}$	60	22.22	32.43
TKG with TextRank centrality (Abilhoa and Castro 2014)	W_1	90	77.77	83.44
	W_f	90	77.77	83.44
	$W_{1/f}$	90	77.77	83.44
TKG with selectivity centrality (Abilhoa and Castro 2014)	W_c	60	66.66	63.15
KEGBA with degree (Nagarajan et al. 2016)	W_f	70	77.77	73.68
KEGBA with closeness (Nagarajan et al. 2016)	W_f	70	55.55	61.94
NE rank using TF–IDF (Bellaachia and Al-Dhelaan 2012)	W_{f2}	90	66.66	76.59
KCW	W_c	90	77.77	83.44

is used for assigning node weights than KCW model using all the parameters. For IPL, Donald Trump data set, and IPhone5, KCW produces the same results when all the parameters are used and when summation of $D_{C(i)}$ and SC(i) is considered. However, a slight decrease in accuracy is encountered for Uri Attack data set when only $D_{C(i)}$ and SC(i) is used for node weight assignment.

(iii) For three data sets, the results using $D_{C(i)}$, SC(i) and $\text{Neigh}_{\text{Imp}}(i)$ and using all the parameters in KCW model are the same. However, KCW model using all the parameters produces more accu-

Table 9 Performances in Donald Trump data set

Model	Edge-weighting mechanism	Pr in %	Re in %	*F* in %
TKG with closeness centrality (Abilhoa and Castro 2014)	W_1	80	85.71	82.76
	W_f	70	85.71	77.06
	$W_{1/f}$	70	85.71	77.06
TKG with eccentricity centrality (Abilhoa and Castro 2014)	W_1	40	28.57	33.33
	W_f	40	28.57	33.33
	$W_{1/f}$	40	28.57	33.33
TKG with Eigen centrality (Abilhoa and Castro 2014)	W_1	50	42.86	46.16
	W_f	50	42.86	46.16
	$W_{1/f}$	40	28.57	33.33
TKG with TextRank centrality (Abilhoa and Castro 2014)	W_1	80	85.71	82.76
	W_f	80	85.71	82.76
	$W_{1/f}$	80	85.71	82.76
TKG with selectivity centrality (Abilhoa and Castro 2014)	W_c	80	71.43	75.47
KEGBA with degree (Nagarajan et al. 2016)	W_f	80	85.71	82.76
KEGBA with closeness (Nagarajan et al. 2016)	W_f	70	57.14	62.92
NE rank using TF–IDF (Bellaachia and Al-Dhelaan 2012)	W_{f2}	80	85.71	82.76
KCW	W_c	90	85.71	87.80

Table 10 Performances in IPhone5 data set

Model	Edge-weighting mechanism	Pr in %	Re in %	*F* in %
TKG with closeness centrality (Abilhoa and Castro 2014)	W_1	60	66.66	63.15
	W_f	70	66.66	68.29
	$W_{1/f}$	80	77.77	78.86
TKG with eccentricity centrality (Abilhoa and Castro 2014)	W_1	90	77.77	83.44
	W_f	90	77.77	83.44
	$W_{1/f}$	90	77.77	83.44
TKG with Eigen centrality (Abilhoa and Castro 2014)	W_1	80	88.88	84.21
	W_f	80	88.88	84.21
	$W_{1/f}$	80	88.88	84.21
TKG with TextRank centrality (Abilhoa and Castro 2014)	W_1	70	66.66	68.29
	W_f	70	77.77	73.68
	$W_{1/f}$	70	77.77	73.68
TKG with selectivity centrality (Abilhoa and Castro 2014)	W_c	80	77.77	78.86
KEGBA with degree (Nagarajan et al. 2016)	W_f	70	66.66	68.29
KEGBA with closeness (Nagarajan et al. 2016)	W_f	70	66.66	68.29
NE rank using TF–IDF (Bellaachia and Al-Dhelaan 2012)	W_{f2}	80	77.77	78.86
KCW	W_c	90	88.88	89.43

racy than using $D_{C(i)}$, SC(i) and $\text{Neigh}_{\text{Imp}}(i)$, in Donald Trump and IPhone5 data set.

(iv) For three data sets, the results using $D_{C(i)}$, SC(i), $\text{Neigh}_{\text{Imp}}(i)$ and; $F(i)$ and $L(i)$ and using all the parameters in KCW model are the same. However, KCW model using all the parameters produces more accuracy than using $D_{C(i)}$, SC(i), $\text{Neigh}_{\text{Imp}}(i)$ and; $F(i)$ and $L(i)$, in IPL and Iphone5 data set.

From the observations, it can be concluded that distance from the central node ($D_{C(i)}$) is the most important feature in the node weight assignment. Second, selectivity centrality (SC(i)) is found to be the second most significant feature

Table 11 Performances in IPhone5 data set using the crowd-sourced keywords set

Model	Edge-weighting mechanism	Pr in %	Re in %	F in %
TKG with closeness centrality (Abilhoa and Castro 2014)	W_1	60	75	66.67
	W_f	60	62.5	61.22
	$W_{1/f}$	80	75	77.41
TKG with eccentricity centrality (Abilhoa and Castro 2014)	W_1	80	75	77.41
	W_f	80	75	77.41
	$W_{1/f}$	80	75	77.41
TKG with Eigen centrality (Abilhoa and Castro 2014)	W_1	80	87.5	83.58
	W_f	80	87.5	83.58
	$W_{1/f}$	80	87.5	83.58
TKG with TextRank centrality (Abilhoa and Castro 2014)	W_1	70	75	72.41
	W_f	70	87.5	77.78
	$W_{1/f}$	70	87.5	77.78
TKG with selectivity centrality (Abilhoa and Castro 2014)	W_c	80	87.5	83.58
KEGBA with degree (Nagarajan et al. 2016)	W_f	70	62.5	66.08
KEGBA with closeness (Nagarajan et al. 2016)	W_f	70	62.5	66.08
NE rank using TF–IDF (Bellaachia and Al-Dhelaan 2012)	W_{f2}	80	75	77.41
KCW	W_c	80	100	88.88

while calculating the weights of the nodes. Term frequency seems to be important than neighbouring nodes ($\text{Neigh}_{\text{Imp}}(i)$) and position of nodes ($F(i)$, $L(i)$). However, neighbouring nodes ($\text{Neigh}_{\text{Imp}}(i)$) and position of nodes ($F(i)$, $L(i)$) seem to be equally significant in the weight assignment. From the observations of Table 5, it can be concluded that though all the features are significant for the node weight assignment, the distance from the central node ($D_{C(i)}$) and selectivity centrality ($\text{SC}(i)$) is the most important features in the proposed KCW model.

For the comparison of our proposed model, we have considered three baseline models as follows:

(i) Twitter keyword graph (TKG) (Abilhoa and Castro 2014): all the variants are considered.
(ii) Keyword extraction using graph-based approach (KEGBA) (Nagarajan et al. 2016).
(iii) NE rank using TF–IDF as node weighting scheme (Bellaachia and Al-Dhelaan 2012).

Three edge-weighting possibilities are considered for existing model (Abilhoa and Castro 2014): same weight assignment (W_1), weight as co-occurrence frequency (W_f) and weight as inverse co-occurrence frequency ($W_{1/f}$). Degree centrality (C_d), closeness centrality (C_c), and eccentricity centrality (C_e) are used to measure the relevance of nodes/terms. The existing model (Bellaachia and Al-Dhelaan 2012) uses the co-occurrence value between any two words within a specific window size as the weight of the edges (W_{f2}) between two nodes. The window size used is equal to 2. The proposed model uses Eq. (2) to find edge weight and Eq. (7) to find node weight.

The performances of TKG, KEGBA, and NE rank using TF–IDF and proposed KCW for Uri Attack, IPL, Harry Potter, Donald Trump, and IPhone5 data sets are shown in Tables 6, 7, 8, 9, and 10, respectively. The pre-processing described in Sect. 3.1 is done in exactly the same way for all the four methods to assure the fairness of the evaluation. The following observations are made from the experimental results when the number of retrieved keywords is 10. The proposed model provides the highest precision, recall, and F measure in Uri Attack data set. However, the Precision is equal to the Precision produced by KEGBA with degree centrality. The model also provides the highest precision, recall, and F measure in IPL data set. The proposed model provides the highest precision, recall, and F measure in Harry Potter data set; however, the Precision is equal to the precision produced by TKG with TextRank centrality and NE rank using TF–IDF model. For Donald Trump data set, the proposed KCW model provides better precision and F measure than TKG and KEGBA model; however, the recall is equal to the recall produced by TKG with TextRank and closeness centralities; KEGBA with degree centrality and NE rank using TFIDF. For IPhone5 data set, the model provides the highest precision, recall, and F measure. However, for IPhone5 data set, the precision is equal to the precision produced by TKG with eccentricity centrality and the recall is equal to the recall produced by TKG with Eigen centrality.

To show the relevance of the proposed work, the study is also conducted using a crowd-sourced data set IPhone5.

Table 12 Keywords extracted in IPL data set

Model	Edge-weighting mechanism	Extracted keywords
TKG with closeness centrality (Abilhoa and Castro 2014)	W_1	***'ipl' 'rpsvkkr'*** 'first' 'overs' ***'dhoni' 'rps' 'ipl2017'*** '1' **'msdhoni'** ***'scores'***
	W_f	***'ipl' 'rpsvkkr'*** 'first' 'will' ***'dhoni' 'rps' 'ipl2017'*** '1' **'msdhoni' 'uthappa'**
	$W_{1/f}$	***'ipl' 'rpsvkkr'*** 'first' 'will' ***'dhoni' 'rps' 'ipl2017'*** **'msdhoni' 'uthappa'** 'thanks'
TKG with eccentricity centrality (Abilhoa and Castro 2014)	W_1	***'ipl2017'*** 'commitment' 'rakhikicommentary' 'ind' 'mumbai' 'gulzar' 'consecutive' 'move' ***'kkrvrps'*** 'fantasyleague'
	W_f	**'msdhoni'** ***'rps'*** **'commentary'** ***'ipl2017'*** 'rakhikicommentary' 'ind' 'mumbai' 'gulzar' 'consecutive' 'move'
	$W_{1/f}$	**'msdhoni'** ***'rps' '*** **commentary'** ***'ipl2017'*** 'rakhikicommentary' 'ind' 'mumbai' 'gulzar' 'consecutive' '1'
TKG with Eigen centrality (Abilhoa and Castro 2014)	W_1	***'kkrvrps' 'kkr'*** **'stokes'** ***'dhoni'*** 'consecutive' ***'rps' 'ipl2017' 'match'*** 'commitment' 'today'
	W_f	***'kkrvrps'*** **'raina'** ***'dhoni' 'rps' 'ipl' 'match'*** ' finally' 'today' 'mumbai' 'gulzar'
	$W_{1/f}$	***'kkr'*** **'raina'** ***'dhoni' 'rps' 'ipl2017' 'match'*** 'finally' 'today' 'mumbai' 'gulzar'
TKG with TextRank centrality (Abilhoa and Castro 2014)	W_1	***'ipl' 'rpsvkkr'*** 'shivil' ***'kkr' 'dhoni' 'rps' 'ipl2017'*** 'level' ***'match'*** **'uthappa'**
	W_f	***'rpsvkkr' 'scores' 'kkr' 'dhoni' 'rps' 'ipl2017' 'match'*** 'updates' **'uthappa'** 'dive'
	$W_{1/f}$	***'rpsvkkr' 'scores' 'kkr' 'dhoni' 'rps' 'ipl2017' 'match'*** **'uthappa'** 'dive' 'rpsupergiants'
TKG with selectivity centrality (Abilhoa and Castro 2014)	W_c	***'rpsvkkr'*** 'rakhikicommentary' ***'kkr' 'dhoni'*** **'pune'** ***'rps' 'ipl2017' 'scores'*** 'will' 'fantasyleague'
KEGBA with degree (Nagarajan et al. 2016)	W_f	***'ipl'*** **'msdhoni'** ***'rpsvkkr' 'kkr' 'dhoni'*** 'will' ***'rps' 'ipl2017' 'scores'*** 'first'
KEGBA with closeness (Nagarajan et al. 2016)	W_f	***'ipl' 'rpsvkkr'*** 'first' 'overs' ***'dhoni' 'rps' 'ipl2017'*** '1' **'msdhoni' 'gambhir'**
NE rank using TF–IDF (Bellaachia and Al-Dhelaan 2012)	W_{f2}	***ipl' 'rpsvkkr'*** **'msdhoni'** ***'rps'*** 'mumbai' 'gulzar' 'consecutive' 'move' ***'match' 'scores'***
KCW	W_c	***'ipl' 'rpsvkkr' 'kkr'*** **'msdhoni'** ***'rps' 'ipl2017' 'match' 'kkrvrps'*** 'over' ***'scores'***

A survey is conducted using Google spreadsheets within the institution, where scholars and faculties are instructed to extract keywords for the data set based on the same norms and instructions as the three human extractors. Around 250 individuals responded. Using the keywords selected by around 250 individuals, instead of that selected by three human extractors, as shown in Table 4, Precision, Recall, and F measure are calculated for the IPhone5 data set using Eqs. (10)–(12), respectively, which is shown in Table 11. For the crowd-sourced Iphone5 data set, number of Inter_Relevant keywords is 8. From Tables 10 and 11, it can be seen that the results obtained using the keywords selected by the three human extractors are mostly better than that obtained using the crowd-sourced keywords. It is also be observed that out of 17 different methods, 12 methods extracted the same number of Relevant keywords while calculating the precision, and while calculating the recall, six different methods extracted the same number of Inter_Relevant keywords for the two different sets of keywords, i.e., one selected by 3 humans and the other selected by almost 250 individuals. This shows that the study of the proposed work can also be conducted using crowd-sourced user bases and thus establishes the relevance of the proposed work.

Table 12 shows the top ten extracted keywords for IPL data set according to their ranks, for the three existing methods used for comparison and the proposed model. The keywords highlighted in bold are the ones which are in at least one of the human lists. The keywords which are both bold and italic are the ones which are present in the intersection set of the three human lists. The keywords without any highlight are the ones which are misclassified. It is observed from Table 12 that keywords misclassified by the different methods are mostly irrelevant for the topic. It is also observed that KCW model succeeds to extract significantly important keywords for a particular topic in comparison with other existing methods.

From Tables 6, 7, 8, 9, 10, 11 and 12, it can be observed that the proposed model shows significant improvement in comparison with other existing methods. This significant improvement is a contribution of different factors. KCW model uses an effective pre-processing phase along with the use of AOF which eliminates the irrelevant tokens/terms. TKG originally performs simple stopwords removal and tokenization as the pre-processing step. KEGBA originally does not use any pre-processing step. KCW model considers five significantly important features of a node for determining the node weight. TKG does not combine different measures to find weight of the nodes. KCW considers both frequency and relation between the nodes for calculating the weight of the edges in a balanced and effective manner using both frequency and co-occurrence frequency of the nodes. TKG finds weight of the edges by the same weight edges/co-occurrence frequency/inverse co-occurrence frequency. TKG uses closeness and eccentricity centralities to determine node weight and degree centrality as the tie breaker; however, closeness and eccentricity centralities do not work well for disconnected graphs. Finally, two well established methods are used by KCW for determining the ranks of the keywords, i.e., degree centrality and NE rank. KCW uses an improvised version of NE rank for keyword extraction, i.e., the node weight used in NE rank is determined by the summation of different significantly important features of nodes. Even though TKG is implemented with different centrality measures, it does not perform better than KCW, because originally TKG uses simple pre-processing and edge-weighting techniques, and does not combine different measures to find weight of the nodes. KEGBA represents graph using simple co-occurrence relations between words to find weight of the edges. Finally, degree and closeness centralities are separately used to find weight of the nodes. Thus, noises are not removed and edge weights are not properly captured. Centrality measures and others are not combined to overcome each other's disadvantages to find weight of the nodes. NE rank using TF–IDF solely depends on frequency of words and disregards the relationship between words for determining the node weights. Sometimes, an important word appears less number of times which is not taken care by TF–IDF. TF–IDF does not perform better keyword extraction in twitter data due to the diversity and informality in twitter contents.

6 Conclusions

Keyword extraction is one of the most important tasks in analyzing textual data of micro-blogging sites like Twitter. This paper proposes an efficient keyword extraction model called KCW which consists of four phases: pre-processing, textual graph representation, node weight assignment, and keyword extraction. Pre-processing phase is executed to remove unwanted noise, stop words, and perform tokenization. Textual graph representation involves node assignment and establishment of edges between these nodes. A new node weight assignment scheme is introduced for the NE-rank approach. Node weight assignment phase determines node weight based on its position, frequency, centrality, distance from the central node, and strength of the neighbours. Finally, the most important keywords are extracted using NE-rank centrality with degree as tie breaker. Experimental results show that each of the five different features used for the node weighing scheme is significantly important for the overall accuracy of the proposed model. To show the superiority of the proposed model, it is compared with variants of two existing graph-based models and one existing non-graph model. It is observed from the experimental results that the performance of KCW model is far better than others. It is also observed that the most important keywords extracted by KCW model are certainly important for a particular topic in comparison with other existing methods.

Being a generalized model, the proposed KCW model can be used in sentiment analysis for keyword extraction. This domain pursues great potential of quality research in the near future. Different angles of this domain can be explored and can be augmented with other significant algorithms. New edge weighing and node weighing methods can be proposed and they can be used in NE-rank centrality or other centrality measures to find rank of keywords. Different cascaded or individual approaches for other centrality measures can also be adopted in the future to get better results. Semantic methods for keyword extraction can be explored along with the proposed model to get better results.

References

Abilhoa WD, de Castro LN (2014) A keyword extraction method from twitter messages represented as graphs. Appl Math Comput 240:308–325

Beliga S, Mestrovic A, Martincic-Ipsic S (2015) An overview of graph-based keyword extraction methods and approaches. JIOS 39(1):1–20

Bellaachia A, Al-Dhelaan M (2012) NE-rank: a novel graph-based key phrase extraction in twitter. In: International Joint conferences on web intelligence and intelligent agent technology, vol 1. IEEE, WIC, ACM, pp 372–379

Berry MW, Kogan J (2010) Text mining: applications and theory. Wiley, West Sussex

Bordag S, Heyer G, Quasthoff U (2003) Small worlds of concepts and other principles of semantic search. In: Bhme T, Heyer G, Unger H (eds) IICS, 2003, lecture notes in computer science, vol 2877, pp 10–19

Boudin F (2013) A comparison of centrality measures for graph-based keyphrase extraction. In: International joint conference on natural language processing (IJCNLP), pp 834–838

Bougouin A, Boudin F, Daille B (2013) TopicRank: graph-based topic ranking for keyphrase extraction. In: International joint conference on natural language processing (IJCNLP), pp 543–551

Chen P, Lin S (2010) Automatic keyword prediction using Google similarity distance. Expert Syst Appl 37(3):1928–1938

Cohen-Kerner H (2003) Automatic extraction of keyword from abstracts. In: Automatic extraction of keyword from abstracts, lecture notes in computer science, vol 2773, pp 843–849

Ediger D, Jiang K, Riedy J, Bader DA, Corley C, Farber R, Reynolds WN (2010) Massive social network analysis: mining twitter for social good. In: 39th international conference on parallel processing. IEEE, pp 583–593

Goutte C, Gaussier E, Probabilistic A (2005) Interpretation of precision, recall and F-score, with implication for evaluation. In: Losada DE, Fernández-Luna JM (eds) Advances in information retrieval, ECIR 2005, lecture notes in computer science, vol 3408. Springer, Berlin

Grineva M, Grinev M, Lizorkin D (2009) Extracting key terms from noisy and multi-theme documents. In: 18th international conference on World Wide Web, NY, USA, pp 661–670

Hemalatha I, Saradhi Varma GP, Govardhan A (2013) Sentiment analysis tool using machine learning algorithms. Int J Emerg Trends Technol Comput Sci (IJETTCS) 2(2):105–109

Hotho A, Nürnberger A, Paab G (2005) A brief survey of text mining. LDV Forum GLDV J Comput Linguist Lang Technol 20(1):19–62

Hulth A (2003) Improved automatic keyword extraction given more linguistic knowledge. In: Conference on Empirical methods in natural language processing, pp 216–223

Jin W, Srihari R (2007) Graph-based text representation and knowledge discovery. In: Proceedings of the SAC conference, pp 807–811

Khan TM, Yukun, Kim J (2016) Term ranker: a graph based re-ranking approach. In: FLAIRS conference (AAAI), pp. 310–315

Kwon K, Choi CH, Lee J (2015) A graph based representative keywords extraction model from news articles. In: International conference on big data applications and services. ACM, pp 30–36

Lahiri S, Choudhury SR, Caragea C (2014) Keyword and keyphrase extraction using centrality measures on collocation networks. arXiv:1401.6571 [cs.CL]

Litvak M, Last M, Aizenman H, Gobits H, Kandel A (2011) DegExt—a language-independent graph-based keyphrase extractor, In: Mugellini E, Szczepaniak PS, Pettenati MC, Sokhn M (eds) Advances in intelligent web mastering—3. Advances in intelligent and soft computing, vol 86. Springer, Berlin, pp 121–130

Medelyan O, Witten IH (2006) Thesaurus based automatic keyphrase indexing. In: 6th ACM/IEEE-CS joint conference on digital libraries, pp 296–297

Nagarajan R, Nair DSAH, Aruna DrP, Puviarasan N (2016) Keyword extraction using graph based approach. Int J Adv Res Comput Sci Softw Eng 6(10):25–29

Nguyen TD, Kan MY (2007) Keyphrase extraction in scientific publications. In: 10th international conference on Asian digital libraries: looking back 10 years and forging new frontiers, pp 317–326

Ravinuthala MKVV, Reddy ChS, Graph TT (2016) A text representation technique for keyword weighting in extractive summarization system. Int J Inf Eng Electron Bus (MECS) 8(4):18–25

Rousseau F, Vazigiannis M (2013) Graph-of-word and TW-IDF: new approach to ad hoc IR. In: Proceedings of the 22nd ACM international conference on conference on information and knowledge management 2013, pp 59–68

Savita DB, Gore PD (2016) Sentiment analysis on twitter data using support vector machine. Int J Comput Sci Trends Technol (IJCST) 4(3):365–370

Sonawane SS, Dr PA, Kulkarni (2014) Graph based representation and analysis of text document: a survey of techniques. Int J Comput Appl 96(19):1–8

Song HJ, Go J, B.Park S, Park SY, Kim KY (2017) A just-in-time keyword extraction from meeting transcripts using temporal and participant information. J Intell Inf Syst 48(1):117–140

Wang Z, Feng Y, Li F (2016) The improvements of text rank for domain-specific key phrase extraction. Int J Simul Syst Sci Technol 17(20):11.1–11.5

Witten IH, Paynter GW, Frank E, Gutwin C, Nevill-Manning CG (1999) KEA: practical automatic keyphrase extraction. In: Fourth ACM conference on digital libraries, pp 254–255

Wu J, Xuan Z, Pan D (2011) Enhancing text representation for classification tasks with semantic graph structures. Int J Innov Comput Inf Control 7(5B):2689–2698

Zahang C, Wang H, Liu Y, Wu D, Liao Y, Wang B (2008) Automatic keyword extraction from documents using conditional random fields. J CIS 4(3):1169–1180

Zhang K, Xu H, Tang J, Li J (2006) Keyword extraction using support vector machine. In: 7th international conference on advances in web-age information management, pp 85–96

Zhao WX, Jiang J, He J, Song Y, Achananuparp P, Li EP, Li X (2011) Topical keyphrase extraction from twitter. In: Proceedings of the 49th annual meeting of the ACL, Portland, Oregon, June 19–24. ACL, pp 379–388