Semi Supervised Graph Based Keyword Extraction Using Lexical Chains and Centrality Measures

Ayush Aggarwal[1], Chhavi Sharma[1], Minni Jain[1], Amita Jain[2]

[1] Delhi Technological University, Delhi, India

[2] Ambedkar Institute of Advanced Communication Technologies and Research, Delhi, India

{aggarwal96ayush, sharma.chhavi96}@gmail.com
minnijain@dtu.ac.in, amita_jain_17@yahoo.com

Abstract. This paper presents keyword extraction using lexical chains and graph centrality measures, derived from the semantic similarity of the words by analysis of the graphical network created using WordNet. The hypothesis is presented using a small-world approach where every paragraph in a document is constrained to a local point, while the document in all is centered on a global concept. Creating lexical chains for each paragraph and combining the best via scoring methods and graph based algorithms, we present parallels to baseline system to extract the keywords from the document.

Keywords. Graph centrality, keyword extraction, lexical chains, semantic similarity, small world approach, WordNet.

1 Introduction

Keyword extraction is used to describe the basic subject of a document by identification of terms that best describe it [2, 10]. This research focuses on an algorithm for automatic extraction of these keywords and thus provides with the document's summary. Unlike text summarization, which provides certain important extracts of the document, keyword extraction also extends the functionality by helping in the indexing of document for search engines or text categorization by providing singular words that best represent the document. In the present day scenario, where there is a lot of data available, it becomes seemingly impossible to sift through the entire collection manually, with the aim of finding relevant information whereas keywords allow us to identify the main point of the author.

Unfortunately, in many cases we do not have labeled text pieces with their keywords and thus keyword extraction helps automate the task. In this paper, we focus our work on *keywords* rather than *key-phrases* as key phrases are composed of more than one word while we are extracting only the keywords.

Existing Automatic keyword extraction tools provide generally the words with highest occurring frequency. Moreover they do not correlate with various synsets (groups of synonymous English words) of the same word, thus resulting in multiple keywords of the same root-word or failing in classifying a word as a keyword due to its occurrence multiple times, but as different synsets of the same root-word [16]. These problems can be removed by using WordNet corpus, where there is an existing database of synsets, along with various relations to other words such as hyponymy, hyperonymy and hypernymy.

Lexical chains are sets of semantically related words [17, 21]. The keywords of the document would be frequently occurring as synsets and hence, would be contained in the lexical chains of the paragraph. A chain can be representative of a small portion portion of the document. Thus various chains are calculated across the document and are scored to identify their relevancy. The chains with higher scores are considered to be representatives of the document.

The similarity between keyword extraction and text summarization roots from the fact that both the NLP techniques take parts of documents to represent the summary. Lexical chains have been shown to be used for text summarization [1, 4] in the past. Recent researches have started to explore the use of lexical chains to extract keywords from a document. Basing the hypothesis on these, we propose an algorithm for keyword extraction using WordNet ontology.

2 Related Work

Various methods of keyword extraction have been tried extensively, especially for the purposes of indexing and improving web searches of the documents. Keyword extraction can be classified into supervised, semi-supervised and unsupervised approaches.

Supervised methods work with a hand annotated data-set of documents and keywords and use domain specific knowledge to classify them in two classes (or binary classification): *keyword or not*. Two standard supervised learning systems in the field of keyword extraction are KEA [23] and GenEX [18]. Both of these approaches are based upon the frequency as well as the location of term in a document while classifying. While GenEx uses a C4.5 Decision tree to classify, KEA uses Naive Bayes for learning and classification. KEA has been further improved over the years using statistical association between key phrases [19] or by using semantic information from a domain specific thesaurus as in KEA [14].

The issues with supervised keyword extraction methods are a need for hand annotated keywords for training purposes and a bias towards the domain specific knowledge base they are trained upon. That is why we now aim at semi-supervised approaches. Graph Based approaches provide a sophisticated way to extract keywords using the structure of the graph created using source statistics.

KeyGraph is an algorithm for automatic indexing by a co-occurrence graph constructed from metaphors [9]. It is based upon segmenting of a graph, representing co-occurrences of the document as clusters. Each cluster represents basically the Small World structure of the document.While the small world structure helped to model and connect different meanings of the document into a global meaning, it was a content sensitive and domain independent algorithm.

SemanticRank uses a graph based technique exploiting the semantic relatedness of the document using knowledge-based measures of WordNet and Wikipedia and graph centrality measures of PageRank [15] and Hits [7]. The success of the above Graph based approaches using WordNet as their knowledge base by performing at par with the supervised learning standard systems led us to incorporate the same in our work.

The motivation behind using lexical chains for Keyword Extraction is behind the fact that lexical chains have been used for text summarization [1] and that there is a great sense of similarity between text summarization and keyword extraction. Prior work on using lexical chains for keyword extraction [3] had shown good preliminary results, which shows that this can be further looked into. But this method used a supervised learning approach to create lexical chains and hence does not solve the problem of non-availability of hand tagged data-set and is highly biased to multiple occurrence of words belonging to same synset or having the same root.

In the proposed work, we combine the graph based semi supervised approach with the benefits of using lexical chains to propose an algorithm that performs much better than both the approaches if handled singularly. It not only makes the algorithm domain independent and free of needing prior knowledge of the domain to extract information but the small world approach of treating each paragraph as a building base of the whole document helps to distribute the global meaning in the proposed algorithm thus reducing bias of the system.

3 The Lexicon

We use a database of English words and their relations: WordNet[1] [11]. It uses the concept of "cognitive synonyms" or synsets, and interlinks them on the basis of lexical relations. It is a free linguistic tool which acts as a dictionary or thesaurus, providing the user with synonym sets, word definitions and their examples.

3.1 Structure

WordNet connects words using various relationships. The main relation among these is *synonymy* on the basis of which we derive various synsets : the basic unit of WordNet relational identity [11]. Each synset, thus derived from the relation of synonym, in turn, is linked to various other synsets by the relations described below.

3.2 Relations

One of the most important relations among the semantically related synsets is the relation of super-subordination called *hyponymy*. While hyponymy connects the broader sense of the word with a specific or a part of the sense, *hypernymy* does the opposite and connects the synsets using a is-a relation [5]. The following Figure 1 shows a semantic network constructed using the relations of *hyponymy* and *hypernymy* of the word dog.

4 Lexical Chains

A lexical chain can be defined as a sequence of related words that denote the semantic context of the piece of text [12]. Hence, determining these chains helps to identify the main topics of a document. They have been previously explored for information retrieval and related areas [22]. In the proposed work, we will be using WordNet to build lexical chain as done by Stairmand [17] as lexical chains require the use of an ontology or a database which has predefined chains of semantically similar words.

Generally the procedure for creating a lexical chain is as follows as postulated by Stairmand [17]:

[1]WordNet is available from http://wordnet.princeton.edu

1. Select a set of candidate words.
2. Determine a suitable chain, calculated from semantic relatedness among members of the chain for each selected word.
3. If a chain exists, add the word and update the chain else create a new chain to fit the word.

The semantic similarity of the proposed algorithm is based on a lexical similarity measure created by Wu and Palmer [24]. The measure examines which lexical chain provides the maximum relevance to the given word and whether it passes a threshold value.

4.1 Performance Function

After creating a lexical chain, we need to score them to determine the best lexical chains to determine the words to be fed into our graph based system. The lexical chains were scored using various methods, and we selected the one with better precision as shown in the next sections.

Method I:
This method builds upon the methods described by LexSum[**?**]. In this scoring method, the score of a chain is the summation of score of all the members in the chain divided by the numbers of members. The score of each member in the chain is a combination of frequency of the word in the paragraph and its similarity index with the members of the chain:

$$chain\_score(C) = \frac{\sum rel_i(w) * freq(w)}{length(C)}, \quad (1)$$

where $rel_i(w)$ denotes the similarity index of member w with i which is the representative of that chain: the member with maximum frequency in that chain. Now the threshold value of a paragraph is calculated:

$$threshold(P) = \mu(c) * \sigma(c) : \forall c \in C, \quad (2)$$

and for every chain whose score is greater than the aforementioned threshold, we add all the member's representative words with similarity score greater than threshold to the final list of

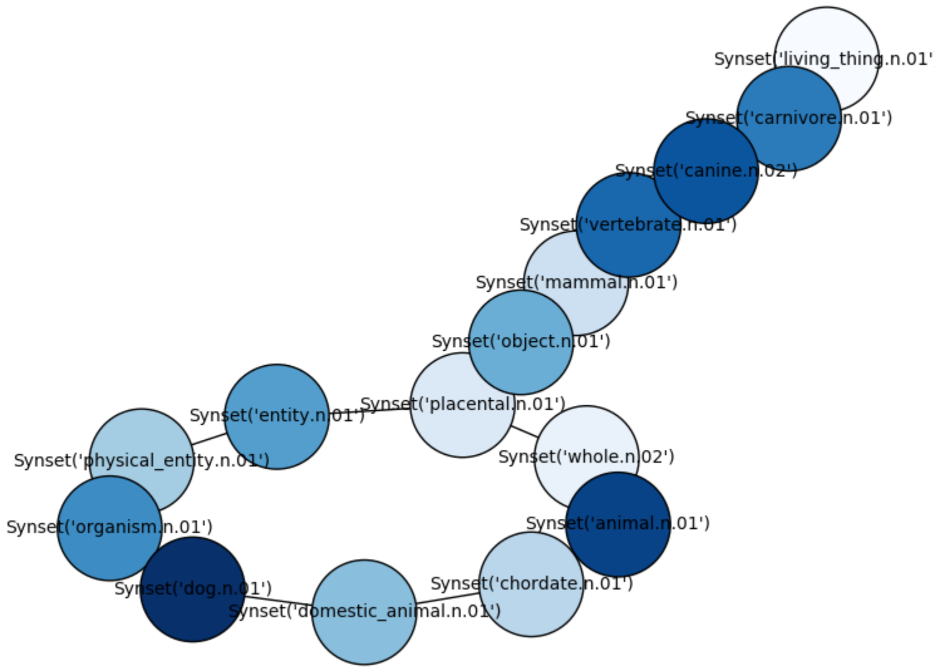


Fig. 1. Graph extracted from WordNet using *synsets* of the word *dog*

words on which we apply the proposed graph based algorithm.

Method II:
We propose another method for scoring of lexical chain. This method of scoring lexical chain is dependent not on the frequency of the word in the paragraph, as common words with broad meaning appear more, but on the semantic distance between the words of each chain. For each member of the lexical chain, we maintain a list of hypernyms and hyponyms, and thus find the semantic distance of a lexical chain by iterating through the lexical chain and scoring a hypernym/hyponym match with *0.5* and an identical/ synonym match with *1*.

Let the set of lexical chains be denoted by $L = \{L_1, L_2, ..., L_n\}$ and for every lexical chain we say $L_i = \{w_1, w_2, ..., w_n\}$ be the set of connected words where every word will has a score of c_j, then:

$$chain\_score(L_i) = \frac{\sum c_j}{length(L_i)} \forall c_j \in L_i. \quad (3)$$

We calculate the threshold of all the chains of the paragraph as:

$$threshold(P) = \mu(L_i) + 2 * \sigma(L_i) : \forall L_i \in L, \quad (4)$$

as taken by Braizaley [1] and for every chain whose score crosses the threshold , we select the representative of the said chain where representative refers to the member with maximum member score in the chain.

5 Proposed Scheme

The algorithm that we propose incorporates the features from a lexical chain based algorithm, which makes the algorithm identify the emphasized sense of the document by creating lexical chains for every paragraph and using the best chains to feed it to a graph based WSD algorithm as suggested by Navigali and Lapata [13]. The advantage of using the graph approach on top of a lexical chain based segregation is that it removes the shortcomings of the supervised learning algorithm and makes it domain independent and not in requirement of a pre-fed training data to adapt. The use of WordNet assures us that the algorithm can be extended to any use-case. The keyword extraction algorithm uses a small world approach to ensure the central meaning of the document is identified.

The algorithm parses every paragraph of the document and first extracts all the nouns (proper nouns and compound nouns) from the paragraph. Now using these lists of nouns, lexical chains for a particular paragraph are built as follows:

— Calculate the similarity of the current word with all the previously created lexical chains.

— If the maximum similarity value is greater than a threshold value, then add this word to the lexical chain else create a new lexical chain with the word as its first entry.

After creating the lexical chains for each paragraph they are scored and then strength, average score and standard deviation for these lexical chains are calculated separately. Using these values, we calculate the cut-off score for lexical chains in a particular paragraph and determine the *best chains* i.e. all the lexical chains which have a score greater than this cut-off score.

We determine these best chains for all paragraphs and add all the members of these chains to a word list which will be the probable list of keywords. We feed this word list to the graph algorithm that finds semantic similarity between members of different lexical chains.

The algorithm to construct a graph is built upon the method used in word sense disambiguation [13]. The method has been proposed to be used in more language processing tasks [6] and the idea behind this algorithm is to find most important node using network analysis. We build a graph $G = (V, E)$ using the word senses as the nodes and the semantic relation between them as the un-directed edges. We then select a sense u ϵ senses and execute the following algorithm on it:

— Generate sets of two relations, hyponymy and hypernymy u.

— Perform a Depth First Search for all the words v in those sets. Every time we encounter a node belong to destination along the path, we plot it on the graph.

The algorithm does a DFS from the last vertex processed and uses a recursive approach to explore the un-visited vertices. It involves backtracking to a previous state if all the edges connecting a node have already been visited before. This process continues till we reach a state where no further discovery can be made. Centrality measures help to calculate the degree of relevance of a vertex v in a graph G. It also calculates the influence of the vertex in its network. These measures are pivotal for the algorithm as they analyze the connectivity of the probable keywords with all the other words from the set to identify the most central word as the extracted keyword from the document.

5.1 Proposed Algorithm

Let a document D be a set of paragraphs $P = \{P_1, P_2, ..., P_n\}$, then for every graph $P_i \in D$:

Step I: Extract all nouns using POS and create lexical chains from the set of nouns n.

Step II: To create a lexical chain, pick a word from the set of nouns and calculate wup_similarity with the existing chains. Add it to one of the existing chains or start a new chain.

Step III: For all the lexical chains $L = \{L_1, L_2, ..., L_k\}$, calculate the chain score by using equation 1 OR 3 and create a word list W which are a set of potential keywords.

Use the word list W to create a semantic network using hyponymy, hypernymy and synonymy relation as follows:

Step IV: For every $w_i \in W$, add the word to the graph G.

Step V: For every$w_i \in W$, calculate the hyponyms and hypernyms and synonyms and do a DFS with

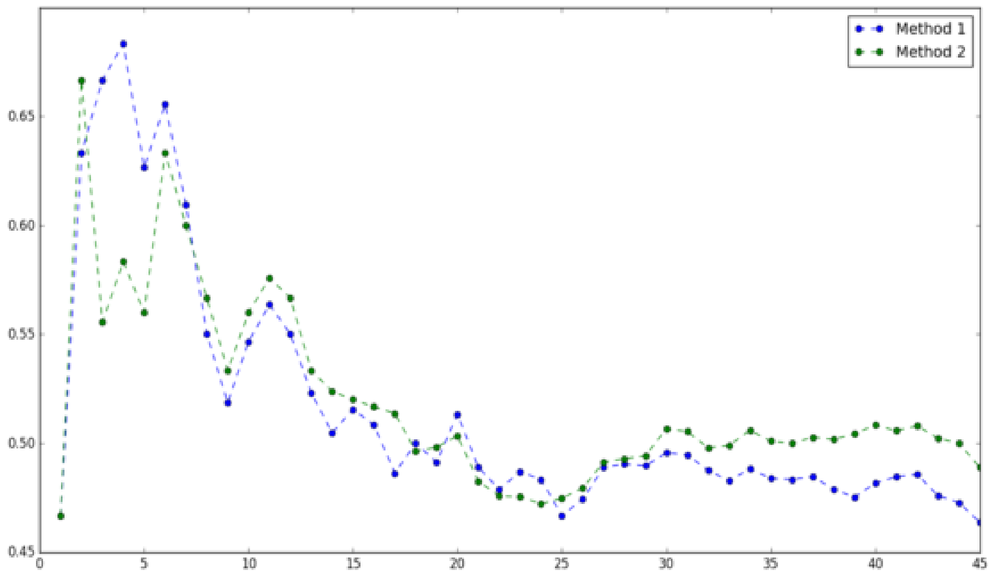


Fig. 2. Scoring Method 1 and Method 2

all the discovered words. If a match is found, a relationship is said to exist between two words.

Step VI: Explore the edges from most recently discovered word w_j till the point there are no new edges to be discovered.

Step VII: After the words are processed, analyze the network using the following centrality methods. If w_i, w_j denotes the nodes of a graph and E the set of edges and W the set of nodes then:

$$Degree(w_i) = |(w_i, w_j) \in E : w_i, w_j \in W|, \quad (5)$$

$$Betweenness(w_i) = \sum \frac{\sigma_{i,j}(w_i)}{\sigma_{i,j}}, \quad (6)$$

$$PageRank(w_i) = \frac{1-k}{|W|} + k \sum_{w_i, w_j \in E} \frac{PR(w_j)}{outdegree(w_j)}, \quad (7)$$

$$Closeness(w_i) = \frac{\sum_{w_j \in W : w_i \neq w_j} \frac{1}{d(x,y)}}{|S| - 1}, \quad (8)$$

$$HITS_A(w_i) = \sum_{w_j \in In(w_i)} HITS_H(w_j), \quad (9a)$$

$$HITS_H(w_i) = \sum_{w_j \in Out(w_i)} HITS_A(w_j), \quad (9b)$$

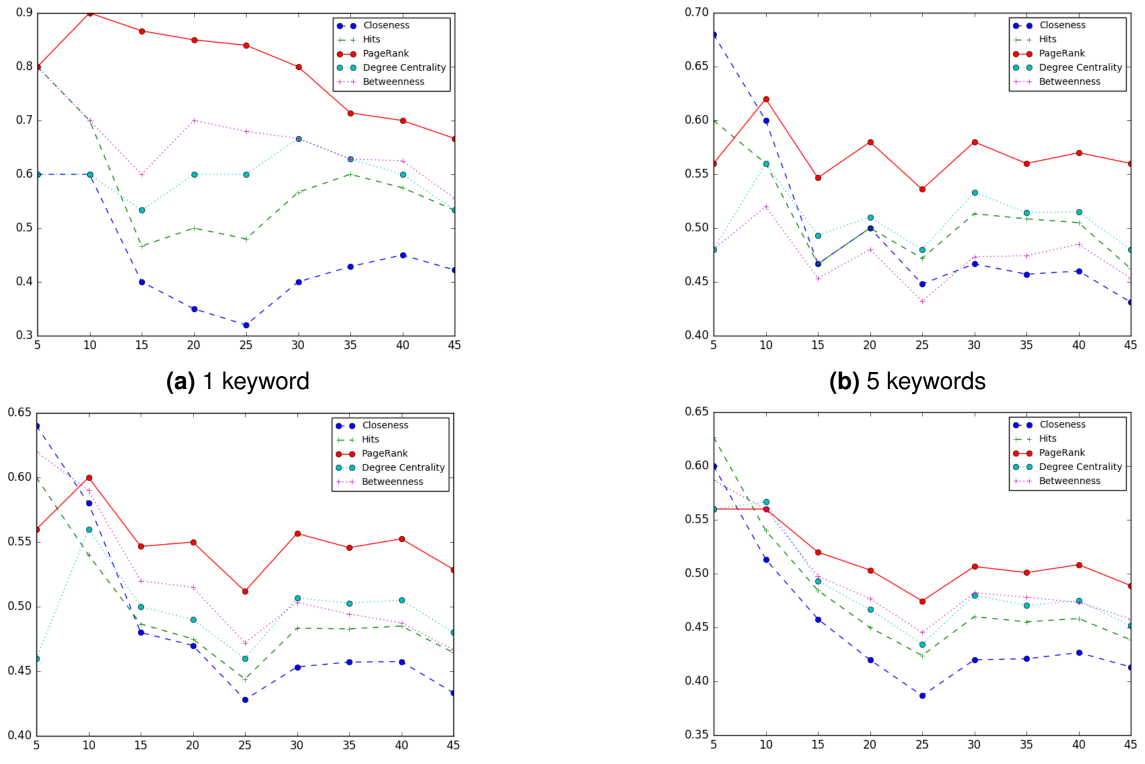
and evaluate the top keywords from every centrality to propose result.

6 Simulation Results

We present the research as an improvement for already existing work by Ercan and Cicekli (2007) [3]. Since it was a supervised learning algorithm, we use another dataset and their baseline results to present our comparison. We use 500N-KPCrowd [8] for testing the hypothesis. The data-set is a corpora of news articles from 10 different topics with hand annotated keywords associated with every news report. We use 45 articles to run the algorithm and aggregate results to present a direct comparison with the baseline results. It also has a list of hand annotated list of keywords by 20 Amazon workers.

We also use two different scoring methods and present a parallel between the two to consider for our algorithm as proposed above. From following figure 2 we notice that as the number of articles in the data-set increases, the accuracy of Method 2 supersedes that of the first method.

This is in accordance with our hypothesis that using small world approach thus makes the algorithm less biased towards word with more frequency in the text. This method also provides better results because it also takes into



(a) 1 keyword **(b)** 5 keywords **(c)** 10 keywords **(d)** 15 keywords

Fig. 3. Comparing precision score for different centrality measures for varying amount of keywords extracted

consideration the strength of the relation between two words and scores it accordingly.

For comparison with baseline results, we run our algorithm to extract various number of keywords on the same documents and compare the precision score thus achieved for various centrality measures. The results are thus plotted on the graph and we compare the performance of received results.

As we depict in the figure 3, PageRank centrality method surpasses other centrality measures as compared to the other centrality methods. This is in line with previous works [20] which also say that PageRank out of all other centrality measures can be be used to extract keywords from graph based network.

Table 1. Table for results observed

.

	Baseline Results	**Our Algorithm**
1 keyword	64.0%	66.7%
5 keywords	45.0%	56.0%
10 keywords	30.0 %	52.8%
15 keywords	26.0%	48.8%

We also compare our results with the baseline results as proposed by Ercan and observe that the algorithm surpasses the baseline results with proving to be a better improvement for extracting higher number of keywords. This is depicted in the Table 1.

Using graph based centrality method after using lexical chains improves the already existing baseline systems with considerate improvement when more keywords are being extracted for the document as proposed.

7 Conclusion and Future Work

We developed and discussed a graph based method, which supersedes the accuracy of pre-existing supervised learning algorithms. By incorporating the advantages of a network analysis approach using centrality methods and lexical chains, we were able to develop an algorithm, which is extendable to any domain and is less biased towards the type or frequency of word used in the document. However, this algorithm currently revolves around extraction of keywords from the document. It fails, when there is a set of words or *key-phrases*. Hence, we would like to expand to extraction of key phrases from a document.

The algorithm is unable to include words, which are not defined in the corpus and for this we plan to look into further knowledge bases such as Wikipedia.

References

1. **Barzilay, R. & Elhadad, M. (1997).** Using lexical chains for text summarization. *In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pp. 10–17.
2. **Beliga, S., Meštrović, A., & Martincic-Ipsic, S. (2015).** An overview of graph-based keyword extraction methods and approaches. *Journal of Information and Organizational Sciences*, Vol. 39, pp. 1–20.
3. **Ercan, G. & Cicekli, I. (2007).** Using lexical chains for keyword extraction. *Inf. Process. Manage.*, Vol. 43, No. 6, pp. 1705–1714.
4. **Erkan, G. & Radev, D. R. (2004).** Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, Vol. 22, No. 1, pp. 457–479.
5. **Franzoni, V., Li, Y., Leung, C. H. C., & Milani, A. (2017).** Semantic evolutionary concept distances for effective information retrieval in query expansion. *CoRR*, Vol. abs/1701.05311.
6. **Jain, A. & Lobiyal, D. K. (2015).** Fuzzy hindi wordnet and word sense disambiguation using fuzzy graph connectivity measures. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, Vol. 15, No. 2, pp. 8:1–8:31.
7. **Kleinberg, J. M. (1999).** Authoritative sources in a hyperlinked environment. *J. ACM*, Vol. 46, No. 5, pp. 604–632.
8. **Marujo, L., Gershman, A., Carbonell, J. G., Frederking, R. E., & Neto, J. P. (2013).** Supervised topical key phrase extraction of news stories using crowdsourcing, light filtering and co-reference normalization. *CoRR*, Vol. abs/1306.4886.
9. **Medelyan, O. & Witten, I. H. (2006).** Thesaurus based automatic keyphrase indexing. *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '06, ACM, New York, NY, USA, pp. 296–297.
10. **Mihalcea, R. & Tarau, P. (2004).** Textrank: Bringing order into text. *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.
11. **Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. (1990).** WordNet: An on-line lexical database. *International Journal of Lexicography*, Vol. 3, pp. 235–244.
12. **Morris, J. & Hirst, G. (1991).** Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Comput. Linguist.*, Vol. 17, No. 1, pp. 21–48.
13. **Navigli, R. & Lapata, M. (2010).** An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 32, No. 4, pp. 678–692.
14. **Ohsawa, Y., Benson, N. E., & Yachida, M. (1998).** Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. *Proceedings of the Advances in Digital Libraries Conference*, ADL '98, IEEE Computer Society, Washington, DC, USA, pp. 12–.
15. **Page, L., Brin, S., Motwani, R., & Winograd, T. (1998).** *The PageRank Citation Ranking: Bringing Order to the Web*.
16. **Ramos, J. (2003).** *Using TF-IDF to Determine Word Relevance in Document Queries*.
17. **Stairmand, M. & Engineering, L. (1996).** *A computational analysis of lexical cohesion with applications in information retrieval*. UMIST.

18. **Turney, P. D. (2000).** Learning algorithms for keyphrase extraction. *Inf. Retr.*, Vol. 2, No. 4, pp. 303–336.

19. **Turney, P. D. (2003).** Coherent keyphrase extraction via web mining. *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 434–439.

20. **Wang, J., Liu, J., & Wang, C. (2007).** Keyword extraction based on pagerank. **Zhou, Z.-H., Li, H., & Yang, Q.**, editors, *Advances in Knowledge Discovery and Data Mining*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 857–864.

21. **Wei, T., Lu, Y., Chang, H., Zhou, Q., & Bao, X. (2015).** A semantic approach for text clustering using wordnet and lexical chains. *Expert Syst. Appl.*, Vol. 42, No. 4, pp. 2264–2275.

22. **Wei, T., Lu, Y., Chang, H., Zhou, Q., & Bao, X. (2015).** A semantic approach for text clustering using wordnet and lexical chains. *Expert Syst. Appl.*, Vol. 42, No. 4, pp. 2264–2275.

23. **Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (1999).** Kea: Practical automatic keyphrase extraction. *Proceedings of the Fourth ACM Conference on Digital Libraries*, DL '99, ACM, New York, NY, USA, pp. 254–255.

24. **Wu, Z. & Palmer, M. (1994).** Verb semantics and lexical selection. *CoRR*, Vol. abs/cmp-lg/9406033.

Article received on 19/12/2017; accepted on 15/02/2018. Corresponding author is Ayush Aggarwal.