2015 11th International Conference on Semantics, Knowledge and Grids

Ranking Sentences in Scientific Literatures

Jiao Tian[#*1], Mengyun Cao[#*2], Jin Liu[*$3], Xiaoping Sun[*4], Zhuge Hai[*5]

[#]*University of Chinese Academy of Sciences*
Beijing, China
[1]tianjiao@kg.ict.ac.cn
[2]caomengyun@kg.ict.ac.cn
[*]*Knowledge Grid Group*
** #CAS Key Lab of Intelligent Information Processing*
Institute of Computing Technology
Chinese Academy of Sciences
Beijing, China
[$]*Wuhan University*
[3]jinliu@wuhan.edu.cn
[4]sunxiaoping@ict.ac.cn
[5]zhuge@ict.ac.cn

***Abstract*—Sentence ranking is one of the most important research issues in text analysis. It can be used in extracted textual summarization and information retrieval. Graph-based methods are a common way of ranking and extracting sentences. In graph based methods, sentences are nodes of graph and edges are built based on the sentence similarities or on sentence co-occurrence relationship. PageRank style algorithms can be applied to get sentence ranks. In this paper, we focus on how to rank sentences in a single scientific paper. A scientific literature has more structural information than general texts and this structural information has not been fully explored yet in graph based ranking models. We investigated several different methods that used structural information such as paragraph and section information to construct a heterogeneous graph for ranking sentences. We conducted experiments on these methods to compare the results on sentence ranking. It shows that structural information can help identify more representative sentences.**

Graph-based method; sentence ranking; scientific literatures; structural information

I. INTRODUCTION

Ranking sentences in a text is to calculate a weight factor for each sentence, sort them in descent order and obtain top-k sentences for further text processing, such as extraction-based text summarization or query answer. For example, in [22] and [23], ranking sentences can help perform the document summarization and query answer tasks. In general, sentence ranking consists of query-based ranking and general purpose ranking. No matter what purpose it is, the key point is how to measure the importance of sentences to best fit our purpose.

There are two major solutions to measure the importance of sentences in sentence ranking. One is supervised-based, which uses annotated text to train a model to predict the weight of each sentence. The features can be TF-IDF value or sentence position etc. Another is unsupervised-based, which does not need any training text. Currently, most studies are unsupervised which uses heuristic information like sentence position, cue phrases etc. Among them, graph-based methods are commonly used [1] [3][6][7][8][10].

Many graph-based sentence ranking methods have used the information of sentence position, cue phrases, the similarity of sentences or the word co-occurrence. Few considered structural information contained in the text. For example, in scientific literatures, a paper is often organized in a certain structure consisting of different sections and paragraphs. This structural information can reflect the structural semantics in some extent.

In this paper, we explored how to use the structural information of paper for ranking sentences and proposed a model which constructs a heterogeneous graph consisting of sentences, paragraphs and sections for ranking sentences in one scientific paper. Ranking sentences in one paper will provide useful information in paper summarization or query-answer for one specific paper.

An iterative graph based ranking algorithm is designed on heterogeneous graph to rank sentences. We also designed three variants of the ranking algorithms which uses different combination of available structural information for ranking sentences. Two experimental tasks are designed to test the effectiveness of these ranking algorithms. The results indicate that sentence information is more effective than paragraph and section information and the paragraph information can help improve the performance. We also compared the proposed algorithms with three other models: a TF-IDF based model, a graph-based model which uses sentences as vertices and a graph-based model which uses word as vertices. All the results indicate that the structural information can improve the precision of sentence ranking.

In section 1, a brief introduction about the purpose and background of this paper is given. Section 2 presents related work about graph-based model and ranking algorithms. Section 3 describes our model in details. Section 4 introduces experiments which compare our model with others in two different tasks.

978-1-4673-9808-4/15 $31.00 © 2015 IEEE
DOI 10.1109/SKG.2015.55

II. RELATED WORKS

Graph-based model is one of the most important unsupervised methods used to solve different problems in networks such as ranking, community discovery, etc. When modeling text into a network or a graph, graph-based methods can be applied to solve syntax or semantic problems. Ranking vertex is an important issue in graph-based algorithms. When modeling text documents with terms, phrases or sentences as nodes, with the co-occurrence relation or similarity as edges, we can apply ranking algorithms to give weights to terms, sentences, etc.. So, graph-based methods are an flexible modeling tools.

A. Graph-based method

Graph-based method has been widely used to solve various problems of networks. The Web is a huge network. Web pages can be seen as nodes and the hyperlinks among different pages can be seen as arcs which has direction to represent whether it links to others or it is linked to by others. PageRank [2] and HITS [4] are two well-known graph-based methods which are used to compute the weight of a Web page. PageRank has one variable to represent the importance of a node while HITS has two: hub weight and authority weight. Both use the adjacent matrix of graph as the iteration factor to iteratively obtain the stationary distributions of node weight vector. PageRank style algorithms have been applied to various networks like the citation network [25]. Graph-based methods have also been applied in IR (information Retrieval) and NLP (Natural Language Processing)[1] [14][17][20].

The key to graph-based method is the way abstracting the problem as a graph. Actually, the node and edge of the graph can be anything as long as they are reasonable. We can model a document by graph in different ways. For example, in [6], a textual document is represented as an un-weighted directed graph whose vertices denote unique terms and whose edges denote co-occurrences among the terms within a fixed-size window. The edge direction is determined by the term order. In [3], sentences are represented as nodes; the relations between sentences are represented as the weighted edge. The text therefore can be seen as a weighted undirected graph. Using the undirected graph iteration formulas, the weight of node will converge after a few iterations. Then all the sentences can be ordered by their weights. [7] and [8] are applications of graph-based method in labeling topics and entity linking.

B. Ranking

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

Ranking plays a very important role in information retrieval. Ranking related Web pages in different orders can make a big difference to user experience. Nobody likes finding what he/she wants in a mess of items. Studies about ranking have never stopped. Graph-based methods have been successfully used in ranking the Web networks, sentence, and texts [2][3][4][23][24]. Current researches try to personalize the search result to reflect different requirements [9][19]. [5] improves the original TF-IDF method for more meaningful ranking.

Ranking has also been widely used to summarize documents [12][13][21][22]. [10] gives a detailed comparison of different learning-to-rank methods in summarization. Query specific sentence ranking is also studied [23]. [11] uses a graph-based sentence ranking method in multi-documents summarization, and aims at getting the update summarization of a collection of documents. Ranking keywords, sentences, paragraphs can help to identify the important information in a document. However, when using the extraction method in summarization, the summarized information is not so coherent [13]. [12] is a complete introduction to summarization and related methods

III. MODEL

A. Assumptions

A scientific literature often has a good structure including sections, sub-sections, paragraphs etc., which is used to show the structural semantics of paper. Figure 1 is a description of such a general structure of a scientific literature. This structure information can tell us a lot. For example, keywords in abstract or in conclusion of a paper are deemed more important and more abstractive or general. This kind of assumptions may help improve the performance of information retrieval over scientific literatures. In this paper, we summarize six such assumptions about the structural relations as follows:

1. One sentence is more closely related to sentences in the same paragraph than to sentences in different paragraphs.
2. One paragraph is more closely related to paragraphs in the same section than to paragraphs in different sections
3. The importance of a word is determined by the sentence it is located, by the paragraphs it is located, and also by the sections it is located.
4. The importance of a sentence is determined by the words in it.
5. The importance of a paragraph is determined by the sentences included in it
6. The importance of a section is determined by the paragraphs included in it.

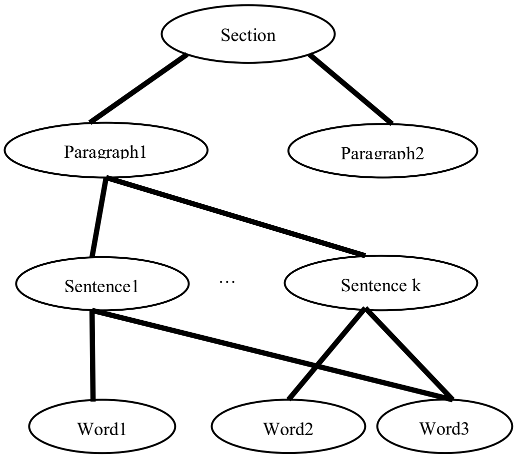

Figure 1. General scientific literature's structures

d	– Document
c	–Section weight vector
n_c	–Section number
p	– Paragraph weight vector
n_p	– Paragraph numbers
s	– Sentence weight vector
n_s	– Sentence numbers
w	– Word weight vector
n_w	– Word number
W_S	– Words to sentences matrix
S_P	– Sentences to paragraphs matrix
P_C	– Paragraphs to sections matrix

Figure 2. Notations

B. General model

Figure 2 gives the notation list used in this paper. The most important part is how to model a document to reflect the assumptions. To reflect those inherent relationships among words, sentences, paragraphs and sections in one paper, we try to build a hybrid graph to incorporate them all. But to this end, we separately define three different matrices to show different levels of relationship. They are constructed as follow:

1. *W_S* matrix: the element in position (i, j) represent the count or TF-IDF of words[i] in sentences[j].
2. *S_P* matrix: the element in position (i, j) represents sentences[i] belongs to sections[j].
3. *P_C* matrix: the element in position (i, j) represent that paragraphs[i] belongs to sections[j]

These three matrices reflect the basic structure of a scientific literature. Then, for each structural element in the paper, a weight vector is defined as *w, s, p* and *c*, for words, sentences, paragraphs, and sections respectively. Finally, to connect those elements into one matrix, we design an iterative process over the relationship matrices and the weight vectors so that they can act with each other to iteratively approach a stable points of weight vectors.

Figure 3 gives the general model which update *w*, *s*, *p* and *c* iteratively. Fig. 4 gives a vivid description about this procedure where numbers with circle indicate the iteration step in Fig 3.

Step 1 randomly initializes vector *w*. Then, in Step 2, the sentence vector *s* is updated by the word vector *w* through the sentence-word matrix. Step 3 reflects assumption 5, which computes paragraph weight as the sum of weight of all the sentences in that paragraph. Step 4 is a reflection of assumption 6. In step 5, we use assumptions (1) to (3) and compute word weight *w* as a simple combination of sentence weight, paragraph weight and section weight where word is located. In whole, all weight vectors will be updated in one iteration though multiplying by three matrices. After enough rounds of iteration, those weight vectors will converge and the iteration is stopped. We can sort the weight vectors to rank word, sentence, paragraph and sections.

Step 1: initialize word weight vector $\widehat{w}^{(0)}$ as a random unit vector with length n_w

Step 2: update and normalize *s*:

$$s^{(n)} = W\_S^T * \widehat{w}^{(n)} \quad (1)$$

$$\hat{s}^{(n)} = \frac{s^n}{sum(s^n)} \quad (2)$$

Step 3: update and normalize *p*:

$$p^{(n)} = S\_P^T * \hat{s}^{(n)} \quad (3)$$

$$\hat{p}^{(n)} = \frac{p^n}{sum(p^n)} \quad (4)$$

Step 4: update and normalize *c*:

$$c^{(n)} = P\_C^T * \hat{p}^{(n)} \quad (5)$$

$$\hat{c}^{(n)} = \frac{c^{(n)}}{sum(c^{(n)})} \quad (6)$$

Step 5: update and normalize *w*:

$$w^{(n+1)} = W\_S \times \hat{s}^{(n)} + W\_S \times S\_P \times \hat{p}^{(n)} + W\_S \times S\_P \times P\_C \times \hat{c}^{(n)} \quad (7)$$

$$\widehat{w}^{(n+1)} = \frac{w^{(n+1)}}{sum(c^{(n)})} \quad (8)$$

Step 6: go back to step 2 until *w, s, p, c* converge.

Figure 3 Iteration Procedure

C. Variants

The matrices can have different implementations. For example, *W_S* matrix can be a zero-one matrix, in which zero in position (i, j) means that word i did not occur in sentence j, one means the other way. *W_S* matrix can also be a TF-IDF matrix. To get this matrix, each sentence can be seen as a unit, and then one document can be seen as a sentence set to apply the TF-IDF method.

For formula (1), (2), (3) and (4) in Fig 3, there are also several variations. Figure 5 shows another update process which changed the formula (4) in the step 5 to the following three steps:

$$p^{(n+1)} = P\_C \times c^{(n)} \quad (9)$$

$$s^{(n+1)} = S\_P \times p^{(n+1)} \quad (10)$$

$$w^{(n+1)} = W\_S \times s^{(n+1)} \quad (11)$$

Formula (5) uses the result section weight vector from the result of step 1 to 4. Formula (6) uses the result paragraph weight vector p of (5) and produces the input for formula (7). The difference from Fig 4 lies in that updating of the word vector w is indirectly by section, paragraph and sentence nodes in Fig 5 while in Fig 4, all updating on the word vector w are directly performed.

In the experiment Section, we will compare these variants with the model in Figure 3 and run them on the benchmark data set to figure out the difference in their performance.

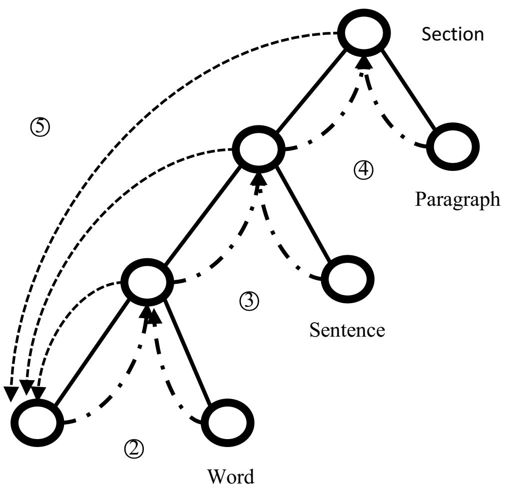

Figure 4. Description of the iteration procedure. The dot-dashed lines represent the update process of step 2, step 3, and step 4. Dashed lines represent the update process of step 5. The circled numbers represent the step number

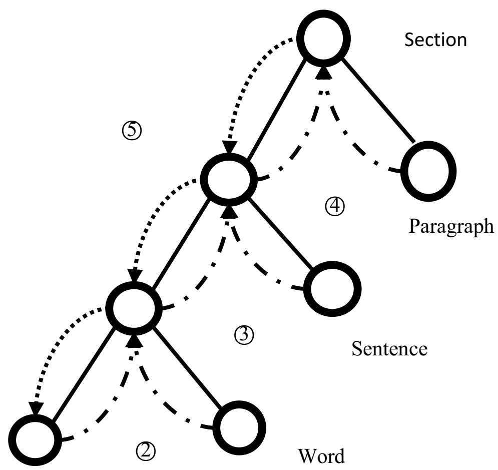

Figure 5. This is another variant, all the symbols have the same meanings as in the Figure 4

D. Matrix manipulation of the model

Actually, the general model presented in Figure 3 can be written in the matrix manipulation form for further analysis. The following matrix representation is equal to the iteration process given in Figure 3. In this process, we ignore the normalization. Because the normalization value is a constant, ignoring it will not influence the derivation process.

From (1), we have that (2) is equal to (8):

$$p^{(n)} = S\_P^T \times W\_S^T \times w^{(n)} \tag{12}$$

From (3) and (8), we have:

$$c^{(n)} = P\_C^T \times S\_P^T \times W\_S^T \times w^{(n)} \tag{13}$$

From (4), (1), (8) and (9), we have:

$$\begin{aligned} w^{(n+1)} = {} & (W\_S \times W\_S^T + \\ & W\_S \times S\_P \times S\_P^T \times W\_S^T + \\ & W\_S \times S\_P \times P\_C \times P\_C^T \times S\_P^T \times W\_S^T) \times w^{(n)} \end{aligned} \tag{14}$$

Let

$$\begin{aligned} A = {} & W\_S \times W\_S^T + \\ & W\_S \times S\_P \times S\_P^T \times W\_S^T + \\ & W\_S \times S\_P \times P\_C \times P\_C^T \times S\_P^T \times W\_S^T \end{aligned}$$

Then we have the final w satisfy equation (10):

$$Aw = w \tag{15}$$

After getting w, it is easy to get s, p and c from equation (1) to (6).

The characteristics of matrix A:

1. A is symmetric
2. A is almost surely fully connected. Basically, the graph's connectivity is determined by the connectivity of two words. If two words occur in the same sentence, or in the same paragraph, or the same section, they will be connected in A. In general there is little possibility that two words are disconnected in A. Besides, we can add a root node upon section nodes to connect all the sections, which will ensure the connectivity of the matrix A.

The above two properties of A will ensure the iteration on w will converge to a stable distribution. Our experiment results indicate that our model will converge after 5~30 iterations.

IV. EXPERIMENT

In order to study the effectiveness of using the structural information on identifying the important information in scientific literature, we compared the proposed methods with other that have less structural information on ranking sentences in one scientific paper. The main comparison is to evaluate the sentence ranking results of different methods. First, we introduce the corpus; then, we evaluate the proposed model and its variants; finally, we show the results of comparisons the model with others models like the TF-IDF model.

A. Dataset, pre-processing and tasks

To evaluate the ranking result, we extracted 175 papers from the proceedings of ACL2014 of the ACL Anthology [18] as our benchmark document set. Words, sentences, section and paragraphs are parsed from the paper documents and *W_S, S_P* and *P_C* matrix are built. In order to have a better understanding of our model's advantages and disadvantages, we designed three tasks.

TASK I. In the first task, we manually annotated key sentences from the introduction section of each literature in the benchmark. Based on the length of introduction, we may give four to twelve key sentences. These sentences can summarize the main point in the introductions. Generally speaking, the extracted sentences are usually representative.

In order to evaluate the performance of different models on TASK I, three measures are given: precision, recall and F1-measure. The formulas are given below:

$$Precision = tp / (tp + fp) \tag{16}$$

$$Recall = tp / (tp + fn) \tag{17}$$

$$F1\text{-}measure = precision * recall / (precision + recall) \tag{18}$$

The elements in above three formulas are explained in table I.

TABLE I SYMBOLS

	Relevant	**Non-relevant**
Retrieved	True Positive: tp	False Positive: fp
Not retrieved	False Negative: fn	True Negative: tn

First, we apply our model to one paper and obtain a ranking result of the whole sentences. Then, we select from it the top-k introduction sentences and use the manually selected sentences as the benchmark to calculate the precision, recall and F-1 scores.

TASK II. In the second task, we define three measures to judge whether the extracted sentences are representative or not. The measures are explained in figure 6.

$P_{important}$: The average portion of important sentences.
P_{title} : The portion of papers in which titles were extracted.
C_{other}: The average number of sentences which did not appear in the sections we listed.

Figure 6. measures explanation

The basic assumption here is that the important sentences are more likely to be sentences from sections which are representative. In the experiment, we regard the result section, the introduction section, the conclusion section, the abstract section, the discussion section and the method section as important sections and sentences from these sections are more representative.

TASK III. In this task, key words were extracted from by the term count model. The term count model just ranked all the words based on their frequencies in a scientific literature. Then, we evaluate the results of word ranking result of our model by comparing it with the term count model to see how this model can reflect the word counts.

B. Compare the variants of our model

In section III, several variants have been proposed. We summarize four versions below and will compare those four versions with the original one:

v1. We use word count matrix as the W_S matrix, where each row represents the word count in different sentences and each column represents the word count distribution of different sentences.

v2. We use the TF-IDF matrix as the W_S matrix. We regard each sentence as one document, and compute the TF-IDF matrix from the sentence set of one paper.

v3. Update *w* with formula (19)

$$w^{(n+1)} = W\_S \times w^{(n)} \tag{19}$$

v4. Update *w* with formula (20)

$$w^{(n+1)} = W\_S \times s^{(n)} + W\_S \times S\_P \times p^{(n)} \tag{20}$$

The performances of these four versions on $P_{important}$, P_{title} and C_{other} are shown from TABLE III to TABLE V. By comparing the results of v1 and v2, we can see that the TF-IDF matrix used in v2 is more helpful for ranking sentences. The results of v2, v3 and v4 indicate that adding the graph information can improve the f-measure in TASK I and have a better performance in TAKS II. In addition, the comparison of v3 and v4 tells us that adding the section information may degrade the performance. The reason behind might be that section information makes sentences harder to differentiate from each other.

C. Compare with other methods

We compared our model with four models which use less structural information on TASK I and TASK II: (1) a pure TF-IDF model (TF-IDF); (2) a graph-based model which uses sentences as nodes and the similarity of sentence as weighted edges (GS); (3) a graph-based model which uses word as nodes and words in a given window will set up edges (GW).

In the TF-IDF model, we sum each column the given W_S matrix to get the sentence weight corresponding to that column. In the GS model, we use the PageRank algorithm to directly derive the weight of each sentence. In the GW model, the weight of each word is known after running PageRank over the graph. Then we compute the weight of each sentence through the summation of the weight of words in that sentence.

Compared the performance of these three methods with our model, we find that our model are more accurate when identifying the important sentences and the TF-IDF model is better compared with other two graph-based methods in ranking sentences (See Table II, III, IV, V).

Finally, we summarize the results in TABLE VII by listing the best score winner. From the table we can see that in most tests, V3 and V4 models achieve the best score and V3 wins more tests than V4. V2 has the best score in C_{other}..

D. Compare the keywords

Table VI gives the top 15 keywords in each models on paper titled 'Simple Negation Scope Resolution through deep Parsing: A semantic Solution to a Semantic Problem' [26] from the ACL2014. The table shows that our model can reflect the word count in the document.

V. CONCLUSION

In this paper, we explored the way to use the structural information to rank sentences in scientific literatures based graph ranking models. The structural information we explored includes the word co-occurrence relation, the sentence-paragraph relation and the paragraph-section relation. By modeling those relationships into an iterative process over the relationship matrices, we can simultaneously obtain the rank of the words, sentences, paragraphs and sections. In this work we mainly focused on how sentences are ranked in the model. In order to have a better understanding of our model, we designed three tasks to explore its characteristics and compared different variants of our model and also compared our model with other models. The experiment result shows that our method can have a better sentence ranking result. The proposed method can be extended to text without paragraph and section information. Future work will be conducted on extending this ranking model to words, paragraphs, and sections together with semantic relationship network from outside sources such as Wikipedia and other literatures or textbooks.

TABLE VII. Summary of Task I && II

Metric	Winner
TASK I. Precision (Top 5)	v4
TASK I. Recall (Top 5)	v3
TASK I. F-score (Top 5)	v4
TASK I. Precision (Top 10)	v3
TASK I. Recall (Top 10)	v3
TASK I. Precision (Top 10)	v3
TASK I. Recall (Top 15)	v4
TASK I. F-score (Top 15)	v3
TASK I. F-score (Top 15)	v4
TASK II. $P_{important}$ (Top 5)	v3
TASK II. $P_{important}$ (Top 10)	v4
TASK II. $P_{important}$ (Top 15)	**TF-IDF**
TASK II. $P_{important}$ (Top 20)	v3
TASK II. P_{title} (Top 5)	v3
TASK II. P_{title} (Top 10)	v3
TASK II. P_{title} (Top 15)	v3
TASK II. P_{title} (Top 20)	v3
TASK II. C_{other} (Top 5)	v2
TASK II. C_{other} (Top 10)	v2
TASK II. C_{other} (Top 15)	v2
TASK II. C_{other} (Top 20)	v2

ACKNOWLEDGMENT

Research Supported by the Open Project Funding of CAS Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences (No. IIP2014-2). This work was also partially supported by National Science Foundation of China (No.61075074 and No.61070183)..

REFERENCES

[1] R. Mihalcea, D. Radev, "Graph-based natural language processing and information retrieval", Cambridge University Press, 2011

[2] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998

[3] R. Mihalcea, "Graph-based ranking algorithms for sentence extraction, applied to text summarization", in *Proc.* of the ACL 2004 on Interactive poster and demonstration sessions, Association for Computational Linguistics, p. 20, 2004.

[4] M. Kleinberg. "Hubs, authorities, and communities." ACM Computing Surveys (CSUR) 31.4es p. 5, 1999.

[5] J. H. Paik, "A novel TF-IDF weighting scheme for effective ranking", In *Proc.* of the 36th international ACM SIGIR conference on Research and development in information retrieval, pp. 343-352

[6] Rousseau, François, and Michalis Vazirgiannis, "Graph-of-word and TW-IDF: new approach to ad hoc IR". In *Proc.* of the 22nd ACM international conference on Conference on information & knowledge management, pp. 59-68.

[7] N. Aletras, M. Stevenson, "Labelling Topics using Unsupervised Graph-based Methods", In *Proc.* of the Association for Computational Linguistics, pp. 631-636, 2014.

[8] X. Han, L. Sun, J. Zhao, "Collective entity linking in web text: a graph-based method", in *Proc.* of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pp. 765-774, 2011

[9] S. Qiu, J. Cheng, T. Yuan, et al, "Item group based pairwise preference learning for personalized ranking", In *Proc.* of the 37th international ACM SIGIR conference on Research & development in information retrieval, pp. 1219-1222. 2014

[10] F. Jin, M. Huang, X. Zhu, "A comparative study on ranking and selection strategies for multi-document summarization", In *Proc.* of the 23rd International Conference on Computational Linguistics: Posters. Association for Computational Linguistics, pp. 525-533, 2010.

[11] X. Li, L. Du, Y. D. Shen, "Update summarization via graph-based sentence ranking", Knowledge and Data Engineering, IEEE Transactions on, vol. 25(5), pp. 1162-1174.

[12] A. Nenkova, S. Maskey, Y. Liu, "Automatic summarization", In *Proc.* of the 49th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts of ACL 2011, Association for Computational Linguistics, p. 3, 2011.

[13] V. Gupta, G. S. Lehal, "A survey of text summarization extractive techniques", Journal of Emerging Technologies in Web Intelligence, vol. 2(3), pp. 258-268, 2010.

[14] C. D. Manning, P. Raghavan, "Introduction to information retrieval", Vol. 1. Cambridge: Cambridge university press, 2008.

[15] https://en.wikipedia.org/wiki/Vector_space_model

[16] K. S. Jones. "A statistical interpretation of term specificity and its application in retrieval." Journal of documentation 28.1 (1972): 11-21.

[17] R. Mihalcea, P. Tarau. "TextRank: Bringing order into texts." Association for Computational Linguistics, 2004.

[18] https://aclweb.org/anthology/

[19] S. Fisher, B. Roark, "Query-focused summarization by supervised sentence ranking and skewed word distributions", In *Proc.* of the Document Understanding Conference, DUC-2006, New York, USA. 2006.

[20] F. Wolf, E. Gibson. "Paragraph-, word-, and coherence-based approaches to sentence ranking: A comparison of algorithm and human performance", Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, p. 383, 2004.

[21] S. S. Ge, Z. Zhang, H. He. "Weighted graph model based sentence clustering and ranking for document summarization." Interaction Sciences (ICIS), 2011 4th International Conference on. IEEE, pp. 90-95, 2011.

[22] N. Gupta, P. C. Saxena, J. P. Gupta. "Document summarisation based on sentence ranking using vector space model." International Journal of Data Mining, Modelling and Management vol. 5.4, pp. 380-406, 2013.

[23] S. Fisher, B. Roark, "Feature expansion for query-focused supervised sentence ranking", Document Understanding (DUC 2007) Workshop Papers and Agenda. 2007.

[24] D. McDonald, H. Chen. "Using sentence-selection heuristics to rank text segments in TXTRACTOR ", In *Proc.* of the 2nd ACM/IEEE-CS joint conference on Digital libraries. ACM, pp. 28-35, 2002.

[25] X. Jiang, X. Sun, H. Zhuge, "Towards an effective and unbiased ranking of scientific literature through mutual reinforcement", In *Proc.* of the 21st ACM international conference on Information and knowledge management. ACM, pp. 714-723, 2012.

[26] W. Packard, E. M. Bender, J. Read, S. Oepen, R. Dridan, "Simple negation scope resolution through deep parsing: A semantic solution to a semantic problem", In Proc. of the 52nd annual meeting of the Association for Computational Linguistics. pp. 69–78 ,2014

TABLE II PERFORMANCE IN TASK I

Methods\measures		Top 5			Top10			Top 15		
		precision	**recall**	**F1 value**	**precision**	**recall**	**F1 value**	**precision**	**recall**	**F-value**
Variants of our model	**V1**	0.4223	0.3082	0.1749	0.3822	0.5349	0.2192	0.3470	0.6894	0.2273
	V2	0.4319	0.3123	0.1779	0.3876	0.5418	0.2220	0.3454	0.6870	0.2262
	V3	0.45	**0.3239**	0.1849	**0.3954**	**0.5533**	**0.2267**	0.3482	**0.6938**	0.2282
	V4	**0.4512**	0.3256	**0.1856**	0.3906	0.5454	0.2237	**0.3490**	0.6935	**0.2286**
Other methods	**Sentence graph (GS)**	0.3897	0.2851	0.1615	0.3539	0.4992	0.2034	0.3317	0.6618	0.2174
	Word graph (GW) (window = 3)	0.1295	0.0908	0.0520	0.1996	0.2741	0.1133	0.2405	0.4711	0.1567
	Pure TF-IDF (TF-IDF)	0.4114	0.2959	0.1689	0.3737	0.5208	0.2139	0.3486	0.6924	0.2282

TABLE III. PERFORMANCE IN TASK II ON $P_{important}$ (Larger is better)

MEASURE ONE		Top 5	Top 10	Top 15	Top 20
Variants of our model	**V1**	0.4287	0.4166	0.4003	0.4034
	V2	0.4011	0.4034	0.3911	0.3882
	V3	**0.5114**	0.4856	0.4662	**0.4568**
	V4	0.4954	**0.5005**	0.4685	0.4566
Other methods	**Sentence graph (GS)**	0.3712	0.3488	0.3329	0.3278
	Word graph (GW) (window = 3)	0.3528	0.3137	0.2892	0.2778
	Pure TF-IDF (TF-IDF)	0.5	0.4850	**0.4693**	0.4540

TABLE IV PERFORMANCE IN TASK II ON P_{title} (Larger is better)

Methods \ measures		Top 5	Top 10	Top 15	Top 20
Variants of our model	**V1**	0.0114	0.0172	0.0229	0.0459
	V2	0.1494	0.2471	0.3390	0.4195
	V3	**0.4022**	**0.5114**	**0.5804**	**0.6666**
	V4	0.2528	0.3850	0.4597	0.5402
Other methods	**Sentence graph (GS)**	0	0.0057	0	0
	Word graph (GW) (window = 3)	0.0057	0.0172	0.0114	0.0172
	Pure TF-IDF (TF-IDF)	0.0172	0.0172	0.0172	0.0287

TABLE V PERFORMANCE IN TASK II ON MEASURE C_{other} (Smaller is better)

Methods \ measures		Top 5	Top 10	Top 15	Top 20
Variants of our model	**V1**	1.9252	3.1839	4.0689	4.8160
	V2	**1.5919**	**2.4655**	**3.1551**	**3.7586**
	V3	1.9022	3.2758	4.3390	5.1149
	V4	1.6264	2.7298	3.7068	4.4770
Other methods	**Sentence graph (GS)**	2.4252	3.9827	5.0689	5.8908
	Word graph (GW) (window = 3)	2.3735	4.4597	6.0632	7.3563
	Pure TF-IDF (TF-IDF)	1.982	3.4942	4.5344	5.4827

TABLE VI TASK III: TOP 15 KEYWORDS GENERATED BY THREE MODELS

V3		**V4**		**Word Count**	
word	**weight**	**word**	**weight**	**word**	**count**
Crawling	0.02971997	scope	0.01635293	scope	64
Scope	0.01708008	system	0.01293138	negation	51
Negation	0.01595256	negation	0.0126892	system	40
Mrs	0.01492882	task	0.01049029	task	37
Cue	0.01187101	crawling	0.00970509	crawling	37
Argument	0.01141487	mrs	0.0095719	cue	30
Label	0.01103595	words	0.00791609	analysis	30
Task	0.01091048	crawler	0.00777409	mrs	27
System	0.01059133	representations	0.00726127	shared	22
Functor	0.01051087	cue	0.00724538	negated	21
Negated	0.00913552	erg	0.00707575	guidelines	21
Ep	0.00901759	exe	0.00701627	crawler	21
Semantic	0.00840691	semantically	0.00680568	semantic	20
Shared	0.00808782	shared	0.00652963	data	18
Algorithm	0.00790919	rules	0.00652699	semantically	17