Semantic based Sentence Ordering Approach for Multi-Document Summarization

P.Sukumar, K.S.Gayathri

Abstract*— With the rapid growth of online information which is unstructured in nature poses a great challenge to the text mining algorithms to retrieve useful and meaningful information in an efficient way. However larger amount of data are readily available, it is very difficult to access the required information at the right time and also in the most appropriate form. Therefore a systematic approach called multi-document summarization is required to generate a summary about particular topic. The main focus of document summarization is sentence ordering and ranking. The existing system for sentence ordering deals with the measures such as chronology, topical, precedence and succedence experts. The main drawback of existing system is, it does not address the semantic relationship between the sentences in the summary which is necessary to create a meaningful summary. The proposed system addresses the semantic relationship between sentences in the summary using wordnet synsets. This system builds an entailment model which infer the logical relationship among the sentences when arranging the sentences in the summary. Graph method is used for ranking the sentences, where nodes represents the sentences and the edges represents the preference value of one sentence over another sentence. The proposed system provides an efficient summary which is considerably increases the meaningfulness of the final summary and also typically recovering the user from the information overload problem by giving quick and efficient access to required information.

***Index Terms*—Multi-document summarization, sentence ordering, sentence ranking, semantic expert, text entailment expert.**

I. INTRODUCTION

Information plays a crucial role in the modern world. Today's era of internet technology gives us a huge volume of unstructured data which is practically impossible for a user to read all the retrieved contents. It is therefore significant to find methods of allowing users to extract the main idea from collection of information. Automatic document summarization of multiple documents would thus be immensely useful to meet such information seeking goals by providing an approach for the user to quickly sight highlights or relevant portions of documents. Multi-document summarization is an automatic procedure that generates a generic or topic-focused summary by reducing documents in size while holding back the important characteristics of the original documents.

Manuscript Received May 2014.

P. Sukumar, PG Scholar, Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, Sriperumbudur, Chennai, India.

K.S.Gayathri, Associate Professor, Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, Sriperumbudur, Chennai, India.

In multi-document summarization, the most challenging issue is more than one document sharing the same topic or similar topic and then it leads to the information overhead. To overcome this problem automatic document summarization is used. There are so many issues associated with the multi-document summarization which are to be addressed such as simple word matching and counting frequencies. These criteria's alone not able to find accurate similarity between sentences in the summary because documents may consists of different words to describe the same events. The semantic relationship among the sentences should also be taken into account. There are two main steps in document summarization, one is information extraction and second is sentence ordering. The proposed system mainly focuses on the sentence ordering and ranking. Sentence ordering is the process of arranging the extracted sentences in probable combination so that the final summary would be meaningful which in turn improves the readability of the summary. There are several methods proposed earlier for sentence ordering, which are not considered the semantic aspects rather word matching. This system uses wordnet tool as external resources to find the meaning of each word in calculating the term frequency of a particular word in the sentences of document. Sometimes sentences which derives the truthfulness of another sentences. In such cases we should identify logical inferences between sentences in the summary which we call it as text entailment relations. If the meaning of one sentence can be inferred from another sentence, we can say both are entailed from each other. The entailment between sentences can be found out by using symmetric measure such as cosine similarity and non-symmetric measures such as lexical and syntactic similarity.

II. RELATED WORK

One of the main challenging task in multi-document summarization is sentence ordering and ranking. There are several methods have been proposed for sentence ordering so far. Document summarization has huge applications in natural language processing. Based on the similarity between sentences, the final summary will be created. So irrespective of the sentences that do not share the same surface form, the similarity judgment should be taken. Recently developed approach for sentence ordering is by Bollegala et al, presented a preference learning approach to sentence ordering for multi–document summarization. This work used five preference measures such as chronology, probabilistic, topical closeness, precedence and succession to find similarity between sentences in the summary. Greedy algorithm is used to find the total ordering among the sentences and graph model is used to order the sentences.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

List wise approach to sentence ranking has been developed by Fen Xia et al. The properties of the loss functions including consistency, continuity, soundness, convexity, efficiency and differentiability are investigated to find rank. A sufficient condition on consistency for ranking is given first time in this work. Using support vector machine, penget et al developed a sentence ordering method for multi-document summarization. The summary sentences are put into different categories group according to the source documents. Sequence of each group is decided based on the directional relativity of adjacent sentences. In [2], a feature specific sentence ranking method has been developed. It uses dependency parser rather than traditional vector space model, so that the dependency between each pair of words is strongly calculated to create effective summary. In [3], an approach is developed which personalizing the vector space model and utilizing Latent Semantic Analysis (LSA) on it to bring forth update summary from multiple documents. A two way textual entailment system was developed by pakray et al. The lexical and syntactic features are used to determine the entailment between text and hypothesis. The machine learning algorithm is trained using these two features. The classifier return either "yes" or "no" based on the entailment decision. All the above existing approach to sentence ordering in multi-document summarization does not addressed the semantic relationship and logical inference between sentences in the summary which is very important to generate meaningful summary. The proposed system mainly focuses on finding the semantic and logical relationship between sentences by building effective sentence ordering strategies.

III. PROPOSED SYSTEM

A. Problem Statement

In multi-document summarization, sentence position in the original document does not provide clue to sentence arrangement. Inter document order of the sentences must be considered. The problem is to find feasible methods to determine the most probable permutation of the given sentences and to improve the quality and readability of the summary.

B. System Design

The very big challenge of natural language processing is providing computer system with the linguistic knowledge in order to perform language oriented task. In multi-document summarization, the sentences extracted from multiple documents are written by various authors in different period of time. Hence it is very challenging task to perform the arrangement of sentences in the summary. The Figure 1 shows the overall framework of sentence ordering.

C. Preprocessing

Input documents are given to the preprocessing block where each sentences are converted into smaller number of units called tokens. Processed tokens may contain words, numbers, punctuation, etc. The terms which does not contribute to the meaning of the sentence called stop words should be removed which in turn avoids the processing overhead by processing unwanted words.

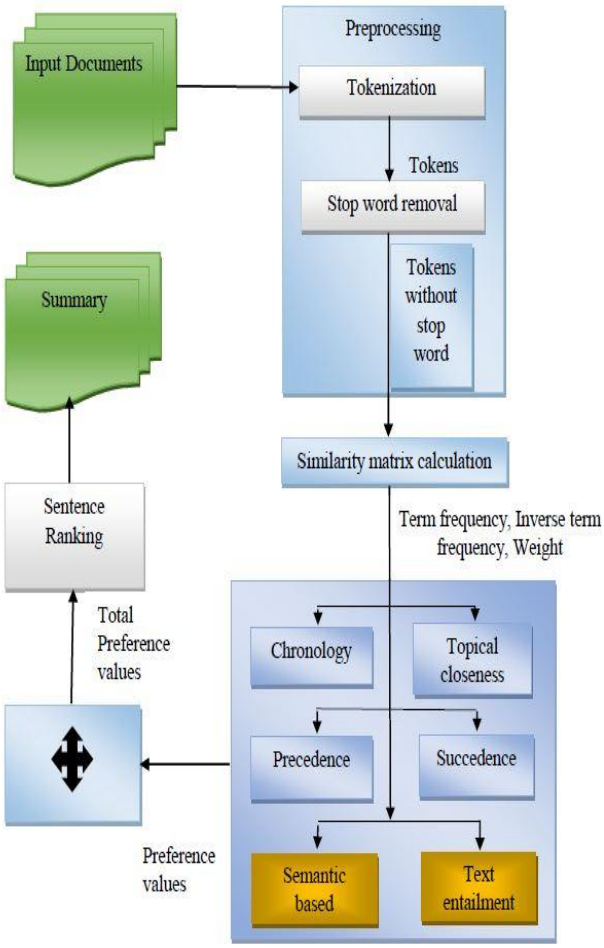

Figure 1. Overview of Sentence Ordering

D. Similarity Matrix Calculation

In order to know how important a word is to a document or summary, word weight is calculated. It is calculated using Term Frequency–Inverse Sentence Frequency (TF–ISF). Term Frequency helps to know the fact that how many times a word appears in a document or summary. Inverse Sentence Frequency measures what percentage of sentences contains that word.Term frequency is given by the Equation (1).

$$\text{TermFrequency} = \frac{n_j}{\sum_k n_k} \quad (1)$$

In Equation (1), nj represents the number of occurrences of wordj in the summary and nk represents the total number of words in the summary. Inverse Sentence Frequency is given by the Equation (2).

$$\text{ISF} = \log \frac{N}{n_i} \quad (2)$$

In Equation (2), N is the total number of sentences in the summary and ni is the number of sentences containing that particular term. Term weight is calculated using the following Equation (3).

$$\text{Term weight} = \text{TF}^*\text{ISF} \quad (3)$$

E. Sentence Ordering

In the proposed system, sentence ordering block includes six preference measures such as chronology, topical closeness, precedence, succedence,

semantic and text entailment experts. Each ordering strategies are independent of each other. The set of input sentences are given to each of the experts and it will return preference value of one sentence over another sentence as values in the range 0 to 1. [5] The preference values are calculated using the preference function as follows,

$$\mathrm{PREF}_e(u,v,Q) \in [0,1] \quad (4)$$

In Equation (4), Q is the set of sentences which has been ordered so far and u, v are two sentences to be ordered. If the expert prefers u – v then it returns a value greater than 0.5. In the extreme case where the expert is absolutely sure of preferring u – v it will return the value 1. On the other hand, if the expert prefers v – u it will return a value less than 0.5. In the extreme case where the expert is absolutely sure of preferring v – u it will return 0. The linear weighted sum of these individual preference functions is taken as the total preference by the set of experts as follows,

$$\mathrm{PREF}_{\mathrm{total}}(u,v,Q) = \sum_{e \in E} W_e \mathrm{PREF}_e(u,v,Q) \quad (5)$$

In Equation (5), E is the set of experts and we is the weight associated with expert $e \in E$. These weights are normalized such that the sum of them equals to 1. Publication timestamps are used to decide the chronological order among sentences extracted from different documents. In topical closeness expert, summary can be created by grouping sentences which are topically related. The information stated in the document from where the sentence was extracted is be considered to judge the order. If the sentences preceding and succeeding the extracted sentence in the original document match well with the so far ordered summary, it is suitable to order the sentence next in the summary. Precedence and succession relations are used to find the ordering among the sentences in a better way. Semantic Expert Wordnet is a large lexical database of English language used to find the meaning of a particular word in a document. The Java interface for wordnet Searching (JAWS) is an API that provides Java applications with the ability to retrieve data from the Wordnet database. It groups English words into sets of synonyms called synsets. It provides the records of various semantic relations between these synonym sets.

Instead of counting the word occurrences of particular word by directly matching, the proposed system consider the wordnet synsets too. The semantic expert finds the similarity between sentences accurately compared to the existing system.

F. Text Entailment Expert

The text entailment expert system is used to find the logical relationship between two sentences in the summary. The proposed entailment system uses various symmetric and non-symmetric measures to find the entailment among sentences. The various similarity measures are listed below,

i. Symmetric Similarity Measures

Cosine similarity measure is used to find the similarity between two n-dimensional vector obtained by finding the cosine angle. It is used to find the similarity between sentences in the summary in statistical way. Given two vectors of attributes S_j and S_k, the cosine similarity θ is calculated using the dot product and magnitude as,

$$\mathrm{sim}(s_j, s_k) = \frac{\overrightarrow{s_j} \cdot \overrightarrow{s_k}}{\left|\overrightarrow{s_j}\right| \left|\overrightarrow{s_k}\right|} \quad (6)$$

In Equation (6), S_j and S_k represents sentences in the document collection or summary.

ii. Non-symmetric Similarity Measures

a. Lexical Similarity

1. Lexical Unigram Match

The presence of various unigrams in the sentence *i* is checked against the sentence *j*. Wordnet synsets are used for identifying unigrams match.

If *n1* = common unigram between sentence *i* and sentence j and *n2* = number of unigrams in sentence *j*, then

Lexical_unigram_match=n1/n2. *(7)*

If the value of Lexical_unigram_match is 75% or more unigrams in the sentence *j* is matches either directly or through wordnet synonyms of sentence *i*, then the pair of sentence considered as an entailment.

2. Lexical Bigram Match

Each bigram in the sentence *j* is searched for their presence in the corresponding sentence *i* part.

Lexical_bigram_match = n1/n2. *(8)*

Where, *n1* is the total number of bigram match between sentence *i* and sentence *j* pair and *n2* is total number of bigrams in the sentence *j*. If the value of Lexical_bigram_match is 50% or more bigrams in the sentence *j* is matches with bigrams in the sentence *i*, then the sentence pair is considered as an entailment pair.

3. Lexical Longest Common Subsequence

The longest common subsequence of sentence *i* – sentence *j* pair is the longest sequence of words that is common to both sentences.

Lexical_LCS_match= LCS (sentence i, sentence j)/length of unigrams in the sentence j. *(9)*

If the value of Lexical_LCS_match is 0.80 or more, i.e., the length of the common words in pair of sentence is greater than the length of the sentence *j*, then the sentence pair is considered as an entailment pair.

4. Lexical Skip-gram Match

A skip gram is any combination of words in the order as they appear in a sentence but allowing gap between word occurrences. In the proposed work 1-skip bigram is considered where 1-skip bigram allowing one word gap between words in a sentence as they appear.

Lexical_1_skip_bigram_match=n1/n2. *(10)*

Where, *n1* is the 1_skip_bigram_match between sentence *i* and sentence *j* and *n2* is the total number of unigrams in the sentence j. If the value of Lexical_1_skip_bigram_match is 0.50 or more, i.e., 1-skip bigram match between sentence *i* and sentence *j* is greater than the length of the sentence *j*, then the sentence pair is considered as an entailment pair.

b. Syntactic Similarity

The various syntactic similarity for finding text entailment relations are described in detail. Part-of-speech tagging is the process of marking up the words in a corpus. Mostly based on both its definition as well as its context relationship with adjacent and related words in a sentence. In order to assign parts of speech to each word the proposed system utilizes Stanford Log-Linear Part-of-speech Tagging as in Marneffe produces tagged documents. Then the tagged documents are passed through Stanford parser to retrieve well-formed relationships in a sentence and the output is represented using Stanford typed habituation.

1. Subject-verb Match

The subject and verb of sentence *i* is compared with subject and verb of sentence *j*. If there is a match, a preference value of 1 is assigned. Otherwise the following match will be performed.

2. Wordnet Based Subject-verb Match

If the corresponding sentence *i* and sentence *j* subjects do match in the subject verb comparison but verb does not match, then synsets of verbs will be find out using wordnet. If there is a match, a preference value of 0.5 is assigned. Otherwise the following match will be performed.

3. Subject-subject Match

The subject of sentence *i* is compared with subject of sentence *j*. If a match is found, a preference value of 0.5 is assigned.

4. Object-verb Match

The object and verb of sentence *i* is compared with object and verb of sentence *j*. If there is a match, a preference value of 1 is assigned. Otherwise the following match will be performed.

5. Wordnet Based Object-verb Match

If the corresponding sentence *i* and sentence *j* objects do match in the object verb comparison but verb does not match, then synsets of verbs will be find out using wordnet. If there is a match, a preference value of 0.5 is assigned. Otherwise the following match will be performed.

6. Object-object Match

The object of sentence *i* is compared with object of sentence *j*. If a match is found, a preference value of 0.5 is assigned.

7. Number Comparison

The number in sentence *i* is compared with number in the sentence *j*. If there is a match, a preference value of 1 assigned.

8. Determiner Comparison

The determiner of sentence *i* is compared with determiner in the sentence *j*. If there is a match, a preference value of 1 assigned.

G. Ordering Algorithm

Using the six preference value functions in the former part, the system work out the total preference function using the equation (5). The problem of determining optimal ordering for a given total preference function is done by sentence ordering algorithm. Given an unordered sentences *X* extracted from a set of documents and total preference function, $PREF_{total}(u,v,Q)$ computes a total ordering function among the extracted sentences.[5] The sentence ordering algorithm is given below,

Algorithm 1: Sentence Ordering Algorithm

Input: A set x of the extracted (unordered) sentences and a total preference function

Output: Ranking score p of each sentence t € x

1. V=x
2. Q= ∅
3. **for each** $v \in V$ do
4. $\Pi(v)=\sum_{u\in V} PREF_{total}(v,u,Q)-\sum_{u\in V} PREF_{total}(u,v,Q)$
5. **end for**
6. **while** $V \neq \varnothing$ do
7. $t=\operatorname{argmax}_{u\in V} \prod(u)$
8. $\rho(t)=|V|$
9. V=V-{t}
10. Q=Q+{t}
11. **for each** $v \in V$ do
12. $\Pi(v)=\Pi(v)+PREF_{total}(t,v,Q)-PREF_{total}(v,t,Q)$
13. end for
14. end while
15. return p

First line of the algorithm assigns set of unordered sentences to the set called V. Initially the set called ordered sentences Q will be empty. For each sentence in the set V, we calculate total preference function value in the line 4. The algorithm lines 6 to 10 does, once first iteration of the total preference value calculation over, the sentence with high preference value will be taken and inserted into the ordered set Q and the sentence will be removed from the unordered set. From line 11 to 15 we have to re-calculate preference values of all sentences. The preference values of sentences are calculated based on the sentence which resides already in the ordered summary. The sentence with next high rank value will be added further into the summary. This above is process repeated until there is no more sentences to order further.

IV. EXPERIMENTAL RESULTS

A. Dataset Description

Text Analysis Conference (TAC) released data set for evaluation in the year 2008. The input documents for the proposed method are taken from TAC 2008 AQUAINT–2 collection of newswire articles. AQUAINT–2 collection of news articles span from October 2004 to March 2006 with 48 topics and each topic consists of 10 documents and its summary.

B. Experimental Setup

The experiments are performed in the system with an Intel Pentium Dual Core Processor and 2 GB main memory, running on Windows 7. The hard disk capacity is 320 GB. The algorithm and other calculations are implemented in Java.

C. Performance Evaluation

The output of each participating system is compared against the manually extracted set of sentences for each topic using precision, recall and F-score measure. To evaluate the performance of the proposed sentence ordering approach, we compare the result of the proposed method with the existing method. Four methods are used to order the extracts. Random ordering, chronology ordering, existing system ordering, proposed system ordering. Chronology, succession, precedence, topical closeness, semantic, Text entailment experts are combined in case of proposed system ordering. Automatic system generated summary is compared against the human annotators generated reference summary.

i. Precision

Precision is the fraction of the retrieved sentences that are relevant to the reference sentences. This is equivalent to measuring a precision of continuous sentences in an ordering against the reference ordering. We define p_n as the precision of n continuous sentences in a sentence ordering as follows,

$$P_n = m/N-n+1 \quad (11)$$

In Equation (11), Pn is the precision value, m is the number of continuous sentences appear in both reference and system generated summary, n is length of continuous sentences and N is the number of sentences in the reference ordering. The number of continuous sentences that appear both in an ordering produced by an algorithm as well as in an ordering made by a human annotator for a set of sentences is an indicator of the readability of a sentence ordering produced by the algorithm. Average continuity measure such as precision captures this notion of readability. In the Figure 2 we plotted graph for the precision scores against the length of n continuous sentences.

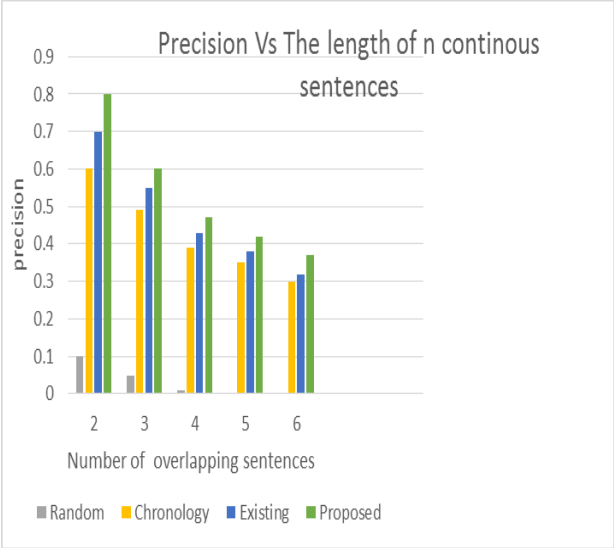

Figure 2. Precision vs Length of n Continous Sentences

ii. Recall

Recall is the fraction of the sentences that are relevant to the query that are successfully retrieved. This is equivalent to measuring a recall of continuous sentences in an ordering against the reference ordering. We define R_n as the recall of n continuous sentences in a sentence ordering as follows,

$$R_n = m/n \quad (12)$$

In Equation (12), R_n is the recall value, m is the number of correct continuous sentences appear in both reference and system generated summary, n is the no of continuous sentences that should have been returned. In the Figure 3 we plotted graph for the recall scores against the length of n continuous sentences.

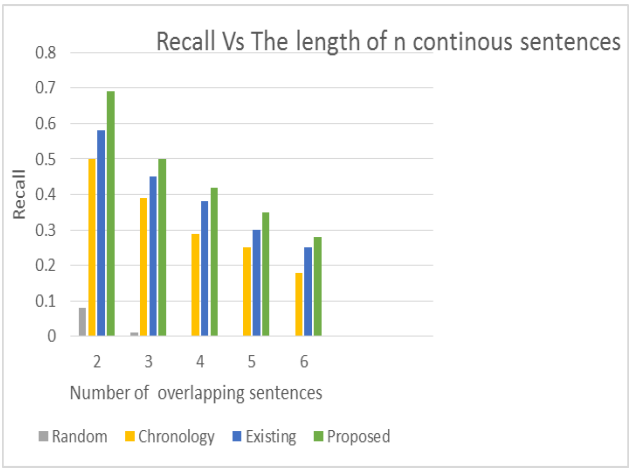

Figure 3. Recall vs Length of n Continous Sentences

V. CONCLUSION AND FUTURE WORK

The proposed entailment model provides a systematic approach for sentences ordering and ranking for multiple documents. A graph model is used for sentence ranking where each nodes represents sentences and edges represents preference value between sentences. The preference values are calculated using chronological, topical closeness, precedence, succedence, semantic and text entailment measures. The advantage of proposed system are, it addresses the semantic relation between sentences and logical inferences between sentences in the summary using symmetric and non-symmetric measures which are not considered in the existing system. Thus this method provides high accuracy compared to statistical methods by providing efficient summary which significantly improves readability and understandability of the user. In future, the causal relation measures will also be taken into account in addition to symmetric and non-symmetric measures to produce coherent summary.

REFERENCES

1. Achananuparp P., Hu X. and Shen X. (2008), 'The evaluation of sentence similarity measures', 10th International Conference on Data Warehousing and Knowledge Discovery 2008.
2. A. Kogilavani, P. Balasubramanie. (2011), 'Semantic summary generation from multiple documents using feature specific sentence ranking strategy', Elixir Comp. Sci. Engg. Volume 40, pp. 5372-5375.
3. A. Kogilavani, P. Balasubramanie. (2012), 'Update Summary Generation based on Semantically Adapted Vector Space Model', International Journal of Computer Applications (0975 – 8887), Vol 42, No.16.
4. Bollegala D., Okazaki N. and Ishizuka M. (2010), 'A bottom-up approach to sentence ordering for multi-document summarization', Information Processing and Management, Vol.46, No.1, pp. 89–109.
5. Bollegala D., Okazaki N. and Ishizuka M. (2012), 'A preference learning approach to sentence ordering for multi-document summarization', Information Science, pp. 78-95.

6. Cai X. and Li W. (2013), 'Ranking Through Clustering: An Integrated Approach to Multi-Document Summarization', Vol.21, No.7, pp. 1424-1433.
7. Castillo J J. (2010), 'An approach to Recognizing Textual Entailment and TE Search Task using SVM', Natural Language Processing, Vol. 44, pp. 139-145.
8. D.O. Seaghdha, A. Korhonen. (2011), 'Probabilistic models of similarity in syntactic context', in: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11), Edinburgh, UK.
9. F. Xia, T.-Y. Liu, J. Wang, W. Zhang, H. Li. (2008), 'Listwise approach to learning to rank: theory and algorithm', in: ICML 2008.
10. Hobbs J. R. (1985). 'Ontological promiscuity', Proceedings of the 23rd annual meeting on Association for Computational Linguistics.
11. J.J. Castillo. (2011), 'A wordnet-based semantic approach to textual entailment and cross-lingual textual entailment', International Journal of Machine Learning and Cybernetics 2, pp. 177–189.
12. Liu Y. and Liang Y. (2013), 'A sentence semantic similarity calculating method based on segmented semantic comparison', Journal of Theoretical and Applied Information Technology, Vol. 48, No. 1, pp. 231-235.
13. Miguel Angel Rios Gaona, Alexander Gelbukh, and Sivaji Bandyopadhyay (2011), 'Recognizing Textual Entailment Using a Machine Learning Approach'.
14. M. Lapata (2003), 'Probabilistic text structuring: experiments with sentence ordering', in: Proceedings of the Annual Meeting of ACL, 2003, pp. 545–52.
15. P.Sukumar and K.S.Gayathri (2014) 'An Effective Sentence Ordering Approach For Multi-Document Summarization Using Text Entailment', in International Journal on Recent and Innovation Trends in Computing and Communication [IJRITCC], Vol.2, Issue.1, pp. 144-149.
16. Partha P., Sivaji B. and Alexander G. (2011), 'Textual entailment using lexical and Syntactic similarity', International Journal of Artificial Intelligence & Applications, Vol. 2, No. 1, pp. 43-58.
17. Peng G., He Y., Tian Y., Tian Y. and Wen W, 'Analysis of Sentence Ordering Based on Support Vector Machine', Pacific-Asia Conference on Knowledge Engineering and Software Engineering 2009.
18. Peng G., He Y., Zhang W., Xiong N. and Tian Y., 'A Study for Sentence Ordering Based on Grey Model', IEEE Asia-Pacific Services Computing Conference 2010.
19. Yogan J K. and Salim N. (2012), 'Automatic Multi Document Summarization Approaches', Journal of Computer Science, Vol.8, No.1, pp. 133–140.

AUTHORS PROFILE

P. Sukumar, Received B.E degree in CSE from VCET (Velalar College of Engineering and Technology), thindal in 2012 and M.E degree from SVCE (Sri Venkateswara College of engineering), Chennai in 2014. He has authored 2 international journals and presented 3 papers in national conferences.

K.S. Gayathri, Received B.E degree in CSE from Madras University in 2001 and M.E degree from Anna University, Chennai. She is doing Ph.D. in the area of Reasoning in Smart Environments. Currently working as an Associate Professor in the Dept. of CSE, Sri Venkateswara College of Engineering **with a teaching experience of 12 years.** Published research papers in one National Conference and two International Conferences. Organized two workshops on Artificial Intelligence.