Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 89 (2016) 404 – 411

Procedia Computer Science

Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016)

ATSSI: Abstractive Text Summarization using Sentiment Infusion

Rupal Bhargava*, Yashvardhan Sharma and Gargi Sharma

Birla Institute of Technology & Science, Pilani, Pilani Campus 333 031, India

Abstract

Text Summarization is condensing of text such that, redundant data are removed and important information is extracted and represented in the shortest way possible. With the explosion of the abundant data present on social media, it has become important to analyze this text for seeking information and use it for the advantage of various applications and people. From past few years, this task of automatic summarization has stirred the interest among communities of Natural Language Processing and Text Mining, especially when it comes to opinion summarization. Opinions play a pivotal role in decision making in the society. Other's opinions and suggestions are the base for an individual or a company while making decisions. In this paper, we propose a graph based technique that generates summaries of redundant opinions and uses sentiment analysis to combine the statements. The summaries thus generated are abstraction based summaries and are well formed to convey the gist of the text.

Keywords: Abstractive Summarization; Condensed Text; Data Redundancy; Sentiment Analysis; Summary; Text Summarization.

1. Introduction

There is a large amount of data on the web which expresses the same opinion over and over again. Summarization of dispensable content, thus is a necessity. While viewing multi-document summaries or the summaries of highly redundant text, extractive summarization would not be of any help as the extractive summaries would be very verbose and biased. Also, the sentences tend to be longer, hence non-essential parts of the sentence also get included. Relevant information is spread across the document and this can't be captured in the extractive summaries. Extractive summaries also face the problem of "dangling" anaphora, implying that sentences that contain pronouns lose meaning when extracted out of context, the resolution of which is presented in Steinberger J. *et al.*[2].

While there has been a lot of work done in the field of extraction based summarization, abstraction based summarization is difficult because of the simple reason that while the computers can statistically select the most important sentence from the text, it is difficult for them to combine important sentences and generate a coherent and concise synopsis. Demand for high quality summary is on the rise whether it is regarding summarization of textual content (for example books etc.) or multimedia content like video transcripts etc. (Ding, Duo, *et al.*[3]).

It has been demonstrated that abstractive summaries perform well than extractive summaries (Carenini, G. *et al.*[4]) whenever documents with a lot of redundant content (e.g. Product reviews, blogs and news articles, etc.). This is

*Corresponding author. Tel.: +91 -9829474312.
E-mail address: rupal.bhargava@pilani.bits_pilani.ac.in

doi:10.1016/j.procs.2016.06.088

because abstractive summaries are compact and present the useful information and are not verbose. But, generating abstract summary is a tougher task than generation of extract summary. Also, it should be noted that single document summarization is somewhat not quite the same as multi document synopsis, since single documents contain lesser data. Thus, a more efficient strategy is required to generate abstractive summaries in case of single documents.

In this paper, method is proposed for compressing and merging information based on word graphs, and then summaries are generated from the resulting sentences. The method assumes no domain knowledge and leverages redundancy in the text. The results show that the summaries generated are agreeable to human compendium and are concise and well formed.

The paper has been split into three sections. In the related works section, previous work on recent abstractive summarization techniques is explained and novelty our approach is stressed with respect to preexisting frameworks for graph based abstractive summarization. In the methodology section, we describe the algorithm that has been used for summarization. In the Results and discussion Section, we present the results of our algorithm and a detailed analysis of the results. In the Conclusion and future work Section, we provide an insight into the possible areas that can be explored in terms of summarization.

2. Related Work

Abstractive techniques in text summarization include rule based approach (Genest, P. E. *et al.*[1]), sentence compression (Knight, K. *et al.*[5], Zajic, D. *et al.*[6], Clarke, J. *et al.*[7]), merging sentence based on their semantics (Liu F. *et al.*[8], Wang D. *et al.*[9]), etc. Graph based algorithms, in particular has been proven to work well on both summarizing texts containing lots of redundant data (Ganesan, K. *et al.*[10], Lloret E., *et al.*[11]), etc.

Sankarasubramaniam Y. *et al.*[23] leverage wikipedia in addition to graph based algorithms to generate extractive summaries. They first map all the sentences to corresponding Wikipedia topic and thus a bipartite graph is obtained where one of vertices represent the wikipedia topics and the other set represent the sentences in the document.[23] then uses an iterative ranking algorithm to find the best candidate sentences in the document.[23] also introduces incremental summarization wherein longer summaries are generated in real-time by simply adding sentences to shorter summaries. Since the summaries generated are extractive, the precision is less when compared to the results of techniques that generate abstractive summaries.

Liu F. *et al.*[8] use the advances in the semantic representation of the text in the form of Abstract Meaning Representation graphs to form summaries. The summarization framework consists of parsing input sentences to form individual AMR graphs, combining the individual AMR graphs to form a summary AMR graph and then generating text from the summary graph. The individual graphs are converted to summary graph using a perceptron model prediction algorithm which predicts with a high accuracy the subgraph that has to be selected for summary generation.

Ganesan K. *et al.*[10] describe an approach that used directed graphs that use the original sentence word order to generate abstractive summaries. Their technique leverages the graphical form of the input text to reduce redundancy. If their algorithm finds two sentences that are collapsible, they use the connectors already present in one of the sentences to be used as the connector for the collapsed sentence. While this technique is effective this still has a drawback since there might be two sentences which are capable of being fused together, but can't be fused because of the absence of a pre-existing connector. Our approach does not face this drawback since we use sentiment analysis to overcome this issue.

Lloret E., *et al.*[11] describes a technique in which they have built a directed weighted word graph where each word text represents a node in the graph and the edge contains the adjacency relation between the words. The weight of the edge is determined by using a combination of their pagerank value and the frequency of the words. To determine important sentences, the first node consists of the first ten words with highest TF-IDF score. Sentence correctness are ensured using the basic rules of grammar like the length a sentence should be greater than 3 words, a sentence must contain a verb and should not end in an article or conjunction. A huge flaw with this methodology is that a lot of important information is lost because of the impositions of grammar on the sentences and the policy of selecting the ten words with highest TF-IDF scores. Furthermore, a lot of redundant sentences will still be present in the summary because the TF-IDF scores will give more importance to them. Our methodology does not face the deficiency that[11] faces because it incorporate the redundancies in our graph structure itself.

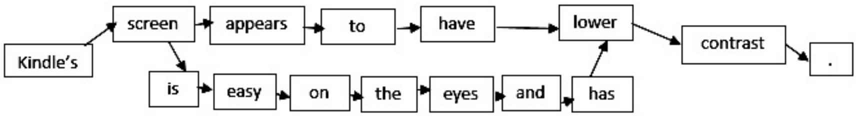

Fig. 1. Graph Capturing Redundancy in the Text.

3. Methodology

3.1 Building the word graph

Graph data structure is used in ATSSI to represent the text. Graphs have been frequently used for abstractive text summarization (Kumar N., *et al.*[12], Liu F., *et al.*[13], etc.) and have shown promising results. The graphs have been used in different forms in the past. Kumar N., *et al.*[12] use graph to represent the bigram relationship between the words in the text. The graphs used by the Liu F., *et al.*[13] incorporate the semantic information in the graphs. Our graph is different from them, as each node represents a word in the text along with the information of the position of the given word in the sentence, and the edges represent the adjacency of the words in the sentence. A document is represented as a directed graph where $V = v_i v_{i+1} \ldots v_n$ is a set of vertices that represents words in the text. Each node stores the information about the POS tag of the word in that node, the position of the word in the sentence and the position of the sentence in the document.

The graph naturally captures the redundancy in the document since words that occur more than once in the text are mapped to the same vertex. Furthermore, the graph construction does not require any domain knowledge. The graph also captures the minor variations in the sentences. For example, Fig. 1.

3.2 Ensuring the sentence correctness

Sentence correctness is ensured using the following set of POS constraints,

- A sentence can contain noun followed by a verb and an adjective or an adjective followed by a noun and a verb or a verb followed by an adjective and a noun or an adverb followed by an adjective and a noun or an adverb followed by a noun.
- The start of the sentence should contain a word which has the average position in all sentences lower than a threshold that we call Start Node. This threshold is enforced to corroborate that the sentences occurring in the summary do not start with words that occur somewhere in the middle of a sentence.
- The sentence should not end in a conjunction like but, yet, etc.

3.3 Getting abstractive summaries

3.3.1 Scoring of paths

The paths are then scored based on the redundancy of the overlapping sentences. This redundancy can be calculated using the intersection of the position of the words in the sentences (P) such that the difference between the positions is no greater than a threshold, P. This redundancy helps us in deciding the number of sentences discussing something similar at each point in the path.

The scores can simply be based on the calculation of the overlap or can include the length of the path as well because if the path is longer, higher redundancy will be of more worth to us than in a shorter path since longer paths provide more coverage.

3.3.2 Fusing sentiments

A node is considered to fuse sentences if its POS is a verb. If a vertex V is being considered as a node that can be used to fuse sentences, then we traverse the previous vertices in the path currently being considered to look for a

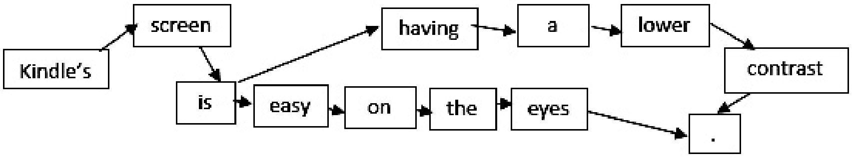

Fig. 2. Example Sentences that can be Fused Together.

Table 1. Example of Sentence and Corresponding Sentiment.

Main Sentence		Sentiment
Kindle's screen **is**	Having a lower contrast	+ve
	Is easy on the eyes	+ve
	Connector	and

connector. An alternative approach is to calculate the sentiment of both the sentences to be fused and to look for a connector that can be accurately used. This sentiment is calculated using SentiWordNet 3.0 (Baccianella, S., *et al.*[15]). Once the sentiment has been calculated, we choose the connector from a preexisting list. For example, if the sentiments of the two sentences are contradictory, we use but, if there are both positive we use and or, etc.

3.3.3 Summarization

Once all the paths are scored as well as the sentences have been fused, we rank the sentences in descending order of their scores. We remove duplicate sentences from our summary using Jaccard index for similarity measure. Then the remaining top most S (number of sentences in the summary specified by the user) sentences are chosen for the summary.

3.4 Pseudocode

a. Generate the graph from the text input such that nodes will contain the information about
   - The position of the word in the sentence,
   - The position of the sentence in the document
   - And the POS tag of the word.
b. For all the nodes in the graph, if the node satisfies the constraint of being lesser than Start Node, then we start traversing the graph.
c. While traversing the graph, if the path overlap is greater than P, we check if the current node is a valid end node and the current sentence is a valid sentence, if it is, we add it to the list of candidate summaries, else we discard it.
d. For all the neighbours of the current node
   - Calculate the redundancy.
   - Check if the node can be used to fuse a sentence.
   - If yes, then calculate the sentiment of the anchor statements, and choose the connector accordingly from the pre-existing conjunction list. If the node cannot be used to fuse sentences, we call the function to traverse the graph again.
e. Call the function to traverse the graph again from all the neighbours of the current node to find the further nodes of the sentence.
f. The new score is computed and the duplicate sentences are removed from the fused sentences. The resulting fused sentence and its final score are then added to the original list of candidate summaries.

g. Once all paths have been explored, duplicates are removed. The rest of the sentences are sorted in descending order of their path scores. The best S candidates are 'picked' for the final summary.

4. Results & Discussion

4.1 Dataset

Two datasets are used for the evaluation:

- National Institute of Science and Technology (NIST) organizes a conference called Document Understanding Conference (DUC) every year. The first dataset comprises of 50 documents from the DUC 2002[18] corpus which have been randomly selected. The documents contain about 500 words on an average. The dataset contains about 500 news articles in English along with gold summaries for each article. The gold summaries have also been provided for the corresponding documents and are about 100 words on an average.
- Whereas second Dataset[10] contains 51 documents pertaining to a single query, for example, Amazon Kindle: buttons, Holiday Inn, Chicago: staff, etc. There are about 100 redundant, unordered sentences in the document for every query. There are 4 peer summaries corresponding to each of these 51 documents.

4.2 Experimental Setup & Results

ROUGE metric was introduced by Lin *et al.* (2003)[19] and has been adopted by the DUC and leading conferences on Natural Language Processing. ROUGE calculates the overlap between the candidate summaries and the reference summaries and it has been found that correlation of ROUGE-1 and ROUGE-2 is the most with human summaries (Lin, C.Y. *et al.*[21]). ROUGE-N is a recall measure that computes the number of matches between the candidate summaries and the reference summaries. The formula to calculate the ROUGE scores[19] is given as:

$$\text{ROUGE-N} = \frac{\sum_{S\in(\text{Reference Summaries})}\sum_{\text{gram}_n\in S}\text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S\in(\text{Reference Summaries})}\sum_{\text{gram}_n\in S}\text{Count}\,(\text{gram}_n)}$$

where Match is the maximum number of N-grams that occur in the reference summaries and the candidate summary. Count is the number of N-grams in the reference summaries. The precision, recall and F-measure[19] is calculated as follows:

$$\text{Precision} = \frac{\text{Match(Sentence)}}{\text{Match\_Candidate(Sentence)}}$$

$$\text{Recall} = \frac{\text{Match(Sentence)}}{\text{Match}_{\text{Best\_Candidate}}(\text{Sentence})}$$

$$F - \text{Measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

where Match_Candidate is the number of sentences of present in the candidate summary. $\text{Match}_{\text{Best\_Candidate}}$ total number of sentences in the best sentences summary.

In our experiments, we have used ROUGE-1 and ROUGE-2 for comparison with Baseline 1 and ROUGE-1 for comparison with Baseline 2.

4.3 Performance analysis

Results are being compared by two baselines on two different datasets, apart from the comparison with human summaries. Baseline1 is defined by the algorithm implemented by Ganesan *et al.*[10] whereas Baseline 2 is described by algorithm mentioned in[11]. Baseline 1 and Baseline 2 have been chosen for comparison since they have used graph based algorithms for summarization and our algorithm fills the gap that[10] and[11] are not able to solve.

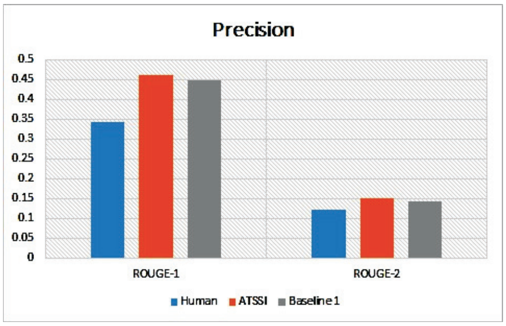

Fig. 3. Evaluated Precision on Dataset[10] with Human Summary and Baseline 1 vs ATSSI.

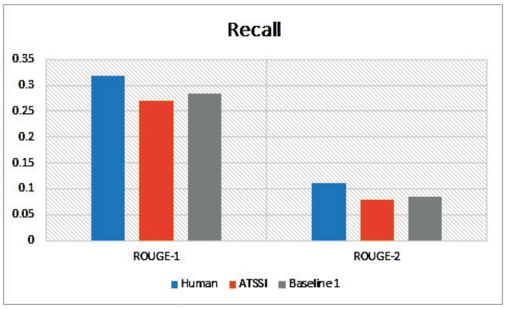

Fig. 4. Evaluated Recall on Dataset[10] with Human Summary and Baseline 1 vs ATSSI.

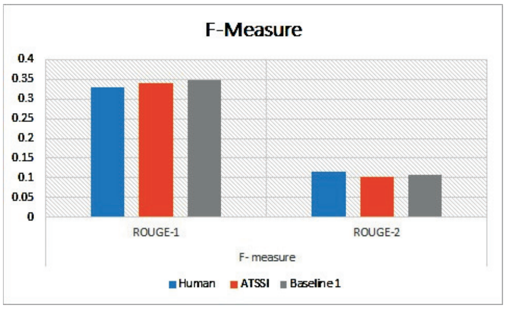

Fig. 5. Evaluated F-measure on Dataset[10] with Human Summary and Baseline 1 vs ATSSI.

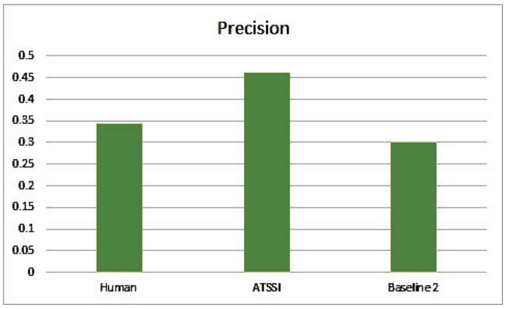

Fig. 6. Evaluated Precision on DUC-2002 Dataset with Human Summary and Baseline 2 vs ATSSI.

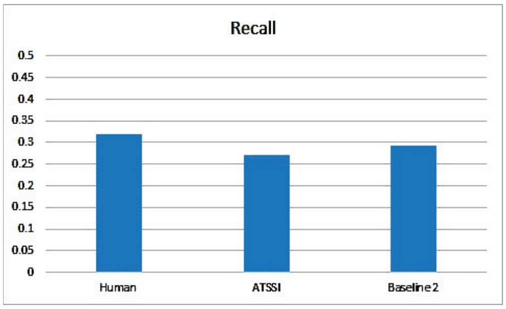

Fig. 7. Evaluated Recall on DUC-2002 Dataset with Human Summary and Baseline 2 vs ATSSI.

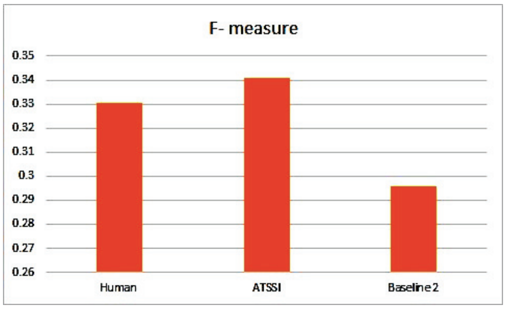

Fig. 8. Evaluated F-measure on DUC-2002 Dataset with Human Summary and Baseline 2 vs Proposed System.

Figure 3 shows that ATSSI has higher precision over the baseline 1, this is because we have overcome the demerit of the approach as stated by[10], that it can only connect the sentences if there is a pre-existing connector. Since the dataset used by[10] already had redundant data with connectors, ATSSI is only showing a marginal difference in precision. Recall on Dataset[10] is resulting in low recall as shown in Fig. 4 because there is a high presence of redundant information on dataset which is leading to infusing of maximum number sentiments in a particular sentence, which in turn results in low frequency of sentences.

When comparing to Baseline 2[10], our algorithm outperforms by 13 percent (Fig. 8), since Baseline 2[10] describes rigid rules for ensuring sentence correctness and has no provision for fusing sentences. ATSSI outperforms

Baseline 2[10] by a huge margin in precision comparison since ATSSI incorporates sentiment infusion and a provision for removing redundancy. Also recall of ATSSI is marginally low as that of Baseline 2 for similar reasons mentioned for baseline 1.

Finally, it is worth noting that generating summaries that are purely abstractive in nature is an onerous task, as shown by F. Liu *et al.*[20] where F-measure values are in the range 13% to 18%.

5. Conclusions and Future Work

Evaluation using the DUC 2002 dataset[18] and on the dataset described[10] outperforms the abstractive summarization algorithms described by Lloret E. *et al.*[11] and Ganesan K., *et al.*[10]. Our approach is able to leverage the sentence word order to form coherent sentences, and hence our summaries are concise, we are able to communicate the information in conjunction with the ability to remove the redundancy from the input text. No domain knowledge is required for proposed algorithm to work. It is adaptable to different types of content as well. Also, because we do not use any semantic information related to the sentences of our input document, we are not able to combine sentences that are semantically related but not related syntactically. To address this shortcoming, abstractive summarization system based semantic representations, like parse trees, Abstract Meaning Representation (AMR) graphs can be used. Semantic representations such as the AMR (Banarescu *et al.*[17]) have shown to perform better than regular graph based algorithms (Liu F. *et al.*[8]). This could be a significant inclusion that can be done in future.

Also, it would be interesting to experiment how the existing extractive technique would work with our abstractive technique. Summarization systems that use both extractive and abstractive summarization techniques, for example the technique used by Kumar, N. *et al.*[22], has shown to perform better than pure abstractive systems. The technique described by Lloret E., *et al.* that implements abstractive summarization on the text that is being generated by extracting techniques, outperforms baseline extractive summary by a huge margin. In future work, we can use the pre-existing extractive techniques to improve recall and abstractive techniques can then be applied to improve precision of the summary.

Acknowledgements

We would like to thanks, Department of Science and Technology, New Delhi, India, for providing financial assistance under major research project scheme [S. No: SERB/F/1266/2013-2014]. Authors are also greatful to the authorities of Birla Institute of Technology and Science, Pilani, for providing basic infrastructure facilities during the preparation of the paper.

References

[1] P. E. Genest and G. Lapalme, Fully Abstractive Approach to Guided Summarization, In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics*, vol. 2, pp. 354–358, July (2012).

[2] J. Steinberger, M. Poesio, M. A. Kabadjov and K. Ježek, Two Uses of Anaphora Resolution in Summarization, *Information Processing & Management*, vol. 43(6), pp. 1663–1680, (2007).

[3] D. Ding, F. Metze, S. Rawat, P. F. Schulam, S. Burger, E. Younessian, L. Bao, M. G. Christel and A. Hauptmann, Beyond Audio and Video Retrieval: Towards Multimedia Summarization, In *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval, ACM*, p. 2, June (2012).

[4] Carenini, Giuseppe and Jackie Chi Kit Cheung, Extractive vs. NLG-Based Abstractive Summarization of Evaluative Text: The Effect of Corpus Controversiality, *Proceedings of the Fifth International Natural Language Generation Conference, Association for Computational Linguistics*, (2008).

[5] J. Clarke and M. Lapata, Models for Sentence Compression: A Comparison Across Domains, Training Requirements and Evaluation Measures, In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pp. 377–384, July (2006).

[6] K. Knight and D. Marcu, Summarization Beyond Sentence Extraction: A Probabilistic Approach to Sentence Compression, *Artificial Intelligence*, vol. 139(1), pp. 91–107, (2002).

[7] D. Zajic, B. J. Dorr, J. Lin and R. Schwartz, Multi-Candidate Reduction: Sentence Compression as a Tool for Document Summarization Tasks, *Information Processing & Management*, vol. 43(6), pp. 1549–1570, (2007).

[8] F. Liu, J. Flanigan, S. Thomson, N. Sadeh and N. A. Smith, Toward Abstractive Summarization Using Semantic Representations, (2015).

[9] D. Wang, T. Li, S. Zhu and C. Ding, Multi-Document Summarization Via Sentence-Level Semantic Analysis and Symmetric Matrix Factorization, In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM*, pp. 307–314, July (2008).
[10] K. Ganesan, C. Zhai and J. Han, Opinosis: A Graph-Based Approach to Abstractive Summarization of Highly Redundant Opinions, In *Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics*, pp. 340–348, August (2010).
[11] E. Lloret and M. Palomar, Analyzing the Use of Word Graphs for Abstractive Text Summarization, In *Proceedings of the First International Conference on Advances in Information Mining and Management*, Barcelona, Spain, pp. 61–6, (2011).
[12] N. Kumar, K. Srinathan and V. Varma, A Knowledge Induced Graph-Theoretical Model for Extract and Abstract Single Document Summarization, In *Computational Linguistics and Intelligent Text Processing, Springer Berlin Heidelberg*, pp. 408–423, (2013).
[13] F. Liu, J. Flanigan, S. Thomson, N. Sadeh and N. A. Smith, Toward Abstractive Summarization Using Semantic Representations, (2015).
[14] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard and D. McClosky, The Stanford CoreNLP Natural Language Processing Toolkit, In *ACL (System Demonstrations)*, pp. 55–60, June (2014).
[15] S. Baccianella, A. Esuli and F. Sebastiani, SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining, In *LREC*, vol. 10, pp. 2200–2204, May (2010).
[16] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar and S. Fienberg, Adaptive Name Matching in Information Integration, *IEEE Intelligent Systems*, vol. 5, pp. 16–23, (2003).
[17] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer and N. Schneider, Abstract Meaning Representation (AMR) 1.0 Specification, In *Parsing on Freebase from Question-Answer Pairs, In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle: ACL, pp. 1533–1544, (2012).
[18] Document Understanding Conference (DUC). ¡http://duc.nist.gov¿.
[19] C. Y. Lin, ROUGE: A Package for Automatic Evaluation of Summaries, In *Proceedings of ACL Text Summarization Workshop*, pp. 74–81, (2004).
[20] F. Liu and Y. Liu, From Extractive to Abstractive Meeting Summaries: Can it be Done by Sentence Compression? In *Proceedings of the ACL-IJCNLP Conference Short Papers, ACL*, pp. 261–264, (2009).
[21] C. Y. Lin and E. Hovy, Automatic Evaluation of Summaries Using n-Gram Co-Occurrence Statistics, In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Association for Computational Linguistics*, vol. 1, pp. 71–78, May (2003).
[22] N. Kumar, K. Srinathan and V. Varma, A Knowledge Induced Graph-Theoretical Model for Extract and Abstract Single Document Summarization, In *Computational Linguistics and Intelligent Text Processing*, Springer Berlin Heidelberg, pp. 408–423, (2013).
[23] Y. Sankarasubramaniam, K. Ramanathan and S. Ghosh, Text Summarization Using Wikipedia, *Information Processing & Management*, vol. 50(3), pp. 443–461, (2014).