Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2017, 4 (6): 410-427

Research Article ISSN: 2394 - 658X

Automatic *Keyword* Extraction for Text Summarization in Multi-document e-Newspapers Articles

Santosh Kumar Bharti, Korra Sathya Babu, and Anima Pradhan

Department of Computer Science Engineering, National Institute of Technology, Rourkela, India
sbharti1984@gmail.com

ABSTRACT

Summarization is the way towards lessening the content of a text file to make it brief that holds all the critical purposes in the content of original text file. In the process of extractive summarization, one extracts only those sentences which are the most relevant sentences in the text document and that conveys the moral of the content. The extractive summarization techniques usually revolve around the idea of discovering most relevant and frequent keywords and then extract the sentences based on those keywords. Manual extraction or explanation of relevant keywords are a dreary procedure overflowing with errors including loads of manual exertion and time. In this paper, we proposed a hybrid approach to extract keyword automatically for multi-document text summarization in e-newspaper articles. The performance of the proposed approach is compared with three additional keyword extraction techniques namely, term frequency-inverse document frequency (TF-IDF), term frequency-adaptive inverse document frequency (TF-AIDF), and a number of false alarm (NFA) for automatic keyword extraction and summarization in e-newspapers articles for better analysis. Finally, we showed that our proposed techniques had been outperformed over other techniques for automatic keyword extraction and summarization.

Keywords: e-Newspapers, Keyword Extraction, Multi-document, NLP, NFA, Text Summarization, TF-AIDF

INTRODUCTION

In the era of internet, plethora of e-Newspapers are freely available for readers such as Times of India, The Hindu, Hindustan Times, Washington Times, The New York Times, The Times, etc. People find a lot of information every day in these newspapers and extracting relevant information from all the articles is a tedious job for the individuals. There is a need for an automated system that can extract only relevant information from these news sources. To achieve this, one need to mine the text from newspapers. Text mining is the process of extracting large quantities of text to derive high quality information. Text mining deploys some of the techniques of Natural Language Processing (NLP) such as part-of-speech (POS) tagging, parsing, N-grams, tokenization, etc., to perform the text analysis. It includes tasks like automatic keyword extraction and text summarization.

Automatic keyword extraction is the process of selecting words and phrases from the text document that can at best project the core sentiment of the document without any human intervention depending on the model [1]. The target of automatic keyword extraction is the application of the power and speed of current computation abilities to the problem of access and recovery, stressing upon information organization without the added costs of human annotators.

Summarization is a process where the most salient features of a text are extracted and compiled into a short abstract of the original document [2]. According to Mani and Maybury [3], text summarization is the process of distilling the most important information from a text to produce an abridged version for a particular task and user. Summaries are usually around 17% of the original text and yet contain everything that could have been learned from reading the original article [4]. In the wake of big data analysis, summarization is an efficient and powerful technique to give a glimpse of the whole data. The text summarization can be achieved in two ways namely, abstractive summary and extractive summary. The abstractive summary is a topic under tremendous research; however, no standard algorithm has been achieved yet. These summaries are derived from learning what was expressed in the article and then converting it into a form expressed by the computer. It resembles how a human would sum-

marize an article after reading it. Whereas, extractive summary extract details from the original article itself and present it to the reader. In this paper, we focus on extractive summarization method. It can be observed that extractive summarization relies heavily on keyword extraction. Therefore, we focused our attention on the integration between them.

Here, an algorithm is proposed for automatic keyword extraction for text summarization. It is capable of handling the limitation of existing techniques like adding a stop list to extract important keyword which might include words that are essential to the document and might lead some relevant words to lose its significance. Also, the rarity of the word in other texts was also considered. The word which is highly frequent in the article under review and is hardly found in other documents gets very high scores so that only the words pertinent to this article gets chosen as keywords. The proposed algorithm follows a hybrid approach of machine learning and statistical method. A Hidden Markov Model (HMM) based POS tagger [5] is used to identify POS information of an article and then a statistical method is used to extract keywords. The algorithm for automatic keyword extraction uses a learned probability distribution to assign scores to each word. The keywords for the article under consideration are determined based on these scores and are used to summarize the article. The summarization algorithm accordingly selects sentences to form the required summary. This algorithm applies to multiple articles at a time for keyword extraction and summarization. It extracts the keyword from all the articles and appends it to a single file. It also eliminates the redundant keywords in the final output file.

In this paper, we proposed an algorithm to summarized multi-documents text to single document and it compared the performance with three additional techniques for text summarization namely, TF-IDF, TF-AIDF, and NFA for better analysis. The related work for automatic keyword extraction followed by text summarization is discussed in next section. The preliminaries used in this paper is also discussed along with other three techniques (NFA, TF-IDF, and TF- AIDF) for keyword detection and extraction. Finally, the performance analysis of the proposed schemes is analysed along with the conclusion.

RELATED WORK

Keyword extraction is the essential phase to perform text summarization. Therefore, this section presents a literature survey on automatic keyword extraction followed by text summarization [7-15].

Automatic Keyword Extraction

On the premise of past work done towards automatic keyword extraction from the text for its summarization, extraction systems can be classified into four classes, namely, simple statistical approach, linguistics approach, machine learning approach, and hybrid approaches [1] as shown in Fig. 1.

Simple Statistical Approach

These strategies are rough, simplistic and have a tendency to have no training sets. They concentrate on statistics got from non-linguistic features of the document, for example, the position of a word inside the document, the term frequency, and inverse document frequency. These insights are later used to build up a list of keywords. Cohen [16], utilized n-gram statistical data to discover the keyword inside the document automatically. Other techniques inside this class incorporate word frequency, term frequency (TF) [17] or term frequency-inverse document frequency (TF-IDF) [18], word co-occurrences [19], and PAT-tree [20]. The most essential of them is term frequency. In these strategies, the frequency of occurrence is the main criteria that choose whether a word is a keyword or not. It is extremely unrefined and tends to give very unseemly results. An improvement of this strategy is the TF-IDF, which also takes the frequency of occurrence of a word as the model to choose a keyword or not. Similarly, word co-occurrence methods manage statistical information about the number of times a word has happened and the number of times it has happened with another word. This statistical information is then used to compute support and confidence of the words. Apriori technique is then used to infer the keywords.

Linguistics Approach

This approach utilizes the linguistic features of the words for keyword detection and extraction in text documents. It incorporates the lexical analysis [21], syntactic analysis [22], discourse analysis [23], etc. The resources used for lexical analysis are an electronic dictionary, tree tagger, WordNet, n-grams, POS pattern, etc. Similarly, noun phrase (NP), chunks (Parsing) are used as resources for syntactic analysis.

Machine Learning Approach

Keyword extraction can also be seen as a learning problem. This approach requires manually annotated training data and training models. Hidden Markov model [24], support vector machine (SVM) [25], naive Bayes (NB) [26], bagging [22], etc. are commonly used training models in these approaches. In the second phase, the document whose keywords are to be extracted is given as inputs to the model, which then extracts the keywords that best fit

the model's training. One of the most famous algorithms in this approach is the keyword extraction algorithm (KEA) [27]. In this approach, the article is first converted into a graph where each word is treated as a node, and whenever two words appear in the same sentence, the nodes are connected with an edge for each time they appear together. Then the number of edges connecting the vertices are converted into scores and are clustered accordingly. The cluster heads are treated as keywords. Bayesian algorithms use the Bayes classifier to classify the word into two categories: keyword or not a keyword depending on how it is trained. GenEx [28] is another tool in this approach.

Hybrid Approach

These approaches combine the above two methods or use heuristics, such as position, length, layout feature of the words, HTML tags around the words, etc. [29]. These algorithms are designed to take the best features from above mentioned approaches.

Based on the classification shown in Fig. 1, we observed the various parameters for automatic keyword extraction as shown in Table - 1. Based on those parameters, we bring a consolidated summary of previous studies on automatic keyword extraction and is shown in Table - 2. It discusses the approaches that are used for keyword extraction, various datasets in different domains in which experiment performed.

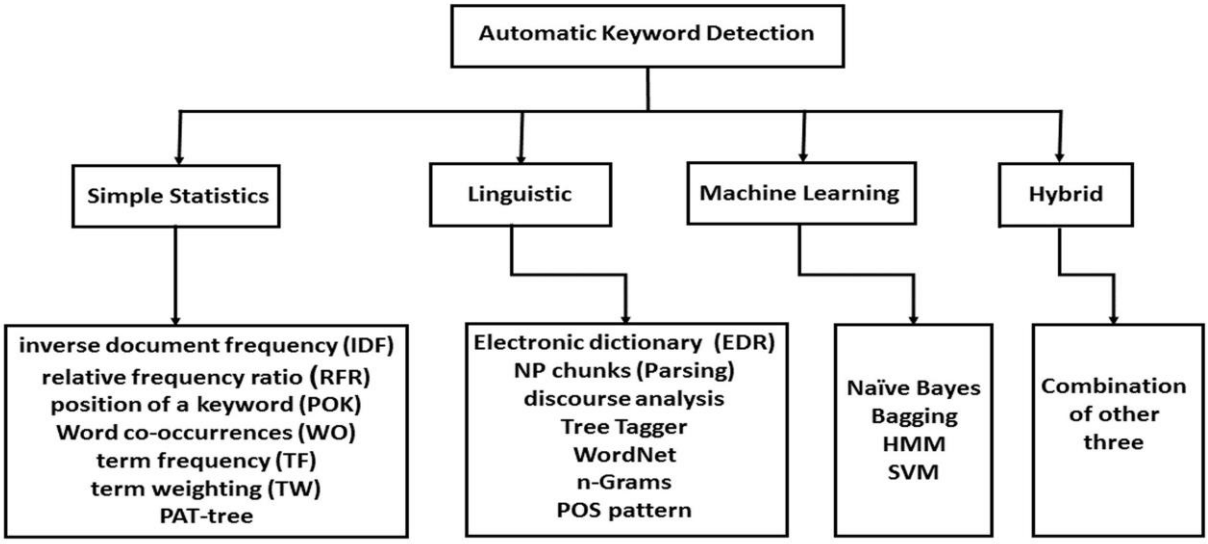

Fig.1 Classification of automatic keyword extraction on the basis of approaches used in existing literature

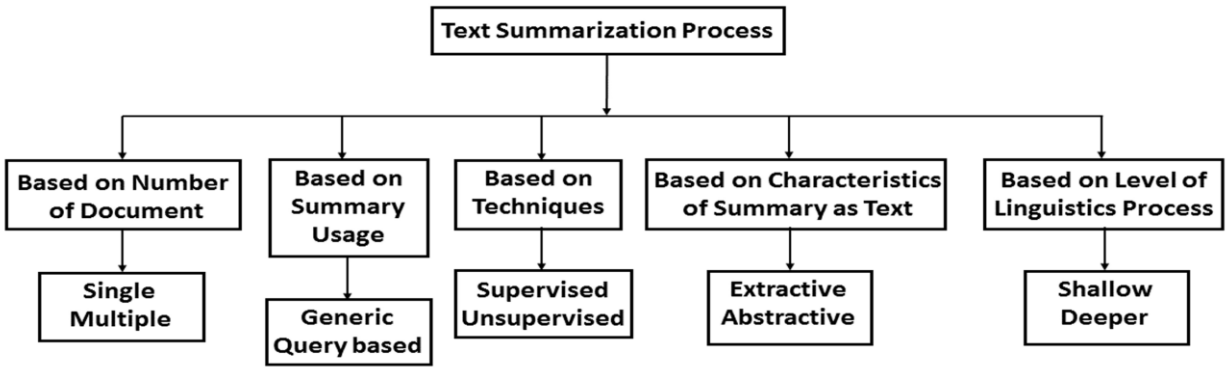

Fig. 2 Characterization of the text summarization process

Table - 1 Types of Approach and Domains used in Keyword Extraction

Types of Approach		Types of Domain	
T1	Simple Statistics (SS)	D1	Radio News (RN)
T2	Linguistics (L)	D2	Journal Articles (JA)
T3	Machine Learning (ML)	D3	Newspaper Articles (NA)
T4	Hybrid (H)	D4	Technical Reports (TR)
		D5	Transcription Dialogues (TD)
		D6	Encyclopedia Article (EA)
		D7	Web Pages (WP)

Table - 2 Previous Studies on Automatic Keyword Extraction

Study	Types of Approach				Domain Types						
	T1	T2	T3	T4	D1	D2	D3	D4	D5	D6	D7
Dennis *et al*[29], 1967		√					√				
Salton *et al*[30], 1991		√								√	
Cohen *et al*[15], 1995	√					√					
Chien *et al*[19], 1997	√					√					
Salton *et al*[22], 1997		√								√	
Ohsawa *et al*[31], 1998	√					√					
Hovy *et al*[2], 1998		√					√				
Fukumoto *et al*[32], 1998	√				√		√			√	
Mani *et al*[3], 1999		√									
Witten *et al*[26], 1999			√					√			
Frank *et al*[25], 1999			√			√		√			
Barzilay *et al*[20], 1999		√									
Turney *et al*[27], 1999			√			√					
Conroy *et al*[23], 2001			√					√			
Humphreys *et al*[28], 2002		√		√							√
Hulth *et al*[21], 2003		√	√			√		√			√
Ramos *et al*[17], 2003	√										
Matsuo *et al*[18], 2004	√										
Erkan *et al*[4], 2004		√									
Van *et al*[6], 2004		√							√		
Mihalcea *et al*[33], 2004			√				√				
Zhang *et al*[24], 2006			√			√					
Ercan *et al*[8], 2007			√			√					
Litvak *et al*[9], 2008			√								√
Zhang *et al*[1], 2008			√			√					
Thomas *et al*[5], 2016	√	√		√			√				

Text Summarization Process

Based on the literature, text summarization process can be characterized into five types, namely, based on the number of the document, based on summary usage, based on techniques, based on characteristics of summary as text and based on levels of linguistics process [1] as shown Fig. 2.

Single Document Text Summarization

In single document text summarization, it takes a single document as an input to perform summarization and produce a single output document [6, 35-38]. Thomas *et al* [6] designed a system for automatic keyword extraction for text summarization in single document e-Newspaper article. Marcu *et al* [36] developed a discourse-based summarizer that determines adequacy for summarizing texts for discourse-based methods in the domain of single news articles.

Multiple Document Text Summarization

In multiple documents text summarization, it takes numerous documents as an input to perform summarization and deliver a single output document [15, 39-45]. Mirroshandel *et al* [45] presents two different algorithms towards temporal relation based keyword extraction and text summarization in multi-document. The first algorithm was a weakly supervised machine learning approach for classification of temporal relations between events and the second algorithm was expectation maximization (EM) based unsupervised learning approach for temporal relation extraction. Min *et al* [41] used the information which is common to document sets belonging to a common category to improve the quality of automatically extracted content in multi- document summaries.

Query-based Text Summarization

In this summarization technique, a particular portion is utilized to extract the essential keyword from input document to make the summary of corresponding document [12, 38, 49, 50, 51, 52]. Fisher *et al* [51] developed a query-based summarization system that uses a log-linear model to classify each word in a sentence. It exploits the property of sentence ranking methods in which they consider neural query ranking and query-focused ranking. Dong *et al* [52] developed a query-based summarization that uses document ranking, time-sensitive queries and ranks regency sensitive queries as the features for text summarization.

Extractive Text Summarization

In this procedure, summarizer discovers more critical information (either words or sentences) from input document to make the summary of the corresponding document [2, 40, 41, 46, 53, 36, 54, 55, 56, 76]. Thomas *et al* [6] de-

signed a hybrid model based extractive summarizer using machine learning and simple statistical method for keyword extraction from e-Newspaper article. Min *et al.* [41] used freely available, open-source extractive summarization system, called SWING to summarize the text in multi-document. They used information which is common to document sets belonging to a common category as a feature and encapsulated the concept of category-specific importance (CSI). They showed that CSI is a valuable metric to aid sentence selection in extractive summarization tasks. Marcu *et al.* [36] developed a discourse-based extractive summarizer that uses the rhetorical parsing algorithm to determine discourse structure of the text of given input, determine partial ordering on the elementary and parenthetical units of the text. Erkan *et al.* [66] developed an extractive summarization environment. It consists of three steps: feature extractor, the feature vector, and reranker. Features are Centroid, Position, Length Cutoff, SimWithFirst, LexPageRank, and QueryPhraseMatch. Alguliev *et al.* [40] developed an unsupervised learning based extractive summarizer that optimizes three properties: relevance, redundancy, and length. It split documents into sentences and select salient sentences from the document. Aramaki *et al.* [76] destined a supervised learning based extractive text summarizer that identifies the negative event and it also investigates what kind of information is helpful for negative event identification. An SVM classifier is used to distinguish negative events from other events.

Abstractive Text Summarization

In this procedure, a machine needs to comprehend the idea of all the input documents and then deliver summary with its particular sentences [35, 53, 57-59]. Brandow *et al.* [57] developed an abstractive summarization system that analyses the statistical corpus and extracts the signature words from the corpus. Then it assigns the weight for all the signature words. Based on the extracted signature words, they assign the weight to the sentences and select few top weighted sentences as the summary. Daume *et al.* [38] developed an abstractive summarization system that maps all the documents into database-like representation. Further, it classifies into four categories: a single person, single event, multiple event, and natural disaster. It generates a short headline using a set of predefined templates. It generates summaries by extracting sentences from the database.

Supervised Learning based Text Summarization

This type of learning techniques used labelled dataset for training [6, 13, 39, 41, 51, 74, 76]. Thomas *et al.* [6] designed a system for automatic keyword extraction for text summarization using hidden Markov model. The learning process was supervised, it used human annotated keyword set to train the model. Mirroshandel *et al.* [45] used a set of labelled dataset to train the system for the classification of temporal relations between events. Aramaki *et al.* [76] destined a supervised learning based extractive text summarizer that identifies the negative event and also investigates what kind of information is helpful for negative event identification. An SVM classifier is used to distinguish negative events from other events.

Unsupervised Learning based Text Summarization

In this technique, there are no predefined guidelines available at the time of training [14, 39, 40, 45, 66]. Mirroshandel *et al.* [45] proposed a method for temporal relation extraction, based on the Expectation-Maximization (EM) algorithm. Within EM, they used different techniques such as a greedy best-first search and integer linear programming for temporal inconsistency removal. The EM-based approach was a fully unsupervised temporal relation based extraction for text summarization. Alguliev *et al.* [40] developed an unsupervised learning based extractive summarizer that optimizes three properties: relevance, redundancy, and length. It split documents into sentences and select salient sentences from the document.

Based on the characterization of text summarization as shown in Figure 2, we bring a consolidated summary of previous studies in text summarization and is shown in Table 3. It discusses the approaches that are used for text summarization; experiment performed using single or multiple documents, types of summary usage, characteristics of the summary and metrics used. The details of the parameters are given in Table 4.

Table - 4 Approaches, Documents, Summary Usage, Characteristics of Summary and Metrics Used in Text Summarization

Types of Approach (TOA)		Document Type (DT)		Types of Summary Usage (TOSU)	
A1	Supervised	D1	Single document	S1	Generic
A2	Unsupervised	D2	Multiple document	S2	Query based
A3	Others				
Characteristics of Summary (COS)		Metrics			
C1	Extractive	M1		ROUGE 1, ROUGE 2, ROUGE L, ROUGE W, ROUGE SU4	
C2	Abstractive	M2		Precision, Recall, F-measure	

Table - 3 Previous Studies on Automatic Text Summarization

Study	TOA					DT		TOSU		COS		DBU				MAT	
	A1	A2	A3	A4	A5	D1	D2	S1	S2	C1	C2	X1	X2	X3	X4	M1	M2
Pollock *et al.* [34], 1975					√	√					√						
Brandow *et al.* [56], 1995					√		√				√						√
Hovy *et al.* [2], 1998					√		√			√							√
Aone *et al.* [45], 1998					√	√		√		√							√
Radev *et al.* [52], 1998					√		√			√	√						
Marcu *et al.* [35], 1999					√	√				√							√
Barzilay *et al.* [57], 1999					√		√				√						√
Chen *et al.* [53], 2000				√			√			√							√
Radev *et al.* [54], 2001a					√		√			√							
Radev *et al.* [55], 2001b				√			√			√							
Radev *et al.* [46], 2001c	√						√	√		√							
Lin *et al.* [59], 2002				√			√			√							√
McKeown *et al.* [60], 2002				√			√			√							√
Daumé *et al.* [58], 2002					√		√			√	√						
Harabagiu *et al.* [36], 2002					√	√	√			√	√						√
Saggion *et al.* [61], 2002					√		√				√						√
Saggion *et al.* [37], 2003					√	√		√	√	√							√
Chali *et al.* [62], 2003				√		√	√			√							
Copeck *et al.* [63], 2003					√	√				√							
Alfonseca *et al.* [64], 2003					√	√				√						√	
Erkan *et al.* [65], 2004				√			√			√						√	
Filatova *et al.* [10], 2004					√		√			√						√	
Nobata *et al.* [66], 2004					√		√			√						√	
Conroy *et al.* [11], 2005	√						√		√	√						√	
Farzindar *et al.* [48], 2005	√						√		√	√						√	
Witte *et al.* [67], 2005					√		√			√						√	
Witte *et al.* [68], 2006				√			√			√						√	
He *et al.* [69], 2006					√		√			√						√	
Witte *et al.* [13], 2007				√			√			√						√	
Fuentes *et al.* [49], 2007	√					√	√		√	√							
Dunlavy *et al.* [70], 2007					√	√	√			√						√	
Gotti *et al.* [71], 2007					√		√			√						√	
Svore *et al.* [72], 2007	√						√			√						√	
Schilder *et al.* [12], 2008	√						√			√						√	
Liu *et al*[73], 2008	√						√			√						√	
Zhang *et al.* [74], 2008					√		√			√						√	
Aramaki *et al.* [75], 2009	√						√			√							√
Fisher *et al.* [50], 2009	√						√		√	√						√	
Hachey *et al.* [47], 2009					√		√	√		√						√	
Wei *et al.* [76], 2010					√		√			√							√
Dong *et al.* [51], 2010					√		√		√								
Shi *et al.* [38], 2010					√		√										
Park *et al.* [87], 2010				√			√		√	√						√	
Archambault *et al.* [14], 2011					√		√										
Genest *et al.* [41], 2011							√			√							
Tsarev *et al.* [42], 2011	√			√			√			√						√	
Alguliev *et al.* [39], 2011				√		√	√			√						√	
Mirroshandel *et al.* [44], 2012	√			√			√			√							
Min *et al.* [40], 2012	√						√	√		√						√	
De Melo *et al.* [43], 2012					√		√			√							√
Thomas *et al.* [5], 2012	√					√				√							√

PRILIMINARIES

The focus of our work lies in enabling a user to search for keywords from within a text file and get the summary of the entire document using those keywords. In this paper, POS tagging, anaphora and cataphora resolution are used as preliminaries and are explained as follows.

POS Tagging

It is a procedure of taking a word from the corpus as input and assigns corresponding part-of-speech to every word as output by its definition and context i.e. association with contiguous and related words in a phrase, sentence, or passage. In this paper, an HMM-based POS tagger is deployed to identify correct POS information of words in given sentences or phrases. For example, POS tag for the sentence, 'Love has no finite coverage' is love—NN, has—VBZ, no—DT, finite—JJ, coverage—NN. In this work, Penn Treebank tag set notations [78] are followed. It is a brown corpus style of tagging having 44 tags. Such as JJ-adjective, NN-noun, RB-adverb, VB-verb, and UH-interjection, etc.

Hidden Markov Model

Hidden Markov model is a supervised machine learning classifier. The automatic labelling of words to their parts of speech is known as POS tagging. POS tagging is a supervised learning problem. Therefore, HMM [5] is used to analyse the POS tags for a given text. With the help of HMM, we can resolve the ambiguity problem in POS tagging such as 'I love writing' and 'Love has no finite coverage.' In the sentence 'I love writing', 'love' act as a verb (VB) while in 'Love has no finite coverage', 'love' act as a noun (NN). HMM analyses this ambiguity correctly and gives accurate POS information as an output.

Anaphora and Cataphora Resolution

The word anaphora is derived from an ancient Greek word which means 'the act of carrying back upstream'. It involves two parts; anaphor: the part which is pointing (reference) and the antecedent: the part which is being pointed to. For example: 'Sachin isn't out yet, but he should be any minute'. In this example, Sachin is the antecedent, and he is the anaphor. Anaphora resolution deals with matching the anaphor to the antecedent. Hence converting the above sentence into Sachin isn't out yet but Sachin should be any minute.

Similarly, cataphora too is derived from the ancient Greek word, meaning 'the act of carrying forward'. It involves replacing an anaphor by an antecedent that occurs later in the text and not before as was the case in anaphora resolution. For example: 'It is tough, it is irritating, and it is annoying. I hate writing the thesis'. Cataphora resolution would resolve the above sentence into 'writing the thesis is tough, writing the thesis is irritating and writing the thesis is annoying. I hate writing the thesis'. This paper followed Lappin *et al* [79] algorithm for anaphora and cataphora resolution in English text.

PROPOSED SCHEME

This section describes proposed system model followed by the collection of articles from ten popular e-Newspapers in India that are shown in Table - 5. Further, it explains the hybrid approach of keyword extraction followed by feature extraction and summary generation.

System Model

In this paper, we proposed a system model for text summarization in multi- document as shown in Fig. 3. It starts with document collection and ends with summary generation. In between, pre-processing of the documents, keyword extraction and feature extraction phase occurs.

Table - 5 List of Indian e-Newspapers from Which Articles Were Taken

The Hindu	The Indian Express	The Times of India	Economic Times	Deccan Chronicle
Hindustan Times	The New Indian Express	The Financial Express	Deccan Herald	The Telegraph

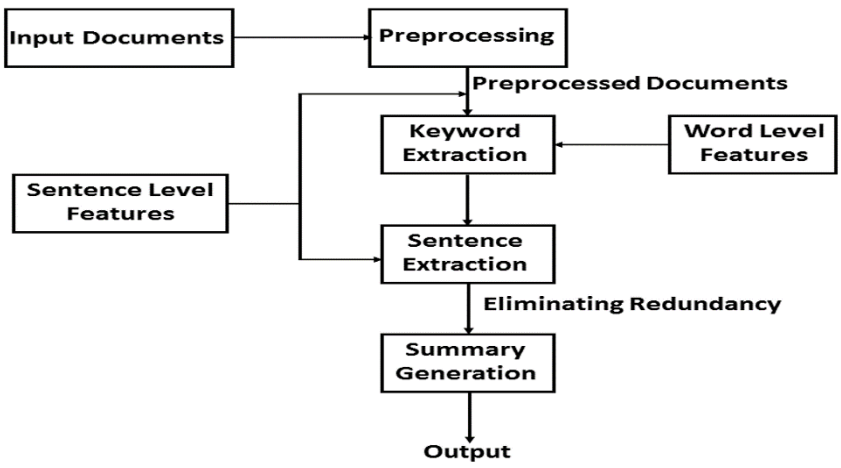

Fig. 3 System model for text summarization in multi-documents

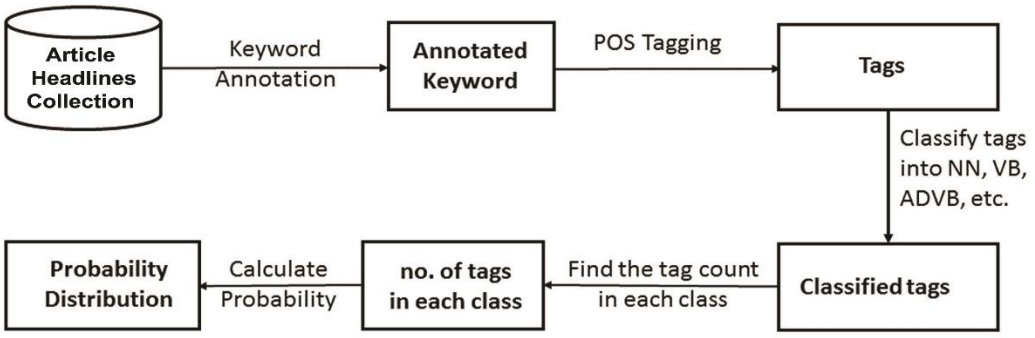

Fig.4 Linear probability distribution model for automatic keyword extraction

e-Newspapers Article Collection

After studying the website of several e-Newspapers, we collected data from ten different e-Newspapers as shown in Table - 5. Our dataset included almost 2000 similar articles from each e-Newspapers ranging from the 1st of December 2016 to 15th of March 2017.

Hybrid Keyword Extraction Training Model

The primary aim of automatic keyword extraction is to point out a set of words or phrases that best represents the document. To achieve this, a hybrid extraction technique has been proposed. The steps are shown in Fig. 4.

Keyword Annotation

For this model, there is a need for human intervention for an annotation to train the proposed algorithm. The human annotators analyse documents and select probable keywords. Due to lack of reliable human annotators, e-newspapers clippings are used as our training dataset. The articles were considered as the target document, and the headlines are used as keywords, thus eliminating the need of human annotators. We collected the headlines from all the articles in the corpus to make the training dataset. Further, these keywords are supplied to the POS tagger to find the POS information.

HMM-based POS Tagging

In this paper, we deployed an HMM-based POS tagger [80] to identify accurate POS information for all the keywords in the training corpus.

Learning Probability Distribution

The POS tag information of all the keywords in the training dataset was analysed and calculate the frequency of each tag which has appeared in the training dataset as a keyword of news headlines. The procedure of finding the value probability distribution is given in Algorithm 1.

Algorithm 1: *Probability distribution for each tag*

Data: *Dataset:* = Corpus of newspaper articles headlines (C)
Keyword: = Human annotated set of keywords for each article.
Result: P (*tag*): Probability distribution of tags
Notation: *t*: tags, *TL*: tag list, *C*: corpus, *A*: articles, *W:* word, *TF:* POS tag file, *TTC*: total tag count
Initialization: *TTC*=0
While *t in TL* **do**
tag_count = 0
End
While *A in C* **do**
While *W in A* **do**
TF = finds_POS_tag (W)
Tag_count (t) = tag_count (t) + 1
TTC = TTC + 1
End
End
While *t in TL* **do**
P (*tag*) = tag count (*t*)/*TTC*
End

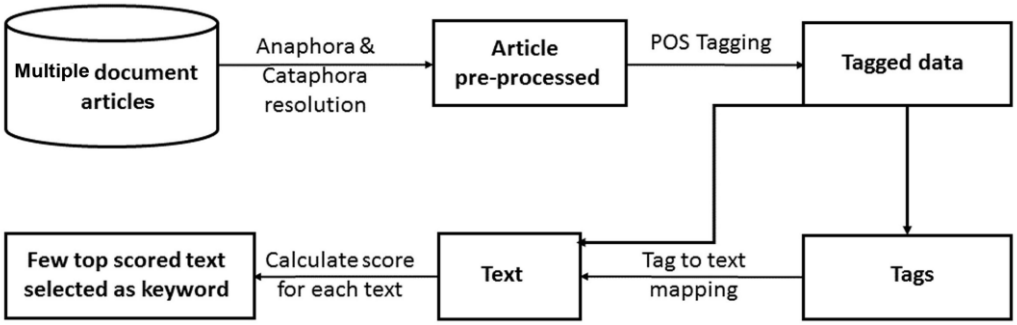

Fig.5 Keywords Extraction Model

Algorithm 1 derives P (tag) from the dataset. Input provided by the algorithm are the dataset of articles along with human annotated keywords for each article. Initially, the variable count value is initialized to 0. This variable stores the number of keywords that has been scanned by the algorithm. First, while loop initializes the tag count to 0 for every POS tag. Next, two while loops find POS tag of the keyword for every keyword in every article of the dataset and the count for that POS tag is increased by 1. Once this terminates, probability distribution, (P (tag)) is determined by dividing the tag count by the total number of keywords. Further, (P (tag)) is used as a probabilistic measure to detect keywords.

Extraction Model

Extraction (testing) model is shown in Figure 5. It requires anaphora and cataphora resolution as pre-processing for test set articles. These pre-processed articles are supplied to the POS tagger to identify POS information of all the word in the articles. The score is calculated for each word, and few top scored words are selected as keywords.

Keyword Extraction

The output file from the POS tagger is now fed into the extraction model. Unlike TF-IDF (keeping the count of the number of times a particular word has appeared), we keep the count of the word-tag pair. For example: [Can, Noun] and [Can, Verb] are treated differently. When a count of the entire document is taken, the keywords are ranked by Equation 1.

$$Score = P(tag) * Count(word, tag) \quad (1)$$

Algorithm 2: *Automatic keywords extraction process*

Data: *Dataset:* = Corpus of articles (C) = {A_1, A_2, ...A_n},
P (tag):= Set of trained probabilities,
Num_of_keywords: = Required number of keywords.
Result: List of extracted keywords (K_w [])
Notation: *P F: pre-processed file, P OSF: POS tag file, C: corpus, A: articles, W: word, T: text, T T C: total tag count, A_C_resolve: anaphora and cataphora resolution*

```
While A in C do
    PF:=A_C_resolve (A)
    POSF:=find_POS_tag (PF)
End
top = 0
While W in POSF do
   f lag:=0
  For i ← 0 to top do
     If W.text= wordset[i].text and W.tag = wordset[i].tag then
        Wordset[i].count:= Wordset[i].count + 1
        flag:= 1
     End
  End
  If flag = 0 then
     Wordset [top + 1].W:= W.W
     Wordset [top + 1].tag:= W.tag
     Wordset [top + 1].count:=1
```

```
        Wordset [top + 1].score:=0
        top: = top+1
      End
    End
    For i ← 0 to size do
      Wordset [i].score:= wordset[i].count × P (wordset[i].tag)
    End
    Sort desc (wordset.score)
    For i ← 0 to Num_keywords do
      Kw [i]:=wordset[i]
    End
```

Algorithm 2 takes the corpus of news articles A= {A1, A2, ...An} from all the ten newspapers, the number of keywords to be extracted and a probability distribution Table - trained during the training as an input for extracting keywords. The output of the algorithm will be saved in an array Keywords []. Wordset [] is another array of structures that keeps the record of the words that have already been scanned and the number of times that word-tag pair has been scanned. The top is the variable that stores the value of the number of words scanned and it is initialized to 0. The input file is passed through an anaphora and cataphora resolver. The output file of anaphora and cataphora resolver is subjected to POS tagger to get the tag information for every article in the corpus. The algorithm then courses through the file, updating existing records in the way creating new ones when needed. When the algorithm is done with parsing the file, the scores are updated. Once the scores are set, the array is sorted according to the scores of each word-tag pair. The top score value of few texts is then extracted as keywords.

Feature Extraction

The system uses both sentence level and word level features which help to extract most important sentences from the news articles.

Sentence level features

- Sentence location feature: the ratio between line number of the sentence and the total number of sentences in the document. The position scores in the document within 0 and 1.
- Length of the sentence: length of sentences must be greater than the threshold value. Here, the threshold value set as 10 (including punctuation).

Word level features

- Thematic word method: This method identifies essential keywords of the articles during assigning probability distribution to each word using keyword extraction algorithm. Sentences that contain the cluster of such thematic words should be critical of the corresponding articles.
- Font based feature: Sentences that contain a word which is written in bold, italic, underlined or combination of these. The probability distribution value of these type words is assigned as 1.

During summation, we used all the above mentioned features for calculating the score of the sentence.

Sentence Extraction

With the help of algorithms explained so far, a set of word-tag pair keywords, as well as their respective scores, sentence level, and word level features along with their respective scores, are attained. For summarization, the proposed algorithm suggests that one derives from many sentences for a keyword from the article as is proportional to the score it received. Further, one can derive these sentences by any means, be it through clustering means or crude scoring.

Summary Generation

Finally, let us discuss the working procedure of the proposed scheme with an example. Suppose there is an article related to pollution that contains around hundred sentences. We take the similar article from all the ten newspapers of the same day as the testing corpus. The names of the different newspapers are shown in Table - 5. The possible keywords which will be common in all the newspapers would be pollution, destruction, harmful, environment, atmosphere, bacteria, sewerage and disease. Assume, one wants to summarize it in twenty sentences using the individual scores of the keywords as shown in Table - 7. Finally, using Equation 2, one will know that how many sentences exactly they need to extract for every keyword to get the desired summary of the articles. Table - 7 refers the required number of sentences need to be extracted for every keyword.

$$DSIA = \frac{Keyword\ score\ * N\ o.of\ sentences\ req}{T\ otal\ score\ of\ all\ the\ keywords} \quad (2)$$

Table - 6 Probability Distribution of Each Keywords

Pollution	Destruction	Harmful	Disease	Bacteria	Atmosphere	Environment	Sewerage
4.8	0.8	2.1	2.8	4.7	0.9	2.9	1

Where, DSIA is the desired number of sentences in summary using each keyword.

Table – 7 Score of Each Keywords

Pollution	Destruction	Harmful	Disease	Bacteria	Atmosphere	Environment	Sewerage
5	1	3	3	5	1	3	1

Procedure for selection of desired number of sentences

The proposed algorithm uses the following method to select required number of sentences to get the desired summary.

- The algorithm parses through the document and makes a count of the number of keywords appeared in each sentence. The algorithm considers this as a score, then sorts them in descending order according to the number of keywords each sentence contains. Next, it starts popping sentences from the top of this sorted stack. Instead of popping a predetermined number of sentences from the top, if a keyword is present in the popped sentence, the number of allocated sentence is reduced by 1. However, if a sentence has no keywords whose allocated sentence is greater than 0, the sentence is rejected. The algorithm continues until the number of allocated sentence for each keyword becomes 0.
- Instead of scoring sentences by the number of keywords it contains, we score them using the scores of the individual keywords present in them. For example: 'Pollution is so harmful that many international agencies have come together to battle pollution'. The sentence has a score of 2 * score (Pollution) + score (Harmful). The algorithm then sorts the sentences according to these scores in descending order and extracts the required number of sentences from the top.

OTHER METHODS OF KEYWORD EXTRACTION

In this paper, three well-known existing keyword extraction methods namely, TF-IDF, NFA, and TF-AIDF are implemented to compare the performance of proposed method. These methods are as follows.

TF-IDF based Keyword Extraction

Term frequency-inverse document frequency (TF-IDF) weight [81] recognizes essential words from the article corpus. In this method, essential keywords are those keywords that frequently occur within a particular document, but that doesn't frequently occur in the rest of the documents in the corpus. The term frequency (TF) measures the number of times a word shows up in the present document while the inverse document frequency (IDF) processes the number of documents in which the word occurs. At the point when the word is more frequent in the sentence but less frequent in the entire document, the TF-IDF value is higher.

TF-IDF is characterized as follows.

Let us consider a collection of N documents as D = {D_1, D_2, ..., D_N } and a word w appears in L documents where L ≤ N as D_{i1}, D_{i2}, ..., D_{iL}} from the collected document then,

$$TF - IDF = TF * IDF \quad (3)$$

$$IDF\ (w) = \log[\frac{N}{(L)}] \quad (4)$$

Where, TF = number of times word w occur in the document, N = Total number of documents in the corpus and, L = Total number of documents in which the word w occurs.

TF-IDF assigns a weight to the word w in document D:

- TF-IDF is the highest, when w occurs many times within a document and not occurs in rest of the documents in the corpus;
- TF-IDF is lower, when w occurs fewer times in a document or occurs in many documents;
- TF-IDF is the lowest, when w occurs virtually in all the documents.

NFA based keyword extraction

Number of False Alarm (NFA) is based on Helmholtz principle [82, 81]. Helmholtz principle of perception says, if a geometric structure has a very low probability to appear in noise, then it is perceptually meaningful. In the case of textual sequential or unstructured data, Balinsky *et al* [82, 81] suggested a qualitative measure for such deviations. To identify the important words from the set of documents, one need to compute the NFA as shown in Algorithm 3. In this algorithm, if the word w appears m times in a particular document and its NFA value is smaller than some meaningfulness factor then the word is known as meaningful. All the meaningful words in a corpus are defined as a set of meaningful keywords in the documents.

The Algorithm 3 takes the corpus of newspapers 'articles as an input and checks the length of each document in the corpus C. If the length is equal then it proceeds further, otherwise, pre-process the document set in the corpus and make sub-documents of equal length. Next, it checks the frequency (m) of each word in the sub-document to identify the meaningful word in the corresponding sub-document. If the value of m is less than 1, means word 'w' is not meaningful and repeat the process for other words in the sub-document. Otherwise, it calculates the number of false alarm (NFA) of word w using Equation 5 and then calculate the meaning (M) of word w using Equation 6. If the value of M is positive, then word w is a meaningful word and append it to Kw and mark was a meaningful word for Pi. Repeat the above process for all the words in the whole corpus. The final list of words in Kw is the desired extracted keywords for summarization.

Algorithm 3: *Probability distribution for each tag*

Data: *Dataset:* = Corpus of newspaper articles documents (C)
Result: Set of meaningful keywords.
Notation: D: document, C: corpus, P: sub-document, w: words,
K: frequency of particular word appears in the document (D_i),
m: frequency of particular word appears in sub-document (P_i),
NFA: number of false alarm, M: meaning,
K_W: List of meaningful keyword
Initialization: $C = \{D_1, D_2, ..., D_N\}$, $D = \{P_1, P_2, ..., P_r\}$, $K_W = \{\varphi\}$
For $i \leftarrow 1$ **to** r **do**
While *each w in* P_i **do**
find the value of m
If $(m < 1)$ **then**
repeat the process of next W in P.
End else
Calculate NFA (w, P, D)
Calculate M (w, P, D)
End
If $(M > 0)$ **then**
Append w to the K_W *and* mark was a meaningful word for P_i.
End
End
End

$$NFA(w,P,D) = \frac{1}{N(m-1)} \cdot \frac{K!}{m!(K-\mathrm{m})!} \quad (5)$$

$$M(w,P,D) = -\left[\frac{1}{m}\right] \log[NFA(w,P,D)] \quad (6)$$

TF-AIDF based Keyword Extraction

Term frequency-adaptive inverse document frequency (TF-AIDF) [81] is an NFA-based TF-IDF. Let us consider corpus (C) contains N number of documents *i.e.* C = {D_1, D_2, ..., D_N }. A word w appears in L number of documents Cw = {D_{i1}, D_{i2}, ..., D_{iL}} out of N and combine them into one document (the document about w) $\check{D}$ = D_{i1} + D_{i2} + ... + D_{iL}. Further, we need to calculate NFA for every appearance of w in $\check{D}$. Using adaptive window size or moving windows,

$$IDF(w) = \log\left[\frac{N}{(L)}\right] \quad (7)$$

If all the documents D_1, D_2, ..., D_N are of same size, then number of false alarm can be calculated as:

$$NFA(w) = \left[\frac{K-1}{K}\right].IDF(w) \quad (8)$$

If all the documents D_1, D_2, ..., D_N are of different size, adaptive IDF can be calculated as:

$$AIDF(w) = \left[\frac{K-1}{K}\right] .\log[\frac{\sum_{i=1}^{N}|D_i|}{\check{D}}] \quad (9)$$

RESULTS AND DISCUSSION

This section describes the performance of proposed method for keyword extraction and text summarization. It also depicts the comparison with three additional methods for automatic keyword extraction i.e. TF-IDF, NFA, and TF-AIDF. Finally, it gives a brief discussion on performances of these four techniques.

Experimental Results

This paper aimed to drive a common summary of multiple articles which are taken from ten different newspaper sources on the same date with the similar article title and content. We have tested these ten newspaper articles under four different techniques such as proposed method, TF-IDF, NFA, and TF-AIDF. The list of ten different newspapers is shown in Table - 5. Each article contains around thirty sentences. Next, we applied the algorithms to extract meaningful keywords from all ten articles. Finally, we used these extracted keywords for summarization of all the ten articles into a single article. As the title of all the ten articles is similar. Therefore, extracted keywords will have many common words and summary of these articles will have most common sentences. We analyse the number of common sentences extracted by all the four keywords extraction algorithms based on their extracted keywords in a final summary. The proposed method has extracted around 96.23% of semantically common sentences among the articles. Similarly, 91.64%, 89.27%, and 87.43% of semantically common sentences are extracted using NFA, TF-AIDF, and TF- IDF keyword extraction algorithms respectively.

Statistical Results

To assess the performance of all the four keyword extraction method, three statistical parameters are considered namely, precision, recall and F-measure. The formula to ascertain precision, recall and F-measure appeared in equations 10, 11 and 12 respectively.

$$Precision = \frac{Tp}{Tp + Fp} \tag{10}$$

$$Recall = \frac{Tp}{Tp + Fn} \tag{11}$$

$$F1 - measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{12}$$

Table - 8 Precision, Recall, and F − Measure of Articles Title using T F − IDF Technique

Articles	Newspapers	TF-IDF		
		Precision	*Recall*	*F-measure*
Article 1	The Hindu	0.881	0.946	0.912
Article 2	Hindustan Times	0.853	0.975	0.91
Article 3	The Indian Express	0.838	0.864	0.85
Article 4	The New Indian Express	0.828	0.876	0.851
Article 5	Deccan Chronicle	0.867	0.838	0.852
Article 6	The Times of India	0.851	0.934	0.89
Article 7	The Financial Express	0.834	0.972	0.898
Article 8	Economic Times	0.879	0.924	0.9
Article 9	Deccan Herald	0.872	0.912	0.891
Article 10	The Telegraph	0.863	0.824	0.843

Table - 9 Precision, Recall, and F − Measure of Articles Title using T F − AIDF Technique

Articles	Newspapers	TF-AIDF		
		Precision	*Recall*	*F-*
Article 1	The Hindu	0.89	0.971	0.929
Article 2	Hindustan Times	0.86	0.936	0.896
Article 3	The Indian Express	0.857	0.873	0.865
Article 4	The New Indian Express	0.831	0.898	0.863
Article 5	Deccan Chronicle	0.91	0.884	0.897
Article 6	The Times of India	0.857	0.938	0.891
Article 7	The Financial Express	0.879	0.914	0.896
Article 8	Economic Times	0.889	0.949	0.918
Article 9	Deccan Herald	0.90	0.959	0.928
Article 10	The Telegraph	0.867	0.935	0.9

Table - 10 Precision, Recall, and F − Measure of Articles title using N F A Technique

Articles	Newspapers	NFA		
		Precision	*Recall*	*F-measure*
Article 1	The Hindu	0.891	0.975	0.931
Article 2	Hindustan Times	0.889	0.987	0.935
Article 3	The Indian Express	0.885	0.964	0.923
Article 4	The New Indian Express	0.843	0.919	0.88
Article 5	Deccan Chronicle	0.971	0.966	0.968
Article 6	The Times of India	0.875	0.935	0.904
Article 7	The Financial Express	0.898	0.975	0.935
Article 8	Economic Times	0.893	0.956	0.924
Article 9	Deccan Herald	0.919	0.984	0.95
Article 10	The Telegraph	0.868	0.977	0.92

Table – 11 Precision, Recall, and F − Measure of Articles Title using Proposed Technique

Articles	Newspapers	Proposed		
		Precision	*Recall*	*F-*
Article 1	The Hindu	0.933	0.967	0.95
Article 2	Hindustan Times	0.898	0.989	0.941
Article 3	The Indian Express	0.918	0.983	0.95
Article 4	The New Indian Express	0.851	0.933	0.89
Article 5	Deccan Chronicle	0.977	0.951	0.964
Article 6	The Times of India	0.878	0.939	0.907
Article 7	The Financial Express	0.899	0.978	0.937
Article 8	Economic Times	0.913	0.95	0.931
Article 9	Deccan Herald	0.926	0.989	0.956
Article 10	The Telegraph	0.886	0.979	0.93

To extract the meaningful keyword from multiple e-Newspapers articles, we deployed all the four algorithms (proposed, TF-IDF, NFA, and TF-AIDF) on a set of articles which are taken from ten different newspaper sources on the same date with the similar article title. However, the headlines were not provided to the algorithms. We have collected the content of all the ten articles and fed into the algorithms to check the performance of all the four algorithms. The input was the newspapers clippings, and the end target was to extract meaningful words that were present in the headlines. On having run the algorithms against the clippings, we got the following results shown in Tables 8, 9, 10, and 11.

RESULTS AND DISCUSSION

The execution of the proposed algorithm was examined on a relatively extensive corpus of documents. To represent the results, we choose a set of more than hundred e-Newspaper articles from ten different newspapers in India. Every article comprises of more than hundred words approximately. At first, the punctuation was expelled from the articles. In pre-processing approach just stop words removal was performed. To address the issue of the variable length articles, adaptive window sizes mi was applied for every news article. In every article, K and mi value is varied. To implement TF-IDF values, IDF is varied for every article.

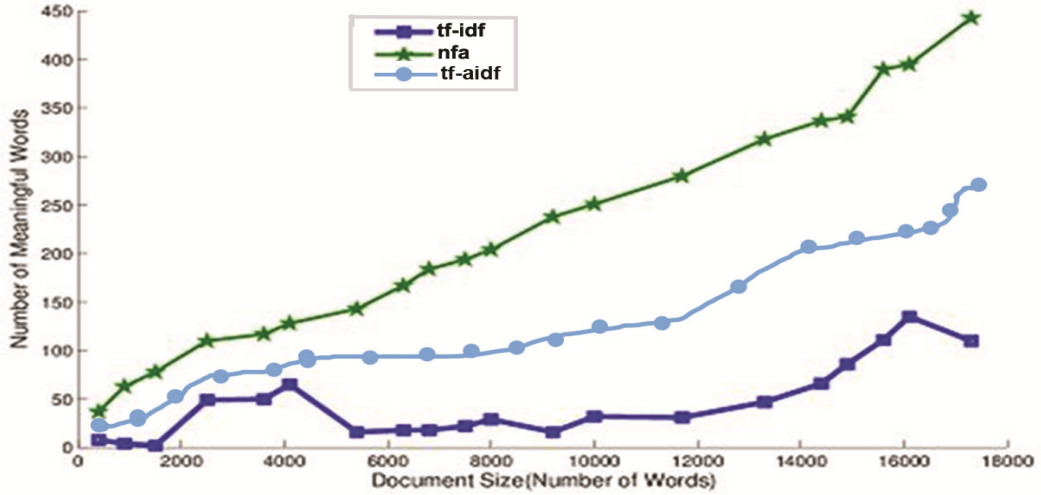

Fig.6 NFA vs TF-IDF and TF-AIDF graph

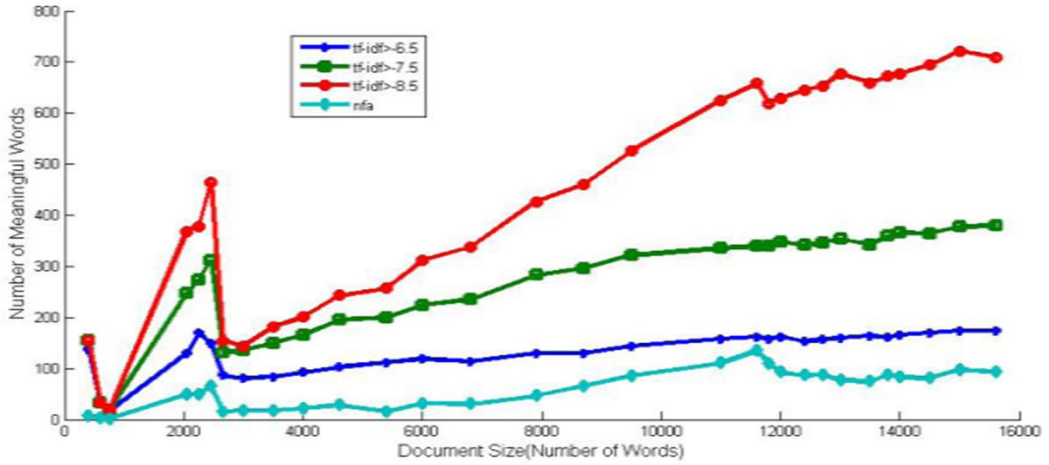

Fig. 7 NFA vs Different values of log (TF-IDF)

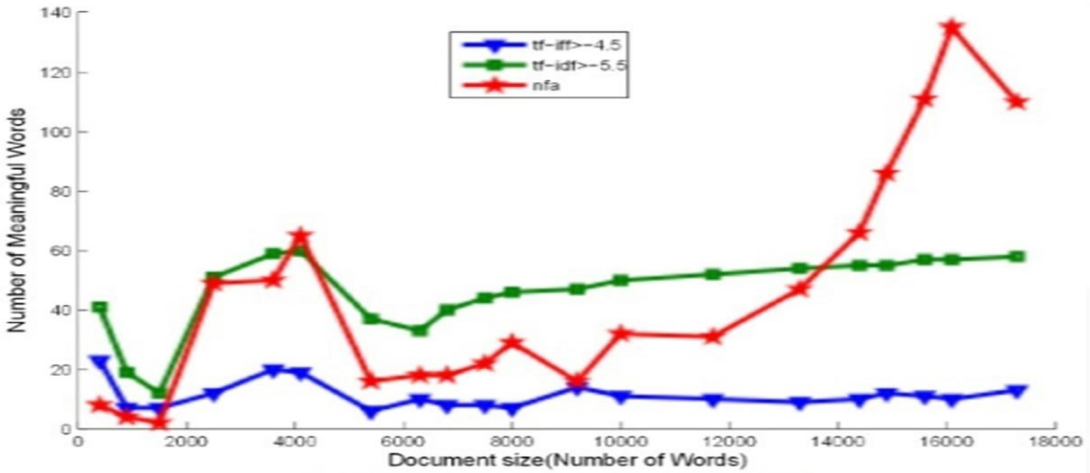

Fig. 8 NFA vs Different values of log (TF-IDF)

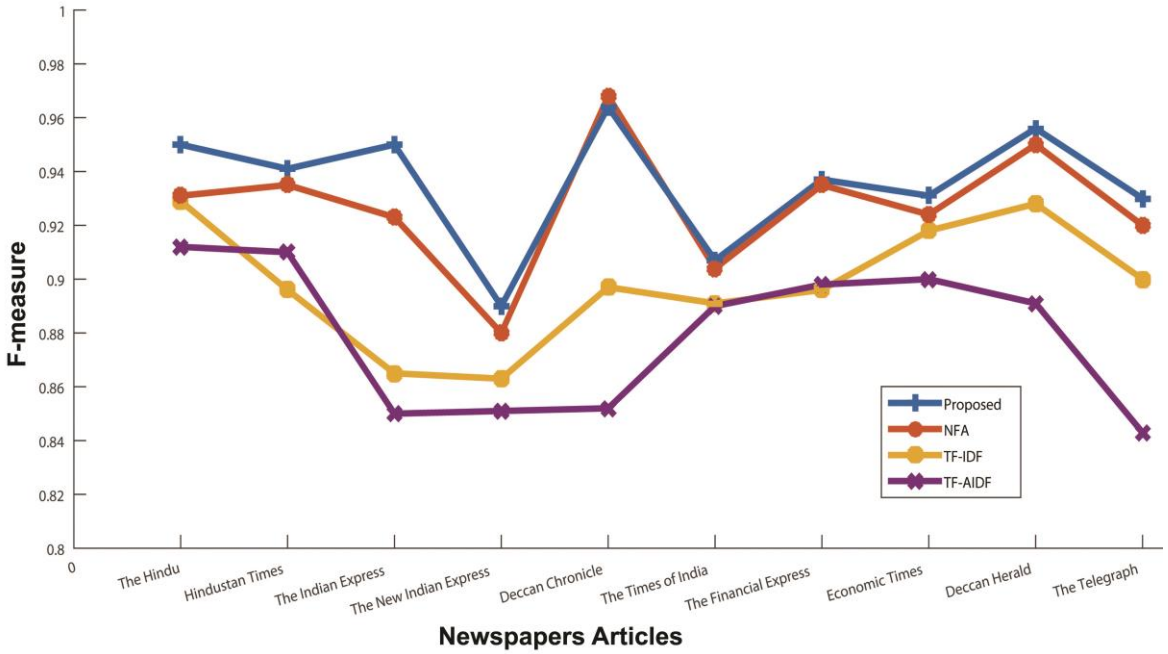

Fig. 9 Comparison of F-measure attained by proposed, NFA, TF-IDF, TF-AIDF for keyword extraction

In this paper, the meaningful words are extricated by four techniques namely, proposed method, TF-IDF, TF-AIDF, and NFA. We showed the comparison of the number of words extracted using TF-IDF, TF-AIDF, and NFA in Fig. 6. The proposed was not compared here as it requires the number of keywords to be fed as input. The total number of extracted keywords in TF-AIDF and NFA are directly proportional to the size of the documents. However, TF-IDF demonstrated that the number of meaningful words does not rely on the size of the documents. To discover TF-IDF, the adaptive window size is applied in every article. The value of IDF and TF is varied in every article. To isolate easily the number of meaningful words extracted using TF-IDF with different threshold values, log function is applied as shown in Fig. 7 and Fig. 8. With the comparison of NFA, the number of meaningful words is more in Log (TF-IDF) with a threshold value greater than -8.5, -7.5, and -6.5 as shown in Figure 7.

In Figure 8, NFA is compared with log (TF-IDF) with a threshold value greater than -4.5 and -5.5. Every data is examined, and the words quantity in these articles changes significantly. For log (TF-IDF) with a threshold value greater than -4.5, the number of meaningful words is more than those in NFA up to 1500 words roughly. After that, the quantity of extracted words decreases. How- ever, for log (TF-IDF) with a threshold value greater than -5.5, the number of meaningful words are extricated more up to 13000 words approximately then reductions occur. Finally, a comparison of f-measure attained by all the four keyword extraction techniques is shown in Fig. 9 and we can observe that proposed method has outperformed over other methods of keyword extraction. The existing algorithms mainly rely on the frequency of a word within a document and very less in rest of the corpus to select a word as an essential word. In this paper, most of the essential keywords are common among articles as it has similar title and content of same day newspapers. Therefore, proposed method works better in this kind of evaluation.

CONCLUSION

This work has proposed interdependent algorithms in keyword extraction and text summarization for multi-document. Three other algorithms are also implemented to compare with the results of the proposed algorithm. We tested the proposed system with ten e-Newspapers article taken from ten different popular Indian newspapers. The keyword detection algorithm worked very efficiently in recognizing keywords and had an impressive precision, recall, and f-measure over other algorithms. While summarizing the articles from ten into one, we observe that the proposed method has outperformed over other three algorithms. The proposed method has extracted around 96.23% of semantically common sentences among the articles. Similarly, 91.64%, 89.27%, and 87.43% of semantically common sentences are extracted using NFA, TF-AIDF, and TF-IDF keyword extraction algorithms respectively.

REFERENCES

[1] C Zhang, Automatic Keyword Extraction from Documents using Conditional Random Fields, *Journal of Computational Information Systems*, **2008**, 4 (3), 1169–1180.

[2] E Hovy and CY Lin, Automated Text Summarization and the Summarist System, *Proceedings of a Workshop on held at Baltimore*, Maryland, ACL, **1998**, 197–214.
[3] I Mani and MT Maybury, Advances in Automatic Text Summarization, Vol. 293, *MIT Press*, **1999**.
[4] G Erkan and DR Radev, Lexrank: Graph-Based Lexical Centrality as Salience in Text Summarization, *Journal of Artificial Intelligence Research*, **2004**, 22, 457–479.
[5] M Banko and RC Moore, Part of Speech Tagging in Context, *Proceedings of the 20th International Conference on Computational Linguistics*, Stroudsburg, PA, USA, **2004**.
[6] JR Thomas, SK Bharti and KS Babu, Automatic Keyword Extraction for Text Summarization in e-Newspapers, *Proceedings of the International Conference on Informatics and Analytics*, ACM, **2016**, 86–93.
[7] L van der Plas, V Pallotta, M Rajman and H Ghorbel, Automatic Keyword Extraction from Spoken Text: A Comparison of Two Lexical Resources: the edr and wordnet, *arXiv preprint* cs/0410062.
[8] MJ Giarlo, A Comparative Analysis of Keyword Extraction Techniques, A Technical Report, *Rutgers, The State University of New Jersey*, **2006**, 1-14.
[9] G Ercan and I Cicekli, Using Lexical Chains for Keyword Extraction, *Information Processing & Management* **2007**, 43 (6), 1705–1714.
[10] M Litvak and M Last, Graph-Based Keyword Extraction for Single-Document Summarization, *Proceedings of the workshop on Multi-Source Multilingual Information Extraction and Summarization*, ACL, **2008**, 17–24.
[11] E Filatova and V Hatzivassiloglou, Event-Based Extractive Summarization, *Proceedings of ACL Workshop on Summarization*, Barcelona, Spain, **2004**, 111.
[12] JM Conroy, JD Schlesinger and JG Stewart, Classy Query-Based Multi-Document Summarization, *Proceedings of the 2005 Document Understanding Workshop*, Boston, Citeseer, **2005**.
[13] F Schilder and R Kondadadi, Fastsum: Fast and Accurate Query-Based Multi-Document Summarization, *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, ACL, **2008**, 205– 208.
[14] R Witte and S Bergler, Fuzzy Clustering for Topic Analysis and Summarization of Document Collections, *Advances in Artificial Intelligence*, Springer, **2007**, 476–488.
[15] D Archambault, D Greene, P Cunningham and N Hurley, Themecrowds: Multiresolution Summaries of Twitter Usage, *Proceedings of the 3rd International Workshop on Search and mining user-generated contents*, ACM, **2011**, 77–84.
[16] JD Cohen, Highlights: Language and Domain-Independent Automatic Indexing Terms for Abstracting, *Journal of the American Society for Information* (*JASIS)*, **1995,** 46 (3), 162–174.
[17] HP Luhn, A Statistical Approach to Mechanized Encoding and Searching of Literary Information, *IBM Journal of Research and Development*, **1957**, 1 (4) 309–317.
[18] J Ramos, Using TF-IDF to Determine Word Relevance in Document Queries, *Proceedings of the First Instructional Conference on Machine Learning*, **2003**.
[19] Y Matsuo and M Ishizuka, Keyword Extraction from a Single Document using Word Co-Occurrence Statistical Information, *International Journal on Artificial Intelligence Tools*, **2004**, 13(1), 157–169.
[20] LF Chien, Pat-Tree-Based Keyword Extraction for Chinese Information Retrieval, *ACM SIGIR Forum*, ACM, **1997**, 31, 50–58.
[21] R Barzilay and M Elhadad, Using Lexical Chains for Text Summarization, *Advances in Automatic Text Summarization*, **1999**, 111–121.
[22] A Hulth, Improved Automatic Keyword Extraction given more Linguistic Knowledge, *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, ACL, **2003**, 216–223.
[23] G Salton, A Singhal, M Mitra and C Buckley, Automatic Text Structuring and Summarization, *Information Processing & Management*, **1997**, 33 (2), 193–207.
[24] JM Conroy and DP O'leary, Text Summarization Via Hidden Markov Models, *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, **2001**, 406–407.
[25] K Zhang, H Xu, J Tang and J Li, Keyword Extraction using Support Vector Machine, *Advances in Web-Age Information Management*, Springer, **2006**, 85–96.
[26] E Frank, GW Paynter, IH Witten, C Gutwin and CG Nevill-Manning, Domain-Specific Keyphrase Extraction, In *16th International Joint Conference on Artificial Intelligence (IJCAI 99)*, **1999,** 2, 668-673.
[27] IH Witten, GW Paynter, E Frank, C Gutwin and CG Nevill-Manning, Kea: Practical Automatic Keyphrase Extraction, *Proceedings of the Fourth ACM Conference on Digital libraries*, ACM, **1999**, 254–255.
[28] P Turney, Learning to Extract Keyphrases from Text, *arXiv preprint cs/0212013*, **2002**, 1-43.
[29] JK Humphreys, Phraserate: An HTML Keyphrase Extractor, *A Technical Report, Department of Computer Science, University of California*, Riverside, California, USA, **2002**.
[30] SF Dennis, The Design and Testing of a Fully Automatic Indexing-Searching System for Documents Consisting of Expository Text, *Information Retrieval: A Critical Review*, Washington DC: Thompson Book Company, **1967**, 67– 94.

[31] G Salton and C Buckley, Automatic Text Structuring and Retrieval-Experiments in Automatic Encyclopaedia Searching, *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, **1991**, 21–30.
[32] Y Ohsawa, NE Benson and M Yachida, Keygraph: Automatic Indexing by Co-Occurrence Graph Based on Building Construction Metaphor, *In Research and Technology Advances in Digital Libraries, 1998. (ADL 98), Proceedings of IEEE International Forum*, IEEE, **1998**, 12–18.
[33] F Fukumoto, Y Sekiguchi and Y Suzuki, Keyword Extraction of Radio News Using Term Weighting with an Encyclopaedia and Newspaper Articles, *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, **1998**, 373–374.
[34] R Mihalcea, Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization, *Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions*, ACL, **2004**, 20.
[35] JJ Pollock and A Zamora, Automatic Abstracting Research at Chemical Abstracts Service, *Journal of Chemical Information and Computer Sciences*, **1975**, 15(4), 226–232.
[36] D Marcu, Discourse Trees Are Good Indicators of Importance in Text, *In I. Mani and M. Maybury (eds), Advances in Automatic Text Summarization*, MIT Press, **1999,** 123-136.
[37] SM Harabagiu and F Lacatusu, Generating Single and Multi-Document Summaries with Gistexter, *Proceedings of Document Understanding Conferences (DUC-2002)*, **2002**, 40–45.
[38] H Saggion, K Bontcheva and H Cunningham, Robust Generic and Query-Based Summarisation, *Proceedings of the Tenth Conference on European Chapter of the ACL*, **2003**, 2, 235–238.
[39] L Shi, F Wei, S Liu, L Tan, X Lian and MX Zhou, Understanding Text Corpora with Multiple Facets, *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, **2010**, 99–106.
[40] RM Alguliev, RM Aliguliyev, MS Hajirahimova and CA Mehdiyev, MCMR: Maximum Coverage and Minimum Redundant Text Summarization Model, *Expert Systems with Applications*, **2011**, 38(12), 14514–14522.
[41] ZL Min, YK Chew and L Tan, Exploiting Category-Specific Information for Multi-Document Summarization, *Proceedings of COLING 2012: Technical Papers*, ACL, **2012**, 2093–2108.
[42] PE Genest and G Lapalme, Framework for Abstractive Summarization using Text-to-Text Generation, *Proceedings of the Workshop on Monolingual Text-to-Text Generation*, ACL, **2011**, 64–73.
[43] D Tsarev, M Petrovskiy and I Mashechkin, Using NMF-based Text Summarization to Improve Supervised and Unsupervised Classification, *11th* IEEE *International Conference on Hybrid Intelligent Systems (HIS)*, **2011**, 185–189.
[44] G de Melo and G Weikum, Uwn: A Large Multilingual Lexical Knowledge Base, *Proceedings of the ACL System Demonstrations*, ACL, **2012**, 151–156.
[45] SA Mirroshandel and G Ghassem-Sani, Towards Unsupervised Learning of Temporal Relations between Events, *Journal of Artificial Intelligence Research*, **2012**, 45(1), 125–163.
[46] C Aone, ME Okurowski and J Gorlinsky, Trainable, Scalable Summarization using Robust NLP and Machine Learning, *Proceedings of the 17th International Conference on Computational Linguistics*, ACL, **1998**, 1, 62–66.
[47] DR Radev, W Fan and Z Zhang, Webinessence: A Personalized Web-Based Multi-Document Summarization and Recommendation System, *Ann Arbor*, **2001,** *1001*, 48103.
[48] B Hachey, Multi-Document Summarisation using Generic Relation Extraction, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ACL, **2009**, 1, 420–429.
[49] A Farzindar, F Rozon and G Lapalme, Cats a Topic-Oriented Multi-Document Summarization System at DUC, *Proceedings of the Document Understanding Workshop*, **2005**.
[50] M Fuentes, H Rodrıguez and D Ferrés, Femsum at duc 2007, *Proceedings of the Document Understanding Conference*, **2007**.
[51] S Fisher, A Dunlop, B Roark, Y Chen and J Burmeister, Ohsu Summarization and Entity Linking Systems, *Proceedings of the Text Analysis Conference*, Citeseer, **2009**.
[52] A Dong, Y Chang, Z Zheng, G Mishne, J Bai, R Zhang, K Buchner, C Liao and F Diaz, Towards Recency Ranking in Web Search, *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, ACM, **2010**, 11–20.
[53] DR Radev and KR McKeown, Generating Natural Language Summaries from Multiple Online Sources, *Computational Linguistics*, **1998,** 24(3), 470–500.
[54] HH Chen and CJ Lin, A Multilingual News Summarizer, *Proceedings of the 18th Conference on Computational Linguistics, Association for Computational Linguistics*, **2000**, 1, 159–165.
[55] DR Radev, S Blair-Goldensohn and Z Zhang, Experiments in Single and Multi-Document Summarization Using Mead, *Ann Arbor*, **2001**, *1001*, 48109.
[56] DR Radev, S Blair-Goldensohn, Z Zhang and RS Raghavan, Newsinessence: A System for Domain-Independent, Real-Time News Clustering and Multi-Document Summarization, *Proceedings of the First International Conference on Human Language Technology Research*, ACL, **2001**, 1–4.
[57] R Brandow, K Mitze and LF Rau, Automatic Condensation of Electronic Publications by Sentence Selection, *Information Processing & Management*, **1995**, 31(5), 675–685.

[58] R Barzilay, KR McKeown and M Elhadad, Information Fusion in the Context of Multi-Document Summarization, *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL, **1999**, 550–557.
[59] H Daum´e, A Echihabi, D Marcu, D Munteanu and R Soricut, Gleans: A Generator of Logical Extracts and Abstracts for Nice Summaries, *Workshop on Automatic Summarization*, **2002**, 9–14.
[60] CY Lin and E Hovy, Automated Multi-Document Summarization in Neat's, *Proceedings of the Second International Conference on Human Language Technology Research, Morgan Kaufmann Publishers Inc*., **2002**, 59–62.
[61] KR McKeown, R Barzilay, D Evans, V Hatzivassiloglou, JL Klavans, A Nenkova, C Sable, B Schiffman and S Sigelman, Tracking and Summarizing News on a Daily Basis with Columbia's Newsblaster, *Proceedings of the Second International Conference on Human Language Technology Research*, Morgan Kaufmann Publishers Inc., **2002**, 280–285.
[62] H Saggion and G Lapalme, Generating Indicative-Informative Summaries with Sumum, *Computational Linguistics*, **2002**, 28(4), 497–526.
[63] Y Chali, M Kolla, N Singh and Z Zhang, The University of Lethbridge Text Summarizer at DUC 2003, *Proceedings of the HLT/NAACL workshop on Automatic Summarization/Document Understanding Conference)*, **2003**.
[64] T Copeck and S Szpakowicz, Picking Phrases, Picking Sentences, *Proceedings of the HLT/NAACL Workshop on Automatic Summarization/Document Understanding Conference*, **2003**.
[65] E Alfonseca and JM Guirao, A Moreno-Sandoval, Description of the UAM System for Generating Very Short Summaries at Duc–2003, *Document Understanding Conference*, Edmonton, Canada, **2003**.
[66] G Erkan and DR Radev, The University of Michigan at DUC 2004, *Proceedings of the Document Understanding Conferences Boston*, MA, **2004**.
[67] C Nobata and S Sekine, CRL/Nyu Summarization System at DUC-2004, *Proceedings of DUC*, **2004**.
[68] R Witte, R Krestel and S Bergler, Erss 2005: Co-reference-based Summarization Reloaded, *DUC 2005 Document Understanding Workshop, Canada*, **2005**.
[69] R Witte, R Krestel and S Bergler, Context-Based Multi-Document Summarization using Fuzzy Co-Reference Cluster Graphs, *Proceedings of Document Understanding Workshop at HLT-NAACL*, **2006**.
[70] YX He, DX Liu, DH Ji, H Yang and C Teng, Msbga: A Multi-Document Summarization System based on Genetic Algorithm, IEEE *International Conference on Machine Learning and Cybernetics*, **2006**, 2659– 2664.
[71] DM Dunlavy, DP OLeary, JM Conroy and JD Schlesinger, Qcs: A System for Querying, Clustering and summarizing Documents, *Information Processing & Management*, **2007**, 43(6), 1588–1605.
[72] F Gotti, G Lapalme, L Nerima, E Wehrli and T du Langage, Gofaisum: A Symbolic Summarizer for DUC, *Proceedings of DUC*, **2007**, 7.
[73] KM Svore, L Vanderwende and CJ Burges, Enhancing Single-Document Summarization by Combining Ranknet and Third-Party Sources, *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, **2007**, 448–457.
[74] Y Liu, X Wang, J Zhang and H Xu, Personalized Pagerank based Multi-Document Summarization, *IEEE International Workshop on Semantic Computing and Systems*, **2008**, 169–173.
[75] J Zhang, X Cheng, G Wu and H Xu, Adasum: An Adaptive Model for Summarization, *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, ACM, **2008**, 901–910.
[76] E Aramaki, Y Miura, M Tonoike, T Ohkuma, H Mashuichi and K Ohe, Text2Table: Medical Text Summarization System Based on Named Entity Recognition and Modality Identification, *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, ACL, **2009**, 185–192.
[77] F Wei, S Liu, Y Song, S Pan, MX Zhou, W Qian, L Shi, L Tan and Q Zhang, Tiara: A Visual Exploratory Text Analytic System, *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, **2010**, 153–162.
[78] MP Marcus, MA Marcinkiewicz and B Santorini, Building a Large Annotated Corpus of English: The Penn treebank, *Computational Linguistics*, **1993**, 19(2), 313–330.
[79] S Lappin and HJ Leass, An Algorithm for Pronominal Anaphora Resolution, *Computational linguistics*, 20(4), **1994**, 535–561.
[80] SK Bharti, B Vachha, R Pradhan, K Babu and S Jena, Sarcastic Sentiment Detection in Tweets Streamed in Real Time: A Big Data Approach, *Digital Communications and Networks*, **2016**, 2(3), 108–121.
[81] A Balinsky, H Balinsky and S Simske, *On the Helmholtz Principle for Data Mining*, Hewlett-Packard Development Company, LP, **2011**.
[82] AA Balinsky, HY Balinsky and SJ Simske, On Helmholtz's Principle for Documents Processing, *Proceedings of the 10th ACM Symposium on Document Engineering*, **2010**, 283–286.
[83] SK Bharti and KS Babu, Automatic Keyword Extraction for Text Summarization: A Survey, *http://arxiv.org/abs/1704.03242*, **2017**, 1-12.