Natural Language Engineering **25** (1): 121–146. © Cambridge University Press 2018
doi:10.1017/S1351324918000414

Extractive multi-document summarization based on textual entailment and sentence compression via knapsack problem

ALI NASERASADI[1], HAMID KHOSRAVI[2] and FARAMARZ SADEGHI[2]

[1]*Department of Applied Mathematics, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran*
e-mail: naserasadi@uk.ac.ir
[2]*Department of Computer Science, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran*
e-mails: hkhosravi@uk.ac.ir, farsad@uk.ac.ir

(*Received 24 October 2017; revised 20 September 2018; accepted 21 September 2018; first published online 31 October 2018*)

Abstract

By increasing the amount of data in computer networks, searching and finding suitable information will be harder for users. One of the most widespread forms of information on such networks are textual documents. So exploring these documents to get information about their content is difficult and sometimes impossible. Multi-document text summarization systems are an aid to producing a summary with a fixed and predefined length, while covering the maximum content of the input documents. This paper presents a novel method for multi-document extractive summarization based on textual entailment relations and sentence compression via formulating the problem as a knapsack problem. In this approach, sentences of documents are ranked according to the extended Tf-Idf method, then entailment scores of selected sentences are computed. Through these scores, the final score of each sentence is calculated. Finally, by decreasing the lengths of sentences via sentence compression, the problem has been solved by greedy and dynamic Programming approaches to the knapsack problem. Experiments on standard summarization datasets and evaluating the results based on the Rouge system show that the suggested method, according to the best of our knowledge, has increased F-measure of query-based summarization systems by two per cent and F-measure of general summarization systems by five per cent.

1 Introduction

One phenomenon of the digital age is the presence of a large amount of online information in various forms on the internet and other computer networks, and their ever-increasing growth rate. In the meantime, textual documents are the most frequently used forms of information. Users' effective access to concise information is one of the problems caused by this phenomenon, so that searching news topics in the search engines offers thousands and perhaps millions of web pages to the users. In this way, the major problem for the user is the time-consuming (and

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

perhaps impossible) process of exploring this data to reach a definite and concise conclusion about the subject. On the other hand, there are also other problems such as inconsistency or conflict between data and information, completion, and updates of this information in short periods of time, uncertainty about coherence and suitability of data and information, etc. (Nenkova 2006).

Automatic summarization of textual documents is a solution that has been widely regarded by researchers in recent years. A summary of information could be produced by using textual documents summarization mechanisms so that the user can be informed about the content of documents by reading the summary. So in a short time, he can decide whether to use the documents or not. The concept of summarization has been mentioned and explored in natural language processing issues for more than six decades. However, this topic has been researched seriously for less than two decades. The most important reasons for this phenomenon are the spreading of the internet and other communication networks, the large amounts of data and information on them, and the need to summarize this data to save time and effort (Das and Martins 2007).

An automatic document summarization system is a system that receives one or more documents as inputs and processes them without human intervention to produce a coherent and versatile summary with a fixed length (Nenkova 2006). By 'summary', we mean a text that has been produced from one or more documents, contains the most important information of the original documents, and its length is shorter than half of the original document (Radev, Hovy and McKeown 2002). In fact, the summarization system aims to produce a summary that has maximum coverage of original information contents with minimum redundancy. This type of summarization is called informative summarization. Also, there is another type of summarization that is called indicative summarization and its goal is to find relevant documents with same information. On the other hand, the informative summary that has been generated by the system must have an acceptable level of performance, compliance and readability in comparison with the summary that has been written by a human expert, so that the reader can be truly conversant with the most important contents of the input documents via reading the summary (Lin 2004; Lin *et al.* 2006; Rankel *et al.* 2013).

Document summarization systems have been categorized into two categories according to the number of input documents: single-document summarization and multi-document summarization. Also, they have been categorized into two groups according to the summary generation approach: extractive summarization and abstractive summarization. In the extractive approach, the summary is generated by selecting some sentences from sentences of the input documents according to a combination of some features (such as sentence position, correspondence with keywords, word frequency and so on) and a ranking strategy. In the abstractive approach, the summary is generated via language generation mechanisms according to syntactic and semantic forms of input documents, and it necessarily does not contain input sentences of documents. Of course, parameters such as relation to the subject domain, summary generation according to a query that has been provided by the user, updating the generated summary by new input documents, etc., also, affect the summary generation approach (Nenkova and McKewon 2012).

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

As mentioned earlier, in extractive multi-document summarization systems, the summary is generated via extracting some sentences from input documents. For this purpose, the sentences of input documents are ranked according to a scoring and ranking strategy and then, the best sentences are selected to generate the final summary (Liu and Liu 2009). Among different approaches to perform this process, nowadays, more attention is paid to semantic relation between sentences. In fact, sentences with close semantic relations are ranked in such a way that one or few sentences are selected among them for generating the final summary, thereby increasing its coverage (Gong and Liu 2001; Schluter and Sogaard 2015; Yasunaga *et al.* 2017). In this regard, using textual entailment relation, as a proper approach for checking the semantic relation between two sentences, can increase the system performance (Bentivogli *et al.* 2009; Gupta *et al.* 2012; Pado *et al.* 2015). On the other hand, due to some redundant information in the sentences of input documents, using sentence compression techniques for decreasing their lengths, while reducing redundancy if the final summary, could increase its coverage, too (Madnani *et al.* 2007; Liu and Liu 2009). In addition, formulating the summarization problem as an optimization problem is one of the most important approaches for selecting the best sentences (according to their scores and lengths while satisfying the length constraint of the final summary) and has been used in some summarization systems (Hirao *et al.* 2013).

The presented paper is focused on multi-document extractive summarization based on textual entailment and sentence compression. In this way, the outlined problem has been formulated as an optimization problem (knapsack problem), where sentences are the problem items, and their value has been calculated by the production of their textual entailment score and Tf-Idf value. Then, greedy and dynamic programming approaches have been used to solve the knapsack problem. Also, before generating the final summary, to reduce the space that has been occupied by each sentence, sentence compression methods have been applied to adjust the length of sentences. Rest of the paper has been organized as follows: in Section 2, we review the related works and subject background and define textual entailment and sentence compression concepts. Section 3 presents the proposed method. In Section 4, experiments and results are presented. Finally, in Section 5, we conclude the paper and present an outline for future works.

2 Background and literature review

Extractive summarization systems are categorized into two categories based on a need to change the form of input documents: systems without transformation and transformation-based system.

2.1 Transformation-based systems

Transformation-based systems transform the input document(s) to a new format, and use this new format to extract the summary.[1] One of the most widespread

[1] The traditional definition of transformation-based systems is not the one adopted here. For traditional definition, see Mani and Maybury (1999).

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

transformation-based summarization systems uses Latent Semantic Analysis which is a reliable and unsupervised method. In this method, input documents are represented as a word–sentence matrix, and then this matrix is used for measuring similarity between sentences and key words via Latent Semantic Analysis. In this matrix, the rows represent the words and the columns represent sentences. To evaluate the sentences, Singular Value Decomposition is used. One of the biggest disadvantages of this method is the matrix dimension; by increasing the input size, the matrix size increases exponentially, thus the chance of generating unsuitable and worthless summaries increases too (Gong and Liu 2001). In Dunlavy *et al.* (2007), this method is combined with the Hidden Markov Model.

As another transformation-based system, we can mention Graph-based systems. In this system, input documents are mapped to a graph at first, so that the graph vertices show the text units, and the graph edges show relation between text units. Then, by processing the resulting graph, suitable text units are selected to generate the summary. As an example of this method, we can mention Brazilay and Elhadad (1997) in which lexical chains are used to understand discourse structure and generate a coherent and cohesive summary. LexRank or PageRank algorithms that have been proposed in Erkan and Radev (2004) and are used by Google search engine for scoring documents are additional examples of these systems. In this algorithm, the graph vertices include sentences from the input document(s) and the edges are created according to some form of semantic similarity (cosine similarity) or content overlap (Tf-Idf vector).

Rhetorical structure theory, which was introduced in 1988, is another example of transformation-based summarization systems, and tries to represent input documents as a tree. In this method, the input is divided into two non-overlapping sections so that there is a semantic relation between them. One of these sections is called nucleus and the other is called satellite. The nucleus section is more important for the author and is independent from the satellite section. Also, the reverse relation is invalid (Marcu 1997). It worth mentioning that different nucleuses and satellites could be selected via different approaches according to the intended relation such as antithesis, concession, condition, circumstance and so on.

2.2 Systems without transformation

Systems without transformation are systems that do not change the content or structure of documents, and work directly on the content of entry. These systems deal with each sentence separately, and generate the final summary via extracting a subset of sentences from input documents. The first step in these systems is extracting features of input sentences. In the next step, the score of each sentence is calculated based on a combination of values of its features. Finally, according to theses scores, the best sentences are selected to make the final summary. Of course, assuring confidence in coherence and non-redundancy of the summary in the final step is essential (Nenkova and McKewon 2011, 2012).

The simplest and oldest methods of these systems use frequency of words (Das and Martins 2007), probability distribution of words (Nenkova, Vanderwende and

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

McKeown 2006; Vanderwende *et al.* 2007; Lin and Chen 2010), position of words (Baxendale 1958; Edmundson 1969; Lin and Hovu 1997; Havy and Lin 1998; Orasan, Pekar and Hasler 2004), cue words (Pollock and Zamora 1999), title words (Lopez, Prince and Roche 2011; Suzuki and Fukumoto 2014) and key phrases (Kutlu, Cigir and Cicekli 2010). Most of these methods do not have desired performances due to focusing just on the words and not paying attention to the semantic relation between words and sentences of input document(s). Another method for systems without transformation uses the centroid of document sets. In this method, each document of the set is represented as a phrase vector, which is called vector space. Vectors include numerical values which show the importance of phrases in the document. In other words, the centroid of a document set is calculated by dividing the sum of document vectors by the number of all documents. The most common technique for determining these values uses Tf-Idf (Filatova and Hatzivassiloglou 2004).

Bayesian models have been used in without transformation summarization systems, too. Bayesian models are one of the most complex approaches of document summarization and their popularity is increasing. The first instance of these models was proposed in Lopez, Buenaga and Gomez-Hidalgo (2004), Daume and Marcu (2006) and Haghighi and Vanderwende (2009), which used some different probability distributions for words of input document(s). These distributions are general probability of the language, which is called G, probability distribution of the whole cluster that is being summarized, which is called C and a probability distribution for each document in the input document set. Each of these distributions includes a word table and their probability or weight. Of course, in each distribution, the weight of words can be different. Then, the most important words are selected according to these distributions and by using a non-supervised learning method. After that sentences are selected to generate the summary. Also, the most popular method for sentence selection after determining their scores, is the greedy method. It is mentionable that for improving evaluation and sentence selection process, Kullback–Leiber divergence has been used in Donaway, Drummey and Mather (2000), Radev *et al.* (2003) and in Litvak, Vanetik and Last (2015), MDL algorithm has been used based on database transactions. Also, Jensen–Shannon divergence has been used for evaluation of the generated summary (Louis and Nenkova 2009).

In query-based systems, a query is given to the system in company with the document(s). This query shows the user preference or need, and it can be expressed as a set of words. In these systems, sentences with more similarity to the query are given higher scores (Baumel, Cohen and Elhadad 2014). A common way to measure similarity between the text and the query is to use the soft cosine similarity. In this method, same as the centroid of documents set, the query sentence is represented as a Tf-Idf vector, and the similarity between each sentence of the text and the query is calculated as equation (1).

$$\mathrm{Cos}(q,s) = \frac{\sum_{i\in N}((\mathrm{tf}*\mathrm{idf})_{i,q} \times (\mathrm{tf}*\mathrm{idf})_{i,s})}{\sqrt{\sum_{i\in N}((\mathrm{tf}*\mathrm{idf})_{i,q})^2} \times \sqrt{\sum_{i\in N}((\mathrm{tf}*\mathrm{idf})_{i,s})^2}} \tag{1}$$

where q is the query, s is the sentence and N is a set that contains all features of q and s (such as words, n-grams or syntactic n-grams).

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Of course, using stemming, semantic similarity of words and word synonyms can improve performance of this process (Baumel *et al.* 2014; Wang *et al.* 2016). Also, we can use more than one feature simultaneously to improve the sentence ranking process. In these conditions, different features should be combined in a predefined way. The most common way to combine different features is by using a linear combination of all features (Saggion and Gaizauskas 2004). One of the problems with this method is determining the weight of each feature in linear combination. For solving this problem, some papers suggested examining the different weight for each feature manually to find the most appropriate weights. It is evident that this process is very time consuming, costly, and in most circumstances, it is impossible. So other ways have been examined to find the best weight for each feature.

For instance, the unsupervised machine learning method and the Naive–Bayesian classifier have been used for combining feature sets (Daume and Marcu 2005; Silva *et al.* 2015). It has been showed that this form of feature combination improves the results (Jin *et al.* 2008). Also, other methods of machine learning, such as decision trees (Chuang and Yang 2000), hidden Markov model (Conroy and O'Leary 2001; Knight and Marcu 2002; Zhou and Hovy 2003), support vector machine (Hirao *et al.* 2002; Fuentes, Alfonseca and Rodriguez 2007), genetic algorithm (Litvak, Last and Friedman 2010), conditional random fields (Galley 2006; Shen *et al.* 2007), integer linear programming models (Martins and Smith 2009; Woodsend and Lapata 2012), log-linear models (Osborne 2002) and supervised and unsupervised machine learning algorithms (Hong and Nankova 2014; Hong, Marcus and Nenkova 2015; Li, Liu and Zhao 2015; Schluter and Sogaard 2015) have also been used to solve this problem.

One problem with the above methods is that ranking algorithms specify whether a sentence is suitable and valuable for generating the summary or not. In these circumstances, if there are many helpful sentences, then the above algorithms would not be useful for selecting the best subset of sentences to maximize coverage of the summary. Mainly, to solve this problem, learning-based ranking algorithms have been used. In these methods, a complementary algorithm is used for ranking proper sentences, and finally, sentences with higher rankings are selected to generate the summary (Amini, Usunier and Gallinari 2005; Toutanova *et al.* 2007; Metzler and Kanungo 2008; Amini and Usunier 2009; Cai and Li 2013). Another method of ranking sentences uses regression. These methods, which use support vector regression, have better classification and ranking quality (Li *et al.* 2007; Ouyand *et al.* 2011). In addition, neural networks have been used for text summarization (Mani and Maybury 1999; Cao *et al.* 2015). Also, Yasunaga *et al.* (2017) have used a graph convolutional network and recurrent neural networks to propose a neural multi-document summarization system; Cao *et al.* (2017) have used convolutional neural networks and text classification models for this purpose; and Cao *et al.* (2015) have used recursive neural networks to develop a sentence ranking framework for summarization. Also, Cao *et al.* (2016) have proposed a summarization system called Attsum, which uses convolutional neural networks to learn query relevance ranking and sentence saliency ranking simultaneously.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

However, despite the importance of paying attention to the semantic relation between the sentences, this concept has received less attention in most of the previous researches. In fact, many of the sentences that ranked as valuable sentences in the most of the above methods are semantically close to each other and the presence of one of them in the final summary is enough. So detecting a semantic relation between sentences via methods such as textual entailment, and using this relation to remove sentences that have meanings close to each other, could decrease the summary length and make it possible to increase its coverage by including other sentences. Of course, in some researches, some instances of relation between sentences, such as discourse relation through an approximate discourse graph (called G-flow) to balance coherence and salience in the summary (Christensen, Soderland and Etzioni 2013) or relations based on G-flow, Tf-Idf cosine similarity and personalized discourse graph (Yasunaga *et al.* 2017) through three sentence relation graphs have been used for sentence ranking or extraction and decreasing redundancy in the summary. Nevertheless, textual entailment as a new and powerful tool for detecting semantic relation between two sentences, has been less used for this goal. Also, most of the sentences that are selected to generate the summary have additional parts (such as relative clauses or redundant constituents), so removing these parts in the compression phase could improve the summary quality, too.

2.3 Textual entailment

One of the ways that have been used in the last decade for detecting the relations of sentences in the summarization systems is textual entailment. Textual entailment is a one-way relation between a text (T) and a hypothesis (H) that shows whether the hypothesis can be inferred from the text, or not. In fact, the purpose of textual entailment is that taking a text (T) and using relevant knowledge, determines whether or not the hypothesis (H) is inferred from the text. We should mention that in this process the focus should not be just on the knowledge and the text must play a vital role. Table 1 shows some sample of textual entailment pairs and their results.

It is mentionable that natural language is ambiguous. In other words, a text could contain more than one meaning, and a single meaning could be expressed by multiple different phrases. This would cause a many-to-many mapping between natural language phrases and their meanings. On the other hand, as mentioned earlier, one of the key processes of text summarization is to identify and select sentences with higher values. In other words, we have to identify sentences which satisfy the constraints on the desired length of summary and also have the most coverage of the text content and meaning. In this way, if we could identify and classify sentences with the same meaning and choose the sentence with minimum length and maximum coverage of meaning from each class of sentences, we could make an important step in generating a proper summary. In fact, we can use textual entailment to determine semantic relation between sentences and, according to the result of this process, we can perform the sentence ranking process more effectively. Different approaches are suggested for performing this task. Among them, we can

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Table 1. *Samples of textual entailment* (Dagan, Glickman and Mafnini 2006)

Text	Hypothesis	Result
The two suspects belong to the thirtieth Street gang, which became embroiled in one of the most notorious recent crimes in Mexico: a shootout at the Guadalajara airport in May, 1993, that killed Cardinal Juan Jesus Posadas Ocampo and six others.	*Cardinal Juan Jesus Posadas Ocampo died in 1993.*	True
Regan attended a ceremony in Washington to commemorate the landings in Normandy.	*Washington is located in Normandy.*	False
Google files for its long awaited IPO.	*Google goes public.*	True
The SPD got just 21.5 per cent of the vote in the European Parliament elections, while the conservative opposition parties polled 44.5 per cent.	*The SPD is defeated by the opposition parties.*	True
According to NC Articles of Organization, the members of LLC company are H. Nelson Beavers, III, H. Chester Beavers and Jennie Beavers Stewart.	*Jennie Beavers Stewart is a shareholder of Carolina Analytical Laboratory.*	False

mention word embedding, logical models, graphical models, rule systems, contextual focusing and machine learning (Dagan, Glickman and Mafnini 2006).

In Tatar *et al.* (2008) and Gupta *et al.* (2012), textual entailment has been used for detecting highly relevant sentences. Also, in Gupta *et al.* (2014), textual entailment and minimum vertex cover have been used to summarize a text. However, all of the above papers have focused on single-document summarization and have high-time complexity because of exploring textual entailment relation between all sentences.

2.4 Sentence compression

Sentence compression is a method for creating a compact version of a sentence while preserving syntactic and semantic relations, and the main information of the original sentence (Wang *et al.* 2016). In fact, sentence compression aims to decrease sentence length, while preserving its syntactic and semantic structures. In other words, if we could identify excessive constituents of each sentence, which do not have much impact on its meaning, and remove them from the sentence while preserving its grammatical structure, we can compress or summarize it. This task could be done via different approaches. For example, in the sentence, 'Microsoft founder, Bill Gates, said ...', removing the phrase, 'Bill Gates', does not have any effect on the meaning of the sentence, but reduces its length and could increase its score in the sentence ranking process. So we can summarize or compress the above sentence as 'Microsoft founder said ...'.

It is worth mentioning that most sentences of textual documents, such as news articles, have redundant information which cause prolongation of the sentence, and result in a decrease of its value in the summarization process. So according to

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

summary length limits, using sentence compression in multi-document summarization processes could improve its performance. In Knight and Marcu (2000), Knight and Marcu (2002), Madnani *et al.* (2007) and Liu and Liu (2009), sentence compression mechanisms have been used in textual document summarization. Also, in Martins and Smith (2009) and Berg-Kirkpatrick, Gillick and Klein (2011), a joint model for sentence extraction and compression has been used in text summarization system. In addition, in Woodsend and Lapata (2012), Almeida and Martins (2013) and Yasunaga *et al.* (2017), compression has been used in multi-document summarization systems, too. However, most of the above researches have focused only on one aspect of text summarization such as redundancy (see Yasunaga *et al.* 2017) or coverage (see Martins and Smith 2009; Woodsend and Lapata 2012) along with compression, but our proposed method is trying to increase coverage and decrease redundancy through deploying sentence compression and textual entailment.

2.5 Knapsack problem

Despite differences between various text summarization methods, in general, these systems can be formulated as a knapsack problem. Knapsack problem is a classic NP-hard problem in combinatorial optimization which, by given a set of n item as $S = \{\text{item}_1, \text{item}_2, \ldots, \text{item}_n\}$ each of them with a weight (w_i) and a value (p_i), aims to choose a subset of S such that maximizes total values, while total weights is less and or equal to a predefined limit (knapsack weight limit or W). In other words, the object of this problem is to satisfy equation (2).

$$\text{maximize} \sum_{i=1}^{n} p_i x_i \qquad \text{Subject to} \sum_{i=1}^{n} w_i x_i \leq W \quad \text{and} \quad x_i \in \{0,1\} \tag{2}$$

where x_i shows whether item_i has been selected in final solution or not.

Accordingly, in extractive summarization systems, a score of a sentence in ranking component can be formulated as the value of item (p_i), the length of sentence as the weight of the item (w_i) and desired length of the summary can be formulated as weight limit of knapsack (W). So the summarization problem can be formulated as knapsack problem, and in this way, we can claim that the summarization problem (using above formulation) is an NP-hard problem.

In Riedhammer *et al.* (2008), Takamura and Okumura (2009) and Hirao *et al.* (2013), this formulation has been used for extractive text summarization, but all of them have focused on single-document summarization. Also, in Nishikawa *et al.* (2012), redundancy constrained knapsack problem has been used for document summarization.

3 Proposed method

Our proposed method is shown in Figure 1, consists of the following components:

(1) Preprocessing component.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

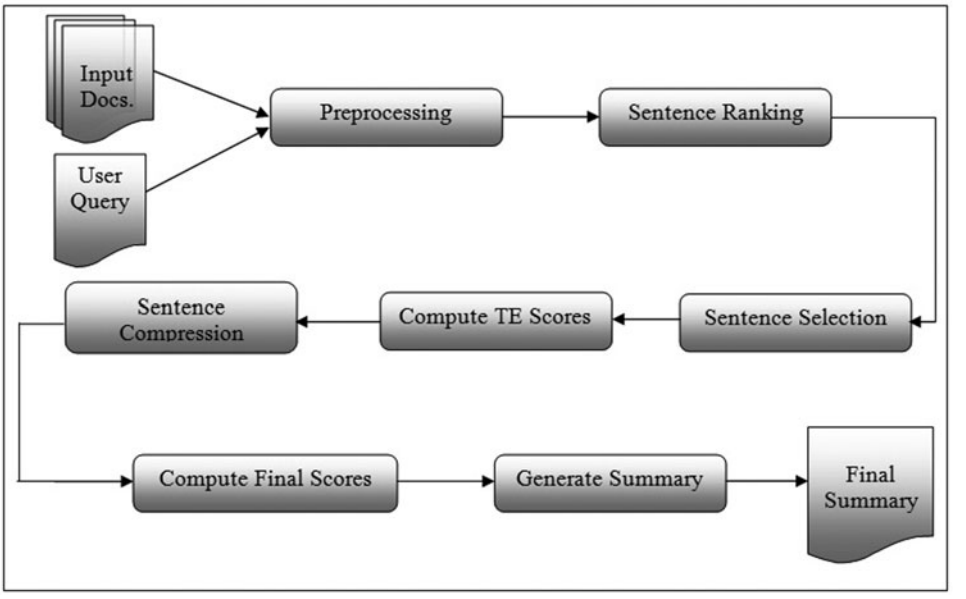


Fig. 1. Outline of the proposed method.

(2) Primitive sentence ranking component.
(3) Sentence selection component.
(4) Textual entailment score calculating component.
(5) Sentence compression component.
(6) Final score calculating component.
(7) Final summary generating component.

We will elaborate on these components in the following sections.

3.1 Preprocessing component

This component is responsible for reading the input documents and the user query (if there exist any) in text format and performing the following process on them:

- Text segmentation: This process is to divide input text into meaningful units such as sentences. Despite the existence of obvious signs for the end of a sentence, this task is not trivial because there are some signs, such as full stop in abbreviations, which make it harder than we think. For example, the abbreviation of, 'Mr.', in the sentence, 'Mr. James went to the park', is not a sentence solitarily. Often a table containing exceptions such as abbreviations is used to solve this problem.
- Tokenization: This term is applied for the task of splitting a text stream into words, phrases, symbols and other meaningful units that are called tokens. Usually, output of this task is used in the next steps of the natural language processing pipeline, such as parsing or text mining. Typically, this process is done at the word level, but defining a word is not trivial, and tokenizers often use some heuristic ways, such as punctuation and whitespace, to identify

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

words. Also, it is possible to use more complex methods, such as applying exception tables or language models in this process.

- Normalization: Normalization is the process of transforming a text into a single canonical form in order to make it consistent for performing integrated processes. In fact, sometimes a single term can be expressed in several ways; but in order to simplify natural language processes, we have to unify them. For example, the terms 'A.I.', 'AI' and 'A.I' could be unified to a single and standard format such as 'AI'; or the terms 'Windows' and 'windows' could be converted to lowercase. To do this task carefully, being informed from text nature and the manner of the next process is critical. It is mentionable that there is no general purpose normalization process.
- Stemming: This term is applied generally to the process of mapping a derived word to its root. For example, all of the words 'argue', 'argued', 'argues' and 'argus' are mapped to 'argue' but the words 'argument' and 'arguments' are reduced to 'argument'. We should mention that a word root is not always a word and there are different algorithms for stemming that differs from each other based on performance and accuracy. The simplest stemming algorithms use a lookup table, which is a simple and fast method and it can handle exceptions easily; but this method cannot find the root of new and unknown words. On the other hand, more complex approaches use Stochastic algorithms, lemmatization, n-gram analysis and suffix-stripping and matching.
- Removing stop words: Stop words are common words that are used repeatedly in sentences of a language. Due to this high occurrence, word-based analyses grant higher values to them and which may be contrary to the truth. Some examples of stop words in English include 'the', 'is', 'at', 'which'. Because of the negative impact of these words on the natural language process, in most processes, these words are removed before doing any other task. Unfortunately, there is no universal, single list for these words, and each natural language processing tool uses its own list to do this important task.

3.2 Sentence ranking component

Because of the high time complexity of calculating the textual entailment score, in this phase, we use a ranking system to select the most appropriate sentences. At first, having the set S contains all sentences obtained from the preprocessing component, each score of each sentence, $s \in S$, is calculated according to equation (3).

$$\text{Score}(s) = \frac{\sum_{a \in s} (\alpha_a \times \text{tf}(a, s) \times \text{idf}(a, S))}{l} \tag{3}$$

where a denotes each word of the sentence, l is the length of the sentence s, tf is the weighted frequency with double normalization 0.5 (as equation (4)) and idf is reverse document frequency (as equation (5)) (Robertson 2004).

$$\text{tf}(a, s) = 0.5 + 0.5 \times \frac{f_{a,s}}{\max_{\{a',s\}} f_{a',s}} \tag{4}$$

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Here, $f_{a,s}$ is the frequency of term a in sentence s, max is maximum function and gets the largest value of the function f and a' denotes each word of sentence s.

$$\mathrm{idf}(a, S) = \log \frac{N}{|s \in S : a \in s|} \tag{5}$$

where N is number of the documents, log is the logarithm function and the denominator is the number of sentences which contain term a.

Also, in equation (3), α is the impact factor of the query. In other words, in the case of generic summarization, this factor is equal to 1 for all words, but in query-based summarization, each word of query has an impact factor greater than 1, according to the system training process based on the training dataset, and this factor is, respectively, 2.17 and 2.03 for two datasets that have been used (see section 4.1). At the end of this phase, a list of sentences is sorted non-increasingly according to their scores.

3.3 Sentence selection

Based on the high time complexity of the textual entailment process and computing its scores (see Section 3.4.1), performing it on all of the sentences is not commodious. So after ranking the sentences according to primitive score, a fraction of them which have the highest score are selected (based on training set analysis and concerning the balance between runtime and accuracy of the results) and stored in a list named Sen′. According to the training data, for two datasets, this fraction is twenty per cent and twenty five per cent, respectively (see Section 4.1).

3.4 Calculation of textual entailment scores of sentences

3.4.1 Primary textual entailment score

In this component, at first, we have defined an entailment score for each pair of text and hypothesis according to equation (6).

$$\mathrm{EntailmentScore}(T, H) = \begin{cases} 0 & \text{If the entailment result is False} \\ C & \text{If the entailment result is True} \end{cases} \tag{6}$$

where T is the text, H is the hypothesis and C is the confidence score of the result of the entailment process and $C \in [0, 1)$. In fact, the output of the textual entailment algorithm (see Section 4.1) includes the entailment result (True of False) and a confidence score. As mentioned earlier, the confidence score of the entailment process is a real number and belongs to $[0, 1)$, in which, the greater the number, the stronger the reliance of the textual entailment result between the pair of sentences. For example, for '*Google flies for its long awaited IPO*.' as text, and '*Google goes public*.' as hypothesis, the entailment result is True (see Table 1) and confidence score is 0.89 . Then, the entailment score of each pair of sentences of Sen′ is calculated. For this purpose, each sentence is considered once as text and once as a hypothesis for all of the other sentences, and then *theEntailmentScore* of each pair is calculated (according to equation (6)). Finally, these scores are stored in a matrix named

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Table 2. *An instance for textual entailment matrix*

	S_1	S_2	S_3	S_4	S_5
S_1	0	0.43	0	0.80	0.65
S_2	0	0	0.75	0	0
S_3	0.35	0	0	0	0.55
S_4	0	0.81	0.64	0	0.91
S_5	0.42	0	0	0	0

TextualEntailmentMatrix or TEM, according to equation (7), which is an $N' \times N'$ matrix where N' is the size of Sen′.

$$\forall s_i, s_j \in \text{Sen}' \quad \text{TEM}(s_i, s_j) = \begin{cases} 0 & \text{If } s_i = s_j \\ \text{EntailmentScore}(s_i, s_j) & \text{Otherwise} \end{cases} \tag{7}$$

For instance, if we have five sentences $S_1, \ldots, S_5$ in Sen', then TEM would be as Table 2.

3.4.2 Final textual entailment score

After calculating TEM, the final textual entailment score of each sentence s_i, is calculated according to equation (8).

$$\forall s_i \in \text{Sen}' \quad \text{FTEscore}(s_i) = \begin{cases} \dfrac{\sum_{s_j \in S'} \text{TEM}(s_i, s_j)}{\sum_{s_j \in S'} \text{TEM}(s_j, s_i)} & \text{If } \sum_{s_j \in S'} \text{TEM}(s_j, s_i) \neq 0 \\ +\infty & \text{If } \sum_{s_j \in S'} \text{TEM}(s_j, s_i) = 0 \end{cases} \tag{8}$$

where the numerator is sum of the confidence score (or Entailment Score) for all sentences that are entailed from s_i, and the denominator is sum of the confidence score (or Entailment Score) for all sentences that entails s_i. For example, in Table 2, FTEscore(S_1) is calculated according to equation (9).

$$\text{FTEscore}(S_1) = \frac{0.43 + 0.80 + 0.65}{0.35 + 0.42} = \frac{1.88}{0.77} = 2.44 \tag{9}$$

It is mentionable that, higher FTEscore of a sentence means that the sentence entails more sentences and is entailed form less sentences. In other words, higher FTEscore of a sentence means that the sentence has more potential to be selected for generating the final summary because it increases the coverage measure of the summary.

3.5 Sentence compression

After calculating the textual entailment score of sentences, and before generating the final summary, sentence compression is used to decrease the length of sentences and to increase their final score. Here, we have used rule-based compression according to the extended form of the framework that has been proposed in Wang *et al.* (2016) to

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Table 3. *Compression rules : Parts between brackets are removed or changed*

Rule	Example
Remove headers	[EIJING, August 17 Xinhua –] Primary and middle schools . . .
Remove footers	The project was sponsored by MacArthur Foundation [-Associated Press Writer David Ho].
Remove relative clauses	There's something [that] you should know. Microsoft confirmed [Monday] [that] there was . . .
Remove lead adverbials	[Suddenly,] the project . . . [At the time,] Daniel noted, I don't thin . . .
Remove noun appositives	Microsoft founder [, Bill Gates,] said . . .
Remove within parentheses	Karen Johnson [(last year's losing finalist)] is expected to . . .
Remove within dashes	They roasted the winning brisket [- the size of pillow -] in a mighty. . .
Named entity abbreviation	[Amnesty International → AI] called on Rwanda Monday to . . .

achieve this goal. In other words, we have used Table 3 rules to compress sentences. Also, to prevent undesirable effects of compression on the sentence, we have avoided removing more than eight words per each compression rule.

3.6 Final score calculation

After sentence compression, the final score of each sentence is calculated based on its Score and FTEscore, as shown in equation (10).

$$\forall s_i \in S' \quad \text{FinalScore}(s_i) = \frac{l_i \times \text{Score}(s_i) \times \text{FEscore}(s_i)}{l_i'} \tag{10}$$

where l_i and l_i' are length of sentence i before and after compression, respectively.

3.7 Final summary generation and postprocessing

After that, we can try to generate the final summary. To this end, and as mentioned in Section 2.5, we have formulated the summary generation problem as a zero-one knapsack problem and tried to solve it using greedy and dynamic programming approaches. So according to each approach, we solve the zero-one knapsack problem. The result of each approach is a set of sentences that satisfies the approach constraints. For example, in the dynamic programming approach, the result is a set of sentences that have the maximum value (value of the set is equal to summation of the final score of each sentence) and whose length is not more than the desired length of the summary. We should mention that the greedy approach is used to ensure speed, and dynamic programming approach is used to guarantee the accuracy of the final results. After that, the resulting sentences were sorted according to their positions in the documents and chronological order. Finally, coreference resolution was done on the sorted sentences to solve the missing references. This process

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Table 4. *A sample part of a query-based summary generated by our system*

Query: What is the scope of operations of Amnesty International and what are the international reactions to its activities? Give examples of charges lodged by the organization and complaints against it.
Generated summary: Amnesty International is part of international lobbies championing the cause [, if any,] of diehard Hutu extremists. Another concern for [Amnesty International → AI] is conscientious objection to military service. [Amnesty International's → AIs] Secretary [General Pierre Sane] told a news conference that 'war crimes are being committed in Kosovo today'. Paul Miller[, an Amnesty International researcher who] returned from Kosovo [on Sunday], cited a number of incidents where there may have been 'grave breaches' of international law. In an audit of Cook's pledge,[Amnesty International → AI] praised Britain for[, it said,] [seeking to set up an International Criminal Court in Rome], publishing an annual report on human rights, banning land mines, and taking 'limited measures' to restrict arms sales to countries that may use them to commit human rights abuses. [BANGKOK, Thailand AP -] The Thailand office of[Amnesty International → AI] announced [Monday] that three publications and two journalists have won its 1998 Journalism Award for Human Rights.

The words between brackets are removed or changed according to the compression rules.

was done on original sentences (before compression) and the obtained results was applied on the compressed sentences to guarantee that it will work as expected. A sample part of a query-based summary which is generated by our system is shown in Table 4.

4 Experiments and results

4.1 Experimental settings

We have performed our experiments on two datasets: DUC 2007 (main task) and MultiLingPilot 2011. DUC 2007 consists of forty five datasets which each include twenty five documents about a news topic and a related query. We have used cross-validation to evaluate the performance of the proposed system. So each time we selected thirty datasets for the training phase and fifteen datasets for the testing phase. In this dataset, summary length is 250 words and it is generated via a query-based summarization approach. MultiLingPilot 2011 consists of ten datasets which each include ten documents about a special topic. We have used cross-validation to evaluate the performance of the proposed system. So each time we selected seven datasets for training and three datasets for testing. In this dataset, summary length is 250 words, and it is generated via a generic summarization approach.

In preprocessing component, we have used Python 3.4.3 and its Natural Language Toolkit (NLTK 3.0) for sentence segmentation, tokenization, normalization, stemming and removing stop words. In ranking, sentence selection, sentence compression, calculating final scores and generating final summary, we have also used the above toolkit. For coreference resolution, we have used Stanford CoreNLP toolkit integrated with NLTK. In the textual entailment component, we have used the

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

BIUTEE component of Excitement Open Platform 1.2.3 which is a transformation-based textual entailment system for calculating the textual entailment score of a pair of sentences (Magnini *et al.* 2014; Pado *et al.* 2015). This component was trained with 400 text–hypothesis pairs from the RTE-5 (Bentivogli *et al.* 2009) dataset and hundred pairs that had been generated manually from training datasets.

4.2 Baselines

We have compared the performance of our system with the baselines in two situations: generic summarization and query-based summarization. To this end, we have used the two best participating systems of each dataset, the average of participating systems of each dataset, the best-proposed system in Wang *et al.* (2016) and the suggested systems of Conroy, Schlesinger and O'Learry (2007), Toutanova *et al.* (2007), Haghighi and Vanderwende (2009), Mason and Charniak (2011), He *et al.* (2012), Cai and Li (2013), Li *et al.* (2015), Canhasi and Kononenko (2016) and Cao *et al.* (2016) as baselines.

4.3 Evaluation metrics

4.3.1 Human evaluation

Nenkova and Passonneau (2004) developed the Pyramid method to manually asses how many relevant facts or Summarization Content Units are captured by summaries that have been generated by an automatic summarization system. So we asked a highly experienced professional annotator (who is not one of the authors and fluent in English) to perform a Pyramid evaluation on eight randomly selected topics from each of the above datasets (DUC 2007 and MultiLingPilot 2011). Also, to evaluate the of the above selected topics, we asked five undergraduate students to read the summaries (generated by the system) examine their linguistic quality in different areas, and rate them from one (very poor) to five (very good).

4.3.2 Automatic evaluation

To compare proposed system performance with the baselines, we have used ROUGE automatic evaluation system (Lin 2004). This system automatically measures the quality of a system by comparing its results with a gold-standard summary (the summary that has been generated by human expert) according to different metrics, and shows its performance as precision, recall and F-measure. Among various metrics of ROUGE, we have selected ROUGE-1, ROUGE-2 and ROUGE SU-4 because of their high corelation with human judgment (Lin and Hovy 2003).

4.4 Results

4.4.1 Human evaluation results

Human evaluation results of the DUC 2007 dataset (query-based summarization) are shown in Table 5. Here, we asked the evaluators to evaluate the summaries in

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Table 5. *Human evaluation Results of our system (greedy and dynamic programming approaches) compared with the best DUC 2007 peers according to ROUGE and linguistic quality (LQ)(Grammaticality (Gr), Referential Clarity (RC), Focus (Fo), Coherence and Structure (CS), Non-Redundancy (Non-R) and Content Responsiveness (CR))*

System	Pyramid	GR	RC	FO	CS	Non-R	CR
Best DUC 2007 (ROUGE)	0.351	3.43	3.3	2.65	3.21	2.17	2.82
Best DUC 2007 (LQ)	0.139	**4.21**	**4.78**	4.34	4.04	4	1.73
Greedy	0.179	4.1	4	4.2	4.1	3.8	2.17
Dynamic programming	**0.363**	4.2	4.68	**4.5**	**4.2**	**4.1**	**2.9**

Table 6. *Human evaluation results of our system (greedy and dynamic programming approaches) compared with the best MultiLingPilot 2011 peers according to ROUGE and Linguistic Quality (LQ)*

System	Overall LQ grade
Best MultiLingPilot 2011 (ROUGE)	3.2
Best MultiLingPilot 2011 (LQ)	3.83
Greedy	3.9
Dynamic programming	**4.1**

the following six areas: Grammaticality (Gr), Referential Clarity (RC), Focus (Fo), Coherence and Structure (CS), Non-Redundancy (Non-R) and Content Responsiveness (CR). Also, human evaluation results of MultiLingPilot 2011 are shown in Table 6. Here and due to a lack of suitable information, we have just evaluated the summaries based on the overall grade that evaluators assigned to the summaries. In these tables, the best value in each column is bolded.

4.4.2 Automatic evaluation results

Automatic evaluation results of DUC 2007 dataset (query-based summarization) are shown in Tables 7 and 8 (respectively, for datasets participating systems and other baselines according to access to suitable information). Also, the automatic evaluation results of MultiLingPilot 2011 (generic summarization) are shown in Table 9. It is mentionable that bold values in each table show the best value per each metric for automatic summarization systems. Also, in Table 7, human summarization results are provided for better comparison between human summarization performance and the automatic systems. Also, Table 10 represents modular evaluation results which show the impact of system component on the final results of DUC 2007 dataset and MultiLingPilot 2011 dataset.

4.5 Analysis

Applying semantic relations between the sentences of an input set of documents has received less attention in the previous multi-document summarization systems. So in

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Table 7. *Query-based summarization results of DUC 2007 (DUC 2007 participating systems)*

	ROUGE-1			ROUGE-2			ROUGE-SU4		
System	R	P	F	R	P	F	R	P	F
Human-best	0.52	0.5	0.51	0.17	0.17	0.17	0.22	0.21	0.21
Human-avg	0.48	0.48	0.48	0.14	0.14	0.14	0.19	0.19	0.19
System 24	0.45	0.44	0.44	0.12	0.11	0.12	**0.17**	0.17	0.17
System 15	0.44	0.43	0.44	0.12	0.12	0.12	**0.17**	0.17	0.17
DUC-avg	0.4	0.4	0.4	0.09	0.1	0.1	0.15	0.15	0.15
Greedy	**0.46**	**0.45**	0.45	**0.13**	0.12	**0.13**	**0.17**	0.16	0.17
Dynamic programming	**0.46**	**0.45**	**0.46**	**0.13**	**0.13**	**0.13**	**0.17**	**0.18**	**0.18**

Table 8. *Query-based summarization results of DUC 2007 (other baselines)*

	ROUGE-1	ROUGE-2	ROUGE-SU4
System	*F*-score	*F*-score	*F*-score
Conroy *et al.* (2007)	0.43	0.10	0.16
Toutanova *et al.* (2007)	...	0.09	0.13
Haghighi and Vanderwende (2009)	0.43	0.1	0.15
Mason and Charniak (2011)	0.43	0.12	0.17
He *et al.* (2012)	0.4	0.07	...
Wang *et al.* (2016)	...	**0.13**	**0.18**
Cai and Li (2013)	0.42	0.12	...
Li *et al.* (2015)	0.4	0.09	0.15
Canhasi and Kononenko (2016)	0.42	0.09	0.16
Cao *et al.* (2016)	0.44	0.12	...
Greedy	0.45	**0.13**	0.17
Dynamic programming	**0.46**	**0.13**	**0.18**

Table 9. *Generic summarization results of MultiLingPilot 2011*

	ROUGE-1			ROUGE-2			ROUGE-SU4		
System	R	P	F	R	P	F	R	P	F
Peer 10	0.52	0.52	0.52	0.25	0.25	0.25	0.27	0.27	0.27
Peer 2	0.46	0.46	0.46	0.17	0.19	0.18	0.2	0.2	0.2
DS-avg	0.42	0.44	0.43	0.14	0.15	0.15	0.18	0.19	0.18
Greedy	0.49	0.52	0.51	0.28	0.29	0.28	0.23	0.26	0.25
Dynamic programming	**0.56**	**0.57**	**0.56**	**0.31**	**0.33**	**0.32**	**0.32**	**0.32**	**0.32**

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Table 10. *Modular evaluation results (F-Score) of DUC 2007 and MultiLingPilot 2011*

			DUC 2007	MultiLingPilot 2011
Without TE and comp.	Without KS	R1	0.17	0.18
		R2	0.02	0.05
		SU4	0.02	0.06
	Greedy KS	R1	0.21	0.24
		R2	0.04	0.10
		SU4	0.03	0.09
	DP KS	R1	0.24	0.26
		R2	0.06	0.17
		SU4	0.06	0.14
TE without comp.	Without KS	R1	0.35	0.38
		R2	0.07	0.14
		SU4	0.09	0.16
	Greedy KS	R1	0.42	0.48
		R2	0.10	0.25
		SU4	0.14	0.22
	DP KS	R1	0.45	0.53
		R2	0.12	0.30
		SU4	0.17	0.30
TE and comp.	Without KS	R1	0.36	0.41
		R2	0.09	0.19
		SU4	0.12	0.19
	Greedy KS	R1	0.44	0.51
		R2	0.12	0.28
		SU4	0.17	0.25
	DP KS	R1	0.46	0.56
		R2	0.13	0.32
		SU4	0.18	0.32

this paper, we tried to apply these relations by using textual entailment as a powerful tool for detecting the semantic relation between two sentences. As Tables 5 and 6 imply, the linguistic quality of our system's generated summaries is higher than the baseline's. This means that our system outperforms the other systems in generating summaries that are closer to human judgment. Using textual entailment to identify the semantic-related sentences in a set of documents, and classifying and ranking the sentences according to the textual entailment score of them, is one of the important reasons for our system's high performance. Also, beside using textual entailment, for

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

sentence ranking, formulating the summarization problem as a knapsack problem has impressively improved the performance of the summarization system.

As the results of Tables 7–9 show, the significant impact of this approach is to improve the recall parameter. Improving the recall parameter means that the system has remarkable potential for generating a summary with the selected sentences which have a higher correlation with human experts' selected sentences. In other words, this approach can select the most similar sentences to human experts' viewpoints for generating the final summary. So it has the closest performance to the best human summary (as shown in Table 7). Also, as expressed before, using sentence compression could decrease length of the selected sentences, making it possible to generate the summary with more sentences and thus increasing information coverage of the summary. In other words, our system increased information coverage of the generated summaries by using sentence compression.

Regarding the diversity aspect of a summarization system, we can mention that using equations (8) and (10) for calculating the final score of each sentence, would increase the score of sentences that entails more other sentences and be entailed from less other sentences. In other words, in a group of similar sentences, the sentence that has more potential to entail others, gives higher score and other sentences give lower scores. So, when generating the final summary, fewer sentences of a similar group of sentences are selected and as a consequence, we could select more diverse sentences and so the diversity of the system will increase. In addition, using sentence compression process increased the number of sentences in the final summary through reducing the length of the sentences, and as a consequence, increased the system's diversity, too. In this regard, non-redundancy parameter of Table 5 can be a good testimony for this claim, which shows the redundancy of the proposed system is fewer than other baseline systems.

On the other hand, in generic summarization, the dynamic programming approach for solving the knapsack problem was completed as expected, because this approach tries to optimize the value according to the different aspects, but the greedy approach has lower performance. However, in query-based summarization, especially for ROUGE-1 and ROUGE-2, the results of the greedy method are very close to the results of dynamic programming. With respect to the correlation of greedy method to the human problem-solving approaches, we can conclude that ROUGE-1 and ROUGE-2 have a higher correlation with the human experts' summary evaluation judgment.

5 Conclusion and future works

This paper presents a novel method for multi-document extractive text summarization. In this paper, we formulated the summarization problem as a knapsack optimization problem and have used extended Tf-Idf, and textual entailment approaches for sentence ranking and sentence compression to decrease the sentences lengths. Finally, we have used greedy and dynamic programming methods to solve the problem and generate the summary. Our methods performance impressively outperforms other similar systems, so that, according to the best of our knowledge,

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

our system has increased F-measure of query-based summarization systems by two per cent and F-measure of generic summarization systems by five per cent.

As future work, we are interested in the reinforcement of our ranking strategy by adding features such as sentence position and also, using sequence-based and tree-based sentence compression to decrease the sentence length. Also, applying our algorithm on update summarization and question answering systems, using phrase-based textual entailment, and applying other semantic similarity approaches such as word embedding formulations are our other future objectives. In addition, improving the scalability and complexity of the proposed system and using other optimization approached such as integer linear programming are other goals for our future researches.

References

Almeida, M., and Martins, A. 2013. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 196–206.

Amini, M., and Usunier, N. 2009. Incorporating prior knowledge into a transductive ranking algorithm for multi-document summarization. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 704–5.

Amini, M., Usunier, N., and Gallinari, P. 2005. Automatic text summarization based on word-clusters and ranking algorithms. In *Proceedings of the European Conference on Information Retrieval*, Springer, Berlin, Heidelberg, pp. 142–56.

Baumel, T., Cohen, R., and Elhadad M. 2014. Query-chain focused summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 913–22.

Baxendale, P. 1958. Machine-made index for technical literature – an experiment. *IBM Journal of Research and Development* **2**(4): 354–61.

Bentivogli, L., Clark, P., Dagan, I., and Giampiccolo, D. 2009. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the Text Analysis Conference*.

Berg-Kirkpatrick, T., Gillick, D., and Klein, D. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,* vol. 1, pp. 481–90.

Brazilay, R., and Elhadad, M. 1997. Using lexical chains for text summarization. In *Proceedings of the Association for Computational Linguistics (ACL) workshop on intelligent scalable text summarization*, vol. 17, Madrid, Spain, pp. 10–17.

Cai, X., and Li, W. 2013. Ranking through clustering: an integrated approach to multi-document summarization. *IEEE Transactions on Audio, Speech, and Language Processing* **21**(7): 1424–33.

Canhasi, E., and Kononenko, I. 2016. Weighted hierarchical archetypal analysis for multi-document summarization. *Computer Speech & Language* **37**(2016): 24–46.

Cao, Z., Li, W., Li, S., and Wei, F. 2017. Improving multi-document summarization via text classification. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pp. 3053–9.

Cao, Z., Li, W., Li, S., Wei, F., and Li,Y. 2016. Attsum: joint learning of focusing and summarization with neural attention. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING-2016)*, Osaka, Japan, pp. 547–56.

Cao, Z., Wei, F., Dong, L., Li, S., and Zhou, M. 2015. Ranking with recursive neural jnetworks and its application to multi-document summarization. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pp. 2153–2159.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Cao, Z., Wei, F., Li, S., Li, W., Zhou, M., and Wang, H. 2015. Learning summary prior representation for extractive summarization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers)*, Beijing: China, pp. 829–33.

Christensen, J., Soderland, S., and Etzioni, O. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1163–73.

Chuang, W., and Yang, J. 2000. Extracting sentence segments for text summarization: a machine learning approach, In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 152–9.

Conroy, J., and O'leary, D. 2001. Text summarization via hidden markov models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 406–7.

Conroy, J., Schlesinger J., and O'Leary, D. 2007. Classy 2007 at duc 2007. In *Proceedings of the Document Understanding Conference*.

Dagan, I., Glickman, O., and Magnini, B. 2006. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, Springer, Berlin, Heidelberg, pp. 177–90.

Das, D., and Martins, A 2007. *A Survey on Automatic Text Summarization*. Literature Survey for the Language and Statistics II course at CMU 4, pp. 192–5.

Daume, H., and Marcu, D. 2005. Bayesian multi-document summarization at MSE. In *ACL 2005*, Workshop on Multilingual Summarization Evaluation (MSE).

Daume III, H., and Marcu, D. 2006. Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL, pp. 305–12.

Donaway R., Drummey K., and Mather. L. 2000. A comparison of rankings produced by summarization evaluation measures. In *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization*, vol. 4, pp. 69–78.

Dunlavy, D., O'Leary, D., Conroy, J., and Schlesinger, J. 2007. QCS: a system for querying, clustering and summarizing documents. *Information Processing and Management* **43**(6): 1588–605.

Edmundson, H. 1969. New methods in automatic extracting. *Journal of the ACM* **16**(2): 264–85.

Erkan, G., and Radev, D. 2004. LexRank: graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research* **22**: 457–79.

Filatova, E., and Hatzivassiloglou, V. 2004. A formal model for information selection in multi-sentence text extraction. In *Proceedings of the 20th International Conference on Computational Linguistics*, ACL, p. 397.

Fuentes, M., Alfonseca, E., and Rodriguez, H. 2007. Support vector machines for query-focused summarization trained and evaluated on pyramid data. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL, pp. 57–60.

Galley, M. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, ACL, pp. 364–72.

Gong, Y., and Liu, X. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 19–25.

Gupta, A., Kathuria, M., Singh, A., Sachdeva, A., and Bhati, S. 2012. Analog textual entailment and spectral clustering (atesc) based summarization. In *Proceedings of the International Conference on Big Data Analytics*, Springer, Berlin, Heidelberg, pp. 101–10.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Gupta, A., Kaur, M., Singh, A., Goel, A., and Mirkin, S. 2014. Text summarization through entailment-based minimum vertex cover. In *Proceedings of the Third Joint Conference on Lexical and Computational Semantics (SEM-2014)*, pp. 75–80.

Haghighi, A., and Vanderwende, L. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, ACL, pp. 362–70.

Hovy, E., and Lin, C. 1998. Automated text summarization and the SUMMARIST system. In *Proceedings of a workshop on held at Baltimore Maryland*, ACL, pp. 197–214.

He, Z., Chen, C., Bu, J., Wang, C., Zhang, L., Cai, D., and He, X. 2012. Document summarization based on data reconstruction. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-2012)*.

Hirao, T., Yoshida, Y., Nishino, M., Yasuda, N., and Nagata, M. 2013. Single-document summarization as a tree knapsack problem. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP-2013)*, vol. 13, pp. 1515–20.

Hirao, T., Isozaki, H., Maeda, E., and Matsumoto, Y. 2002. Extracting important sentences with support vector machines. In *Proceedings of the 19th International Conference on Computational Linguistics*, ACL, vol. 1, pp. 1–7.

Hong, K., Marcus, M., and Nenkova, A. 2015. System combination for multi-document summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP-2015)*, pp. 107–17.

Hong, K., and Nenkova, A. 2014. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2014)*, pp. 712–21.

Jin, R., Abu-Ata, M., Xiang, Y., and Ruan, N. 2008. Effective and efficient item set pattern summarization: regression-based approaches. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, USA, pp. 399–407.

Kaikhah, K. 2004. Automatic text summarization with neural networks. In *Proceedings of the 2nd International IEEE Conference on Intelligent Systems*, IEEE, vol 1, pp. 40–4.

Knight, K., and Marcu, D. 2000. Statistics-based summarization-step one: sentence compression. In *Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI-2000)*, pp. 703–10.

Knight, K., and Marcu, D. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artificial Intelligence* **139**(1): 91–107.

Kutlu, M., Cigir, C., and Cicekli, I. 2010. Generic text summarization for Turkish. *The Computer Journal* **53**(8): 1315–1323.

Li, P., Bing, L., Lam, W., Li, H., and Liao, Y. 2015. Reader-aware multi-document summarization via sparse coding. In *IJCAI*, pp. 1270–1276.

Li, C., Liu, Y., and Zhao, L. 2015. Improving update summarization via supervised ILP and sentence reranking. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL-2015)*, pp. 1317–22.

Li, S., Ouyang, Y., Wang, W., and Sun, B. 2007. Multi-document summarization using support vector regression. In *Proceedings of Document Understanding Conference (DUC-2007)*.

Lin, C. 2004. Rouge: a package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, vol. 8.

Lin, C., Cao, G., Gao, J., and Nie, J. 2006. An information-theoretic approach to automatic evaluation of summaries. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, ACL, pp. 463–70.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Lin, S., and Chen, B. 2010. A risk minimization framework for extractive speech summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL, pp. 79–87.

Lin, C., and Hovy, E. 1997. Identifying topics by position. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, ACL, pp. 283–90.

Lin, C., and Hovy, E. 2003. 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*, ACL, vol. 1, pp. 71–8.

Litvak, M., Last, M., and Friedman, M. 2010. A new approach to improving multilingual summarization using a genetic algorithm. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL, pp. 927–36.

Litvak, M., Vanetik, N., and Last, M. 2015. Krimping texts for better summarization. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1931–5.

Liu, F., and Liu, Y. 2009. From extractive to abstractive meeting summaries: can it be done by sentence compression?. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACL, pp. 261–4.

Lopez, C., Prince, V., and Roche, M. 2011. Automatic titling of articles using position and statistical information. In *Proceedings of the Recent Advances in Natural Language Processing (RANLP-2011)*, pp. 727–32.

Lopez, M., Buenaga, M., and Gomez-Hidalgo, J. 2004. Multidocument summarization: an added value to clustering in interactive retrieval. *ACM Transactions on Informations Systems* **22**(2): 215–41.

Louis, A., and Nenkova, A. 2009. Automatically evaluating content selection in summarization without human models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, pp. 306–14.

Madnani, N., Zajic, D., Dorr, B., Ayan, N., and Lin, J. 2007. Multiple alternative sentence compressions for automatic text summarization. In *Proceedings of Document Understanding Conference (DUC-2007)*.

Magnini, B., Zanoli, R., Dagan, I., Eichler, K., Neumann, G., Noh, T., Pado, S., Stern, A., and Levy, O. 2014. The excitement open platform for textual inferences. In *Proceedings of the Association for Computational Linguistics (System Demonstrations)*, pp. 43–8.

Mani, I., and Maybury, M. T. 1999. *Advances in automatic text summarization*, MIT Press, Cambridge, MA, USA.

Marcu, D. 1997. From discourse structures to text summaries. In *Proceedings of the Association of Computer Linguistics (ACL) Workshop on Intelligent Scalable Text Summarization*, pp. 82–8.

Martins, A. F., and Smith, N. A. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pp. 1–9.

Mason, R., and Charniak, E. 2011. Extractive multi-document summaries should explicitly not contain document-specific content. In *Proceedings of the Workshop on Automatic Summarization for Different Genres, Media, and Languages*, ACL, pp. 49–54.

Metzler, D., and Kanungo, T. 2008. Machine learned sentence selection strategies for query-biased summarization. In *SIGIR Learning to Rank Workshop*, pp. 40–7.

Nenkova, A. 2006. *Understanding the Process of Multi-Document Summarization: Content Selection, Rewriting and Evaluation*. PhD dissertation, Columbia University.

Nenkova, A., and McKeown, K. 2011. Automatic summarization. *Foundations and Trends in Information Retrieval* **5**(2–3): 103–233.

Nenkova, A., and McKeown, K. 2012. A survey of text summarization techniques. In *Mining Text Data*, Springer, USA, pp. 43–76.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Nenkova, A., and Passonneau, R. 2004. Evaluating content selection in summarization: the pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL-2004): Main Proceedings*, ACL, pp. 145–52.

Nenkova, A., Vanderwende, L., and McKeown, K. 2006. A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 573–80.

Nishikawa, H., Hirao, T., Makino, T., and Matsuo, Y. 2012. Text summarization model based on redundancy-constrained knapsack problem. In *Proceedings of the International Conference on Computational Linguistics (COLING-2012) (Posters)*, pp. 893–902.

Orasan, C., Pekar, V., and Hasler, L. 2004. A comparison of summarisation methods based on term specificity estimation. In *International Conference on Language Resources and Evaluation (LREC-2004)*.

Osborne, M. 2002. Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization*, ACL, vol. 4, pp. 1–8.

Ouyang, Y., Li, W., Li, S., and Lu, Q. 2011. Applying regression models to query-focused multi-document summarization. *Information Processing & Management* **47**(2): 227–37.

Pado, S., Noh, T., Stern, A., Wang, R., and Zanoli, R. 2015. Design and realization of a modular architecture for textual entailment. *Natural Language Engineering* **21**(02): 167–200.

Pollock, J., and Zamora, A. 1999. Automatic abstracting research at chemical abstracts service. *Advances in Automatic Text Summarization* **15**(4): 43–49.

Radev, D., Hovy, E., and McKeown, K. 2002. Introduction to the special issue on summarization. *Computational Linguistics* **28**(4): 399–408.

Radev, D., Teufel, S., Saggion, H., Lam, W., Blitzer, J., Qi, H., Elebi, A., Liu, D., and Drabek, E. 2003. Evaluation challenges in large scale document summarization. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL-2003)*, pp. 375–82.

Rankel, P., Conroy, J., Dang, H., and Nenkova, A. 2013. A decade of automatic content evaluation of news summaries: reassessing the state of the art. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL-2013)*, vol. 2, pp. 131–6.

Riedhammer, K., Gillick, D., Favre, B., and Hakkani-Tur, D. 2008. Packing the meeting summarization knapsack. In *Proceedings of the INTERSPEECH*, pp. 2434–7.

Robertson, S. 2004. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation* **60**(5): 503–20.

Saggion, H., and Gaizauskas, R. 2004. Multi-document summarization by cluster/profile relevance and redundancy removal. In *Proceedings of the Document Understanding Conference (DUC-2004)*.

Schluter, N., and Sogaard, A. 2015. Unsupervised extractive summarization via coverage maximization with syntactic and semantic concepts. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-2015)*, vol. 2, pp. 840–4.

Shen, D., Sun, J., Li, H., Yang, Q., and Chen, Z. 2007. Document summarization using conditional random fields. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence (IJCAI-2007)*, vol. 7, pp. 2862–7.

Silva, G., Ferreira, R., Dueire Lins, R., Cabral, L., Oliveira, H., Simske, S., and Riss, M. 2015. Automatic text document summarization based on machine learning. In *Proceedings of the ACM Symposium on Document Engineering*, ACM, pp. 191–4.

Suzuki, Y., and Fukumoto, F. 2014. Detection of topic and its extrinsic evaluation through multi-document summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL-2014)*, vol. 2, pp. 241–6.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press

Takamura, H., and Okumura, M. 2009. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, ACL, pp. 781–9.

Tatar, D., Tamaianu-Morita, E., Mihis, A., and Lupsa, D. 2008. Summarization by logic segmentation and text entailment. *Advances in Natural Language Processing and Applications* **15**: 26.

Toutanova, K., Brockett, C., Gamon, M., Jagarlamudi, J., Suzuki, H., and Vanderwende, L. 2007. The pythy summarization system: microsoft research at duc 2007. In *Proceedings of the Document Understanding Conference (DUC-2007)*, vol. 2007.

Vanderwende, L., Suzuki, H., Brockett, C., and Nenkova, A. 2007. Beyond SumBasic: task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management* **43**(6): 1606–18.

Wang, L., Raghavan, H., Castelli, V., Florian, R., and Cardie, C. 2016. A sentence compression based framework to query-focused multi-document summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL. pp. 1384–1394.

Woodsend, K., and Lapata, M. 2012. Multiple aspect summarization using integer linear programming. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 233–43.

Yasunaga, M., Zhang, R., Meelu, K., Pareek, A., Srinivasan, K., and Radev, D. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL-2017)*, Vancouver, Canada. pp. 452–62.

Zhou, L., and Hovy, E. 2003. A web-trained extraction summarization system. In*Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology,* ACL, vol. 1, pp. 205–11.

https://doi.org/10.1017/S1351324918000414 Published online by Cambridge University Press