Knowledge-Based Systems 99 (2016) 28–38

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Multi-document summarization using closed patterns

Ji-Peng Qiang[a,b], Ping Chen[b], Wei Ding[b], Fei Xie[a,c], Xindong Wu[a,d,*]

[a] Department of Computer Science, Hefei University of Technology, Hefei 230009, China
[b] Department of Computer Science, University of Massachusetts Boston, Boston 02125, USA
[c] Department of Computer Science and Technology, Hefei Normal University, Hefei 230601, China
[d] Department of Computer Science, University of Vermont, Burlington, VT 05405, USA

ARTICLE INFO

Article history:
Received 6 March 2015
Revised 20 January 2016
Accepted 21 January 2016
Available online 2 March 2016

Keywords:
Multi-document summarization
Closed patterns
Text mining
Diversity
Content coverage

ABSTRACT

There are two main categories of multi-document summarization: term-based and ontology-based methods. A term-based method cannot deal with the problems of polysemy and synonymy. An ontology-based approach addresses such problems by taking into account of the semantic information of document content, but the construction of ontology requires lots of manpower. To overcome these open problems, this paper presents a pattern-based model for generic multi-document summarization, which exploits closed patterns to extract the most salient sentences from a document collection and reduce redundancy in the summary. Our method calculates the weight of each sentence of a document collection by accumulating the weights of its covering closed patterns with respect to this sentence, and iteratively selects one sentence that owns the highest weight and less similarity to the previously selected sentences, until reaching the length limitation. The sentence weight calculation by patterns reduces the dimension and captures more relevant information. Our method combines the advantages of the term-based and ontology-based models while avoiding their weaknesses. Empirical studies on the benchmark DUC2004 datasets demonstrate that our pattern-based method significantly outperforms the state-of-the-art methods. Multi-document summarization can be used to extract a particular individual's opinions in the form of closed patterns, from this individual's documents shared in social networks, hence provides a useful tool for further analyzing the individual's behavior and influence in group activities.

1. Introduction

Multi-document summarization has attracted much attention in recent years. With the rapid development of the World Wide Web, the explosion of electronic documents presents a serious challenge for readers to extract useful information from many relevant and similar documents. The Internet provides access to a huge volume of documents on a variety of topics with a considerable amount of redundancy. It calls for a robust multi-document summarization system, which can generate a succinct representation of a document collection by reducing information redundancy.

A large number of multi-document summarization systems have been presented in the literature. For example, the centroid-based methods [23,37] use clustering algorithms to generate sentences' clusters by calculating sentence similarity, and then select the most representative sentences from different clusters. The graph-based approaches [45,50] build a graph-based model, and then select sentences by means of voting from their neighbors using ideas like the well-known PageRank algorithm [7]. By considering latent semantics of document content, many methods based on latent semantic analysis [14] and non-negative matrix factorization [22,36] have been proposed. In addition, some ontology-based approaches [5,16] have also been used to produce summaries using lexical semantics.

Existing approaches basically fall into two major categories: term-based and ontology-based methods. A term-based method has the advantages of efficiency and maturity for term weight calculation. However, the main drawback is that it only focuses on single word significance without considering the problems of polysemy and synonymy, where polysemy means multiple meanings for a given word, and synonymy means multiple words express the same meanings. To solve these problems, ontology-based approaches take into account of meanings of lexicons. But they are restricted in some specific application domains where ontologies are available, and they cannot attain the semantic meanings of terms that do not exist in the ontology. Meanwhile, the construction of ontology is usually prohibitively expensive. To overcome these inherent weaknesses and keep the advantages of both term-based and ontology-based methods, we propose to generate a

* Corresponding author at: Department of Computer Science, Hefei University of Technology, China. Tel.: +86 055162902373.
E-mail address: xwu@hfut.edu.cn (X. Wu).

http://dx.doi.org/10.1016/j.knosys.2016.01.030
0950-7051/

summary based on closed patterns, because closed patterns can capture the associations among the words, and no additional resources are required.

Over the past decade, a large number of association mining technologies have been proposed for a variety of tasks, including association rule mining, frequent itemset mining, sequential pattern mining, closed pattern mining, and maximum pattern mining [25]. In this paper, we will discuss how to effectively use these patterns in multi-document summarization. There are many types of sequential patterns, including frequent patterns, closed patterns, and so on [11,33]. As closed patterns are more compact and contain more information than frequent patterns without losing any information, we choose closed patterns for term weight calculation. To be specific, this paper will discuss a novel method for multi-document summarization using closed patterns, namely pattern-based summarization, which simultaneously considers content coverage and non-redundancy. It holds good statistical properties and captures more relevant information relative to the term-based methods. Compared with the ontology-based approaches, our multi-document summarization using closed patterns can capture informative terms in the document collection. The method does not rely on any external resources such as lexical knowledge bases. It only relies on the information in a document collection from which the summary is to be created.

Our pattern-based method includes the following steps. First, we mine all closed sequential patterns from a corpus. Then, we present a novel method that represents all sentences using these closed patterns. The model of sentence representation covers the main content of the document collection by calculating pattern weights with respect to the distribution of the closed patterns. Finally, we iteratively choose informative and non-redundant sentences by adopting a variant of the maximal marginal relevance evaluation strategy [8]. Experiments on the standard benchmark DUC2004 data sets demonstrate that the proposed algorithm outperforms the state-of-the-art term-based and ontology-based methods.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 presents pattern-based summarization. Section 4 shows experimental results. Finally, Section 5 concludes the paper.

2. Related work

Depending on the number of documents, automatic document summarization includes single-document summarization and multi-document summarization [3,46]. Single-document summarization only condenses one document into a summary, whereas multi-document summarization condenses a document collection into a single shorter representation. Multi-document summarization is considered as an extension of single-document summarization, and needs more sophisticated technologies and attracts much attention [29,31].

Multi-document summarization methods can be classified into two classes: extractive summarization and abstractive summarization [24,26]. Extractive summarization extracts the most informative document components, and abstractive summarization involves reformulation of contents. Extractive summarization is a simple but robust method without the requirement for advanced post-processing steps. Abstractive summarization requires deep natural language processing techniques for understating the documents [30]. Extractive summarization is more feasible and has become the standard in multi-document summarization. Summarization techniques also can also be categorized into query-based or generic (given a query or not), supervised or unsupervised methods (with a training set or not).

In this paper, we focus on unsupervised, extractive, generic, multi-document summarization. Unsupervised, extractive, and generic methods usually adopt the bag-of-words model for term weight calculation, also called term-based methods, which often use term frequency/inverse sentence frequency (TF∗ISF) weighting model and some extended schemes [20].

Term-based methods can be divided into the following categories. The centroid-based methods, as one of the most popular extractive methods, group document sentences into homogeneous clusters, and then select the representative sentences through computing the similarity values between sentences and the centroids of the clusters. For example, MEAD [23] computes the average cosine similarity between sentences and the rest of the sentences in the document collection as the centroid value of a sentence. Gong and Liu [14] presented a method to identify semantically important sentences using the latent semantic analysis (LSA). They first created a term-sentence matrix with each entry representing the weight of a term in its documents. Then they derived the latent semantic information by applying singular value decomposition (SVD). Some methods were proposed based on non-negative matrix factorization (NMF) [22,36]. The NMF-based methods also first create a term-sentence matrix to select meaningful sentences. Other methods were also developed including conditional random fields [32] and hidden Markov model [9].

The graph-based approaches [12,45,50] also belong to extractive summarization. They first produce a similarity graph, in which each node represents a sentence. When the cosine similarity value between a pair of sentences exceeds a threshold, these two sentences are connected by an edge. Erkan and Radev [12] proposed a method, called LexPageRank, which ranks the sentences based on the similarity graph following the well-known PageRank algorithm. Other improved graph-based algorithms have been proposed [6,45,50]. Bollegala et al. [6] presented a bottom-up approach to arrange sentences extracted for multi-document summarization. They defined four criteria, chronology, topical-closeness, precedence and succession, for capturing the association and order of two sentences.

Compared to the term-based methods, some ontology-based approaches [5,16] have been used to produce summaries. Specifically, ontologies have been used to (i) identify the concepts that are either most pertinent to a query [17,43] or most suitable for performing query expansion [27], (ii) model the context in which summaries are generated in a variety of domains, such as business domain [40], disaster management domain [20] and so on. Baralis et al. [5] proposed an ontology-based approach, called Yago-based summarization, which relied on Wikipedia [39] to map the words to non-ambiguous ontological concepts called entities. Yago-based summarization selects document sentences according to the previously assigned entities. Ontology-based approaches are limited in specific application domains, and also it takes much effort to construct the ontologies.

Pattern-mining techniques have been extensively studied for many years in data mining, such as frequent itemset algorithms (Apriori [1], FP-tree [15]), sequential pattern algorithms (PrefixSpan [28], SPADE [47]), closed sequential pattern algorithms (CloSpan [44], AGraP[13], BIDE [38]). As frequent itemsets or frequent patterns include more contextual semantic information than an individual term, they can improve the effectiveness of text mining applications, for example, text classification [2,41,49] and text clustering [19,48]. Algarni and Li [2] calculated term weights based on both their frequencies in documents and their distributions in sequential patterns. Zhang et al. [48] proposed maximum capturing (MC) for text clustering using frequent itemsets. MC can be divided into two components: constructing document clusters and assigning document topics.

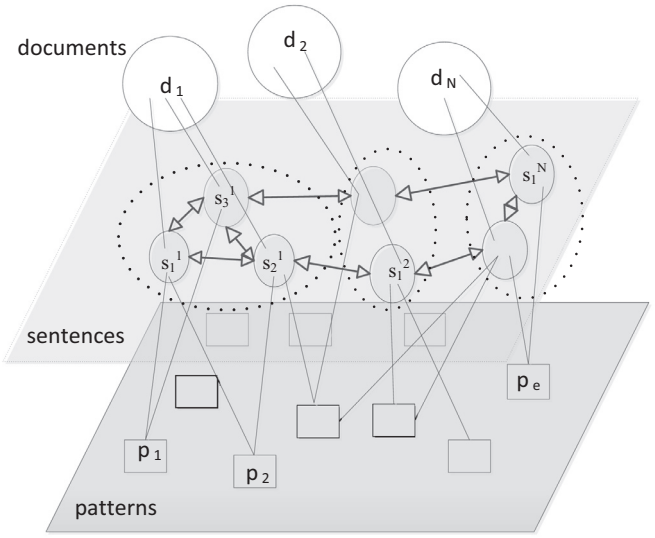


Fig. 1. The sentence-pattern relationships and sentence-sentence relationships.

Baralis et al. [4] presented a method for summarization based on frequent itemsets and the traditional bag-of-word model. This method does not perform very well without considering the low frequency problem. Given a specific topic, a general pattern is usually frequent, and a specific pattern is not [49]. If we decrease the minimum support, many noisy patterns would be generated. So it is a difficult problem how to use frequent patterns to accurately extract a summary. In addition, frequent itemsets do not take the order of terms into account, and include a lot of redundant patterns. In this paper, we conduct a study on multi-document summarization based on closed patterns.

In summary, we aim to generate a summary through closed patterns in this paper. Compared with the term-based methods, our pattern-based method has three advantages. The first is that the representation reduces the dimensionality which only needs these terms from all closed patterns, and term-based methods need all terms in a document collection. The second advantage is that the representation can capture more contextual semantic information than individual terms. The third advantage is that the representation can solve the low frequency problem, because we calculate term weights through the distribution of terms in closed patterns rather than their distributions in the document collection. Compared with the ontology-based methods, our method does not require as much manpower to construct the ontology. Compared with frequent itemsets, closed patterns not only contain the term order information, but also are more concise.

3. Pattern-based summarization

In this section, we discuss how to generate a summary using closed patterns, denoted as pattern-based summarization.

3.1. Overview of pattern-based summarization

Consider a document collection $D=\{d_1,d_2,\ldots,d_i,\ldots,d_N\}$, where d_i represents the ith document of D, N is the number of documents in D. Each document $d_i \in D$ consists of a set of sentences $\{s_1^i,\ldots,s_j^i,\ldots,s_{|d_i|}^i\}$, where s_j^i represents the jth sentence of document d_i. Here, $|d_i|$ is the number of sentences in document d_i. Let U represents all sentences in D, and n be the total number of sentences in D, where $n=\sum_{i=1}^{N}|d_i|$. Let $\{p_1, p_2,\ldots, p_e\}$ be the set of closed patterns from all sentences in D respectively. Fig. 1 illustrates the proposed graph model which is different from the traditional term-based methods. Both sentence ranking and sentence similarity can be calculated through closed patterns belonging to the sentences. The aim is to generate a summary $S=\{s_j^i\}$ $(1 \le I \le N,\ 1 \le j \le l)$ that contains a subset of sentences that are representative of U.

The outline of pattern-based summarization is shown in Fig. 2. Specifically:

(1) Sentence representation using closed patterns. This step obtains all closed sequential patterns from document collection by calling a closed sequential pattern algorithm. Then each sentence is represented by its terms and their corresponding weights that are calculated by accumulating the weight of its covering closed patterns with respect to this sentence.
(2) Sentence ranking. To select the most pertinent and meaningful sentences into the summary, sentences are ranked according to the previous sentence representation.
(3) Sentence selection. The sentences that include more closed patterns with high weight and less similar to other sentences are chosen into the summary. We iteratively select one sentence that owns the highest weight and less similar to the previously selected ones, until reaching the length limit.

3.2. Sentence representation using closed sequential patterns

3.2.1. Mining closed sequential patterns

First, we will mine all closed sequential patterns from a set of all sentences in D. A pattern $p=\{t_1, t_2,\ldots, t_m\}$ is an ordered list of terms. For brevity, a pattern is also written as $p=t_1t_2\ldots t_m$. $p=t_1t_2\ldots t_m$ is a sub-pattern of another pattern $p'=t'_1t'_2\ldots t'_M (m \le M)$, denoted by $p \subseteq p'$ (or p' is a super-pattern of p), if there exists a sequence of positions $1 \le j_1 < j_2 < \cdots < j_m \le n$ s.t. $t_i=t'_{j_i}$ for $i=1, 2,\ldots,m$.

Definition 1 [44]**.** Given a pattern p that is composed of a set of terms, its covering sentences are denoted as $coverSent(p)=\{s|s \in U, p \subseteq s\}$. The support of a *pattern* p is the number of sentences in D containing p, namely $\sup(p)=|coverSent(p)|$, where $|coverSent(p)|$ is the number of sentences.

Definition 2. Given a pattern p, its covering documents are denoted as $coverDoc(p)=\{d|d \in D, p \subseteq d\}$. Here, $|coverDoc(p)|$ is the number of documents.

Definition 3 [49]**.** A pattern p is *frequent* if its support, namely coverSent(p), is more than or equal to a user-specified *minimum support* (*sup*) value.

Example 1. Given a set of sentences from two news reports D in Table 1, and minimum support 3. In preprocessing, all stop words of the input sentences are removed, and all letters are converted into lowercase. We only consider the first occurrence when a sentence has multiple occurrences of one term. These terms that occur more than 3 times are saved in Table 2. Suppose pattern $p=$ "Obama McConnel", we can see there are three sentences "$s_1{}^1$, $s_1{}^2$, $s_2{}^2$" that include this pattern. So this pattern is a frequent pattern. Through pattern mining algorithm, all the frequent patterns are shown in Table 3 with minimum support 3.

Definition 4 [44]**.** A pattern p is a closed pattern if and only if there does not exist any super-pattern p' $(p \subseteq p')$ s.t. $\sup(p)=\sup(p')$, namely, $\sup(p) > \sup(p')$ for all patterns $p' \supset p$. p is a non-closed pattern if and only if there exists a super-pattern p' s.t. $\sup(p)=\sup(p')$.

For example, in Table 3, the support of pattern "Obama leader" is 4, which is a frequent pattern. And the support of its all super-patterns is smaller than 4, so the pattern "Obama leader" is a closed pattern.

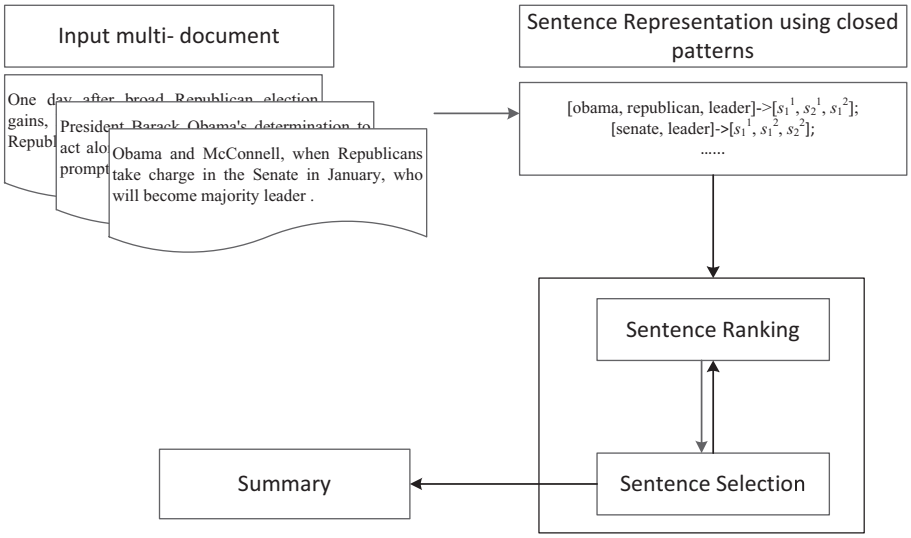


Fig. 2. The outline of pattern-based summarization.

Table 1
A set of sentences from two news reports.

Document	Sentence	Text
d_1	$s_1{}^1$	Obama and McConnell, when republicans take charge in the senate in January, who will become majority leader.
	$s_2{}^1$	President Obama's determination to act alone to change the immigration system promptly drove a wedge Wednesday into the post-election commitment from the president and republican leaders to find common ground under the new political alignment.
d_2	$s_1{}^2$	President Barack Obama and senate republican leader McConnell pledged to try to turn divided government into a force for good rather than gridlock on Wednesday, yet warned of veto showdowns as well.
	$s_2{}^2$	No one was more invested in tuesday's election than senate minority leader McConnell, whose power in Washington is about to vastly expand, and with it his ability to thwart President Obama's agenda.

Table 2
Terms to represent the sentences.

Sentence	Terms
$s_1{}^1$	Obama, McConnel, republican, senate, leader
$s_2{}^1$	President, Obama, republican, leader
$s_1{}^2$	President, Obama, senate, republican, leader, McConnell
$s_2{}^2$	Senate, leader, McConnell, president, Obama

In all, there are 23 frequent patterns. Not all frequent patterns in Table 3 are useful. For example, pattern {leader} always occurs with term "Obama", i.e., the shorter pattern {Obama} is always part of the longer pattern {leader, Obama} in all its covering sentences. Therefore, the shorter one, {Obama}, is a redundant pattern, and we only keep the longest patterns, namely closed patterns. After pruning non-closed patterns, we only have four closed patterns, which are shown in bold font.

We shall point out that a pattern is different from N-gram or term co-occurrence. N-gram is a contiguous sequence of n terms from a given set of sentences, and pattern allows flexible gaps between items. Term co-occurrence only considers the frequency between two words, and a pattern is a set of ordered items that appear frequently together in a set of sentences.

3.2.2. Sentence representation

In this section, we will represent all sentences using closed patterns. At first, we introduce the composition operation $\oplus$. Let ts be a term-weight pair composing of a set of terms and their weights, such as $\{(t_1,a_1),\ldots,(t_i,a_i),\ldots,(t_x,a_x)\}$, where t_i denotes a single term, and a_i is its weight. A closed pattern can be expressed as a term-weight pair.

Definition 5. Let $p_i = \{(t_1,\ldots,t_i,\ldots,t_m),\{s_1,\ldots,s_j,\ldots,s_y\}\}$ denote a closed pattern, where m is the number of the terms, and y is the number of its covering sentences. The weight of the closed pattern can be calculated as follows:

$$w(p_i) = |\text{coverSent}(p_i)| * |coverDoc(p_i)|/N \quad (1)$$

Table 3
Frequent patterns and their covering sentences with $sup = 3$.

Frequent patterns	Covering sentences	Frequent patterns	Covering sentences
Obama	$s_1{}^1$, $s_2{}^1$, $s_1{}^2$, $s_2{}^2$	Obama republican	$s_1{}^1$, $s_2{}^1$, $s_1{}^2$
McConnel	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$	Senate leader	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$
Senate	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$	Leader president	$s_2{}^1$, $s_1{}^2$, $s_2{}^2$
Leader	$s_1{}^1$, $s_2{}^1$, $s_1{}^2$, $s_2{}^2$	Obama president	$s_2{}^1$, $s_1{}^2$, $s_2{}^2$
President	$s_2{}^1$, $s_1{}^2$, $s_2{}^2$	Obama senate leader	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$
Republican	$s_1{}^1$, $s_2{}^1$, $s_1{}^2$	Obama McConnel leader	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$
Obama McConnel	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$	Leader McConnel senate	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$
Obama senate	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$	Obama McConnel senate	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$
McConnel leader	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$	Obama republican leader	$s_1{}^1$, $s_2{}^1$, $s_1{}^2$
McConnel senate	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$	Obama president leader	$s_2{}^1$, $s_1{}^2$, $s_2{}^2$
Obama leader	$s_1{}^1$, $s_2{}^1$, $s_1{}^2$, $s_2{}^2$	Obama leader McConnel senate	$s_1{}^1$, $s_1{}^2$, $s_2{}^2$
Republican leader	$s_1{}^1$, $s_2{}^1$, $s_1{}^2$		

Algorithm 1 SentRep(*U, sup*).

Input: a collection document D that consists of a set of sentences $U = \{s_1, s_2, \ldots, s_n\}$, minimum support *sup*;
Output: the term-weight pairs of all sentences TW = {tw(s_1), tw(s_2), ...,tw(s_n)}.
1: $P \leftarrow$ closedMining(*U, sup*);
2: **for** each $p_i \in P$ **do**
3: w(p_i)=|coverSent(p_i)|*|*coverDoc*(p_i)|/*N*;
4: **end**
5: *TW*← ∅;
6: **for each** $p_i \in P$ **do**
7: **for each** *sentence* $s_j \in$ coverSent(p_i)
8: $tw(s_j) \leftarrow tw(s_j) \oplus w(p_i)$;
9: **end**
10: **end**
11: **return** *TW*;

where |coverSent(p_i)| is the number of p_i's covering sentences, |*coverDoc*(p_i)| is the number of p_i's covering documents, and N is the number of all documents. Through this formula, one closed pattern existing in more documents will have a higher weight than the other closed pattern existing in fewer documents under the same support.

For example, closed pattern {Obama, republican, leader} and its covering sentences (s_1^1, s_2^1, s_1^2) in Table 3 can be represented as one term-weight pair, tw={(Obama, 3), (republican, 3), (leader, 3)}, where 3 is the weight of this closed pattern according to (1).

Definition 6. Let $tw_1 = \{(t_1,a_1),\ldots,(t_i,a_i),\ldots,(t_x,a_x)\}$ and $tw_2 = \{(w_1,b_1),\ldots, (w_j,b_j),\ldots,(w_y,b_y)\}$ be two term-weight pairs associated to two patterns, the composition operation $\oplus$ between tw_1 and tw_2, $tw_1 \oplus tw_2$ denoted as,

$$tw_1 \oplus tw_2 = \{(t_i, a_i + b_j) | (t_i, a_i) \varepsilon tw_1, (w_j, b_j) \varepsilon tw_2, t_i = w_j\} \cup \left\{(t_i, a_i) | (t_i, a_i) \varepsilon tw_1, t_i \notin \bigcup_{j=1}^{y} w_y\right\} \cup \left\{(w_j, b_j) | (w_j, b_j) \varepsilon tw_2, w_j \notin \bigcup_{i=1}^{x} t_i\right\} \quad (2)$$

The composition operation is interchangeable, namely, $tw_1 \oplus tw_2 = tw_2 \oplus tw_1$. Here, $tw_1 \oplus \emptyset = tw_1$, where $\emptyset$ represents null.

For example, if $tw_1 = \{(t_1,3),(t_2,2),(t_4,4)\}$, $tw_2 = \{(t_2,5),(t_3,2),(t_5,1)\}$, we have $tw_1 \oplus tw_2 = \{(t_1,3),(t_2,7), (t_3,2), (t_4,4), (t_5,1)\}$.

Let $\{tw_1, tw_2,\ldots, tw_e\}$ be the term-weight pairs of closed patterns from all sentences U in D. Here, we only use a closed pattern whose size is more than 1, because it can capture the semantic relationship of terms. Let $\{tw_{i_1}, tw_{i_2}, \ldots, tw_{i_r}\}$ be the set of closed patterns with respect to sentence s_i, where $1 \le i_1, i_2, \ldots, i_r \le e$. Finally, the representation of sentence s_i can be obtained using the following formula:

$$tw(s_i) = tw_{i_1} \oplus tw_{i_2} \oplus \cdots \oplus tw_{i_r} \quad (3)$$

For example, using Table 3, the term-weight pairs of all closed patterns are expressed as,

tw_1 = {(Obama, 4), (leader, 4)}, tw_2 = {(Obama, 3), (republican, 3), (leader, 3)}, tw_3 = {(Obama, 3), (president, 3), (leader, 3)}, tw_4 = {(Obama, 3), (leader, 3), (McConnel, 3), (senate, 3)}.

Definition 7. A sentence is a set of term-weight pairs that appear in this sentence.

For example, using tw_1, tw_2, tw_3 in Table 3, we have s_1^1 = {tw_1, tw_2, tw_4}, s_2^1 = {tw_1, tw_2, tw_3}, s_1^2 = {tw_1, tw_2, tw_3, tw_4}, s_2^2 = {tw_1, tw_3, tw_4}. Then we can obtain the representation of each sentence below,

tw(s_1^1) = {(senate, 3), (leader, 10), (Obama, 10), (republican, 3), (McConnel, 3)},
tw(s_2^1) = {(Obama, 10), (republican, 3), (president, 3), (leader, 10)},
tw(s_1^2) = {senate, 3), (McConnel, 3), (leader, 13), (Obama, 13), (republican, 3), (president, 3)},
tw(s_2^2) = {(senate, 3), (leader, 10), (Obama, 10), (president, 3), (McConnel, 3)}.

Algorithm 1 describes the process of obtaining the set of sentence representation. For all sentences in a document collection, all closed patterns P are discovered using closed pattern mining algorithm (e.g. CloSpan [44], AGraP [13], SPMW [42]) in Step 1, which returns all closed patterns and their corresponding covering sentences. The weight of each pattern in P can be calculated using (1) from Steps 2 to 4. Thereafter, from Steps 6 to 10, the representation of each sentence in U can be obtained using (3).

Let n be the number of all sentences, e be the number of closed patterns, l be the average number of the covering sentences of each closed pattern, and k be the average number of terms of each sentence. Step 1 calls the algorithm of closed pattern mining that takes O(*CP*), where *CP* is the time complexity of the algorithm of closed pattern mining. It takes O(el) from Steps 2 to 4 and O(n) at Step 5. The $\oplus$ operation takes O(k^2) at most. So it takes O(elk^2) from Steps 6 to 10. Therefore, the time complexity of sentence representation is O($CP + n + (el + elk^2)$ = O($CP + n + elk^2$).

3.3. Sentence ranking

Each sentence has been represented using term-weight pairs. In this step, pattern-based summarization ranks all sentences according to their sentence representation. The basic idea of sentence ranking is that these sentences will have high scores that contain more closed patterns with high weight. Let s_j^i be the jth sentence in document d_i. Hence, the score of the sentence s_j^i is denoted as score(s_j^i) and defined as follows:

$$score\left(s_j^i\right) = \frac{weight\left(s_j^i\right)}{|s_j^i|} \times \left(1 - \frac{j-1}{|d_i|}\right), \quad (4)$$

where weight(s_j^i) is a weight assigned to the sentence s_j^i using the weighting scheme described in Section 3.4. $|s_j^i|$ is the number of words in sentence s_i. The aim is to get the highest average weight that is calculated as weight(s_j^i) divided by $|s_j^i|$. By multiplying "$1-\frac{j-1}{|d_i|}$", we integrate the position of each sentence in its document into the sentence ranking. Intuitively, the starting sentences in a document contain more new information than the following sentences. Through the second part of Eq. (4), the first sentence of a document gets maximum value, and the last sentence gets the minimum value. For example, in Table 1, the value of s_1^1 is 1,

Algorithm 2 PatSum(D, sup, L).

Input: a collection document D that consisted of a set of sentences $U = \{s_1, s_2, \ldots, s_n\}$, minimum support *sup*, limitation length L;
Output: Summary (S)
1: $TS \leftarrow$ SentRep(D, *sup*);
2: R$\leftarrow$0;
3: **for each** $p_l \in P$ **do**
4: **for** any two *sentences* s_i, $s_j \in$ cover *Sent*(p_l)
5: $R_{ij} \leftarrow R_{ij} + |p_l| \times w(p_l)$;
6: **end**
7: **end**
8: $S \leftarrow \emptyset$;
9: $len \leftarrow 0$; // record the length of S
10: **when** $len < L$
11: $highScore \leftarrow 0$; // record the highest score
12: $id \leftarrow 0$; // record the id of the sentence owns the highest score
13: **for each** $s_i \in U$ **and** $i \notin S$ **do**
14: $weight(s_i) = \lambda \sum_{j=1}^{k} a_j^i - (1-\lambda) \max_{s_j \in S} R_{ij}$;
15: score(s_i)=formula 4;
16: **if** *score*(s_i) > highScore then $id = i$, $highScore$ = score(s_i);
17: **end for**
18: $S = S \cup \{id\}$;
19: $len = len + |s_{id}|$;
20: **end when**
21: **return** S;

and the value of $s_2{}^1$ is 0.5 when only computing the second part of Eq. (4).

From the closed patterns, we know that they represent these terms with high frequency in the document collection. And the weight of each sentence is decided by the number of closed patterns belonging to the sentence and the support of each closed pattern. Thus, we can draw a conclusion that these sentences containing more closed patterns have high scores. Note that the sentences that do not contain any closed patterns have minimal scores (i.e., 0).

3.4. Sentence selection

Given the result of sentence ranking, we focus on generating the summary of the document collection considering both content coverage and non-redundancy, until a given length of the summary is reached. For reducing redundancy in the summary, some methods measure the similarity of next candidate sentence to that of previously selected ones, and select it if its similarity is below a threshold [31]. The most popular method is maximal marginal relevance (MMR), which was first introduced in query-based summarization [8]. At each iteration, MMR tries to select sentences that are dissimilar from the ones already selected. As our method is not query-based, we adapt the former selection strategy.

The most popular method of computing sentence similarity in MMR is the term frequency–inverse sentence frequency (TF–ISF) of the candidate terms [8,23]. As sentences are very short comparing to text, most terms only have one occurrence in a sentence. Therefore, TF of TF–ISF cannot play a role in computing sentence similarity. So TF–ISF can hardly make a distinction without considering the context information of each term. We thus propose a new method of computing sentence similarity using their common closed patterns.

Here, let $U = \{s_1, s_2, \ldots, s_n\}$ be all sentences in D. We use an adjacency matrix R to describe sentence–sentence similarity with each entry corresponding to the weight of a link in Fig. 1. $R = [R_{ij}]_{n \times n}$ is defined as follows:

$$R_{ij} = \sum_{l=1}^{e} \{|p_l| \times w(p_l) | p_l \in s_i \cap p_l \in s_j\}, \tag{5}$$

where R_{ij} is the similarity value of sentence s_i and sentence s_j, and $|p_l|$ is the number of terms in pattern p_l. Through this formula, the similarity value between sentences is generated based on their common closed patterns and the distribution of these closed patterns.

Let s_i be an arbitrary sentence in U whose representation is $ts(s_i) = \{(t_1^i, a_1^i), \ldots, (t_k^i, a_k^i)\}$, and S is the set of sentences already selected. The weight of each sentence is calculated as:

$$\text{weight}(s_i) = \lambda \sum_{j=1}^{k} a_j^i - (1-\lambda) \max_{s_j \in S} R_{ij}, \tag{6}$$

where λ is a user-specified parameter that controls the tradeoff between content coverage and non-redundancy: if $\lambda = 1$, the content coverage is highlighted; if $\lambda = 0$, only the non-redundancy is considered. The impact of λ on the summarization performance is discussed in Section 4.

Pattern-based summarization adopts (4) to rank sentences. Using the ranking function, we iteratively select the top $|S|$ sentences having the highest scores, where the length of these $|S|$ sentences must be less than or equal to the length limitation L.

The main process of pattern-based summarization (PatSum) is illustrated in Algorithm 2. Step 1 calls SentRep (see Algorithm 1). Steps 3 to 7 compute sentence similarity, which takes $O(ek^2)$, where e is the number of the closed patterns, and k is the average number of terms of each sentence. Steps 13 to 17 rank sentence, whose time complexity is $O(n)$. Steps 10 to 20 take $O(|S|n)$, where $|S|$ be the number of sentences in summary. The time complexity of Algorithm 2 is $O(ek^2 + |S|n)$.

Based on the time complexity of Algorithms 1 and 2, the total time complexity of PatSum is $O(CP + n + elk^2 + ek^2 + |S|n) = O(CP + elk^2 + |S|n)$.

4. Experimental results

We implemented pattern-based summarization in JAVA. All the experiments are performed on a Windows machine with an Intel 2.9 GHz CPU and 8GB memory.

4.1. Experiment setting

In order to assess the effectiveness of our method for multi-document summarization, we use the standard benchmark

Table 4
ROUGE values for methods using DUC2004 dataset.

Summarizer		ROUGE-2			ROUGE-4		
	Recall	Precision	F-measure	Recall	Precision	F-measure	
Top ranked DUC2004 peers	peer124	0.083*	0.081*	0.082*	0.012*	0.012*	0.012*
	peer65	0.092*	0.091*	0.091*	0.015*	0.015*	0.015*
	peer19	0.080*	0.081*	0.080*	0.010*	0.010*	0.010*
	peer44	0.076*	0.079*	0.077*	0.012*	0.013*	0.012*
	peer81	0.081*	0.079*	0.080*	0.013*	0.013*	0.013*
	peer104	0.086*	0.084*	0.085*	0.011*	0.011*	0.011*
TexLexAn		0.067*	0.067*	0.067*	0.007*	0.007*	0.007*
ItemSum		0.083*	0.085*	0.084*	0.012*	0.014*	0.014*
Baseline		0.092*	0.091*	0.092*	0.014*	0.014*	0.014*
Yago		0.095*	0.094*	0.095*	0.017*	0.017*	0.017*
MSSF		0.098	0.098	0.098	0.017*	0.017*	0.017*
PatSum		**0.102**	**0.102**	**0.102**	**0.020**	**0.020**	**0.020**
DUC2004 Humans	A	0.088*	0.092*	0.090*	0.009*	0.010*	0.010*
	B	0.091*	0.096*	0.092*	**0.013***	**0.013***	**0.013***
	C	0.094*	0.102	0.098*	0.011*	0.012*	0.012*
	D	0.100	**0.106**	0.102	0.010*	0.010*	0.010*
	E	0.094*	0.099	0.097	0.011*	0.012*	0.012*
	F	0.086*	0.090*	0.088*	0.008*	0.009*	0.009*
	G	0.082*	0.087*	0.084*	0.008*	0.008*	0.007*
	H	**0.101**	0.105	**0.103**	0.012*	**0.013***	0.012*

DUC2004 dataset, which is from Document Understanding Conference (DUC) (http://duc.nist.gov) for generic summarization evaluation. There are 50 document clusters in DUC2004, where each cluster consists of 10 English documents. To evaluate automatic methods, DUC2004 provides at least four human-generated summaries in each cluster. Participants to the DUC2004 contest submitted their own summaries and were evaluated against human-generated summaries.

The preprocessing of DUC2004 dataset is as follows. All documents are segmented into sentences using a script distributed by DUC. In our experiments, stop-words are removed using the stop-word list[1] and the remaining words are stemmed using Porter's scheme[2].

We use the ROUGE toolkit [21] (version 1.5.5) to evaluate the summarization performance, which is adopted by DUC as the official metric for document summarization[3]. ROUGE evaluates the quality of a summary by counting the unit overlaps between the candidate summary and a set of human-generated summaries. The summary that achieves the highest ROUGE score is considered to be the most similar to the human-generated summary. To carry out a fair comparison, we normalize the generated summaries by truncating each of them at 665 bytes before using the ROUGE toolkit. As previously done in these literatures [4,5], we also choose the ROUGE-2 and ROUGE-4 representative scores [21] in this paper.

4.2. Evaluation results on DUC2004

We compare our method with systems participated in DUC2004, four newly proposed methods, one commonly used method, and 8 human-generated summaries provided by DUC2004.

(1) 35 methods: automatic multi-document summarization participated in DUC2004.
(2) 8 human-generated summaries: provided by the DUC2004.
(3) TexLexAn : one widely used open source text summarization [35].
(4) ItemSum: based on frequent itemsets and tf-idf [4].
(5) MSSF: based on submodular functions [18].
(6) Yago-based summarization (Yago): relies on an ontology-based selection of the sentences [5].
(7) Baseline: A baseline version of Yago-based summarization, which adopts an established term relevance evaluator, i.e., TF–ISF score, rather than ontology-based score [5].
(8) Pattern-based summarization (PatSum): proposed by this paper.

For all existing methods we tune the algorithm parameters to their average best performance by following the instructions given by the authors. For pattern-based summarization (PatSum), two parameters λ and minimal support (*sup*) are set to 0.5 and 4, respectively. In Section 4.4, we analyze the impact of both parameters on pattern-based summarization performance in detail.

As there are 35 methods provided by DUC2004, we only select six methods with the highest scores. Table 4 shows the results of all methods including the six top-ranked methods in the DUC2004 contest, TexLexAn, ItemSum, Baseline, Yago-based summarization, MSSF, pattern-based method (PatSum), and eight human-generated summaries.

To validate the statistical significance of the pattern-based summarization performance improvement against other methods, we performed the paired t-test [10] at 95% significance level for all of the evaluated measures. If a method is statistically worse than PatSum, it is marked with a star (*). The bold entries represent the best performing methods separately for automatic and human-generated summaries in terms of each ROUGE evaluator and measure.

As shown in Table 4, we can see that the performance of PatSum is better than other methods in terms of the results of ROUGE-2 (Recall, Precision, and F-measure) and ROUGE-4 (Recall, Precision, and F-measure) metrics. Based on all the ROUGE metrics, PatSum achieves the best performance (0.102 and 0.020). Among term-based methods, the closest method to PatSum is MSSF that takes advantage of the submodularity to modify the general greedy algorithm when choosing sentences. Yago-based summarization (Yago) that relies on Yago ontological knowledge base [34] also has better performance. Ontology has been used to identify the entities in Yago-based summarization. Although PatSum does not adopt additional knowledge, PatSum still outperforms Yago. PatSum has better performance than a baseline version of Yago-based

[1] ftp://ftp.cs.cornell.edu/pub/smart/english.stop.
[2] http://www.tartarus.org/martin/Porter-Stemmer.
[3] The provided command is: ROUGE-1.5.5.pl -e data -x -m -2 4 -u -c 95 -r 1000 -n 4 -f A -p 0.5 -t 0 -d -a.

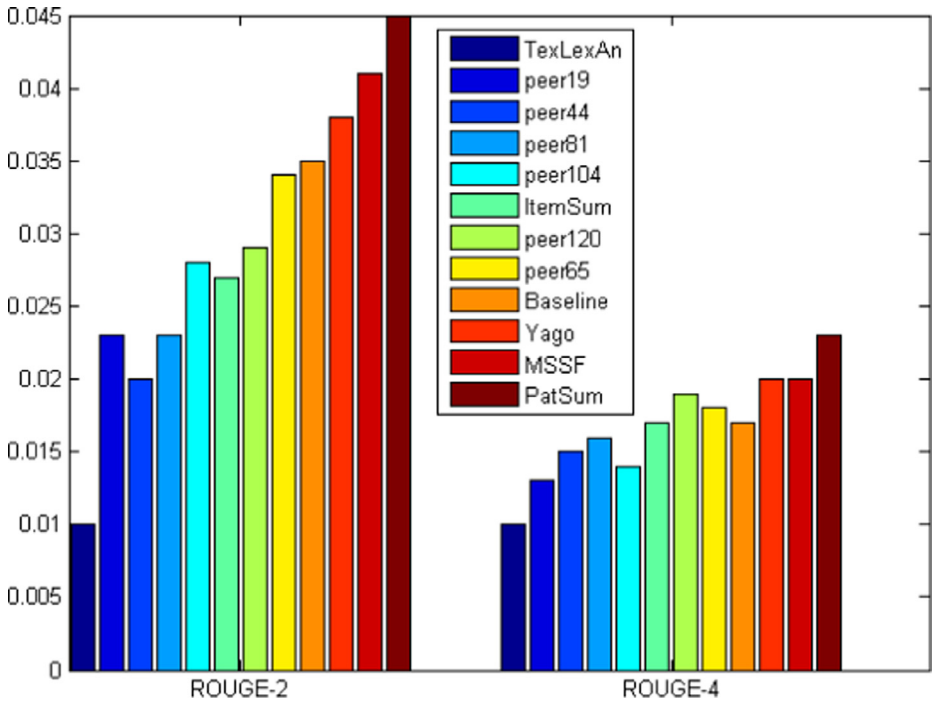


Fig. 3. Comparison of the methods in terms of ROUGE-2 and ROUGE-4 F-measures.

Table 5
Comparison of PatSum with other automatic methods.

Summarizer		ROUGE-2			ROUGE-4		
		Recall	Precision	F-measure	Recall	Precision	F-measure
Top ranked DUC2004 peers	peer124	+0.19	+0.21	+0.20	+0.08	+0.08	+0.08
	peer65	+0.11	+0.12	+0.12	+0.33	+0.33	+0.33
	peer19	+0.28	+0.26	+0.28	+1.00	+1.00	+1.00
	peer44	+0.34	+0.29	+0.32	+0.67	+0.54	+0.67
	peer81	+0.26	+0.29	+0.28	+0.54	+0.54	+0.54
	peer104	+0.19	+0.21	+0.20	+0.82	+0.82	+0.82
TexLexAn		+0.52	+0.52	+0.52	+1.86	+1.86	+1.86
ItemSum		+0.23	+0.20	+0.21	+0.67	+0.43	+0.43
Baseline		+0.11	+0.12	+0.11	+0.43	+0.43	+0.43
Yago		+0.07	+0.09	+0.07	+0.18	+0.18	+0.18
MSSF		+0.04	+0.04	+0.04	+0.18	+0.18	+0.18

summarization compared to Yago. It means that ontology is useful for Yago. Hence, pattern-based ranking and selection strategies are more effective than other strategies.

We also compare PatSum with the six top-ranked methods in the DUC2004 contest. It is clear that PatSum outperforms all the methods from DUC2004. We further compare our results with the eight human-generated summaries. We see that in terms of ROUGE-2 F-measure, only one out of eight outperforms our method, and none outperforms our method in terms of ROUGE-4.

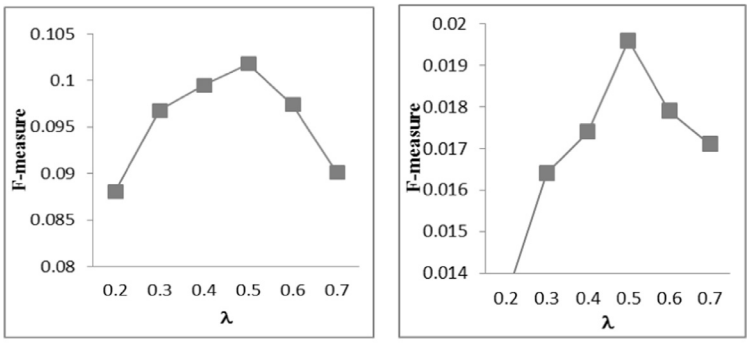
For better demonstration of the results, Fig. 3 visually illustrates the comparison. Note that we subtracted the ROUGE scores of the worst method TexLexAn in all automatic methods and added the number 0.01 in these figures, thus the difference can be observed more clearly. We show ROUGE-2 and ROUGE-4 F-measures in Fig. 3.

4.3. Result analysis

A more detailed comparison is discussed in this section. We adopt the relative improvement that is defined as $(b - a)/a$, when b is compared to a. In our experiments, b is PatSum, and a is the method we compare with. Table 5 shows the results of PatSum compared with other automatic methods in terms of ROUGE-2 and ROUGE-4.

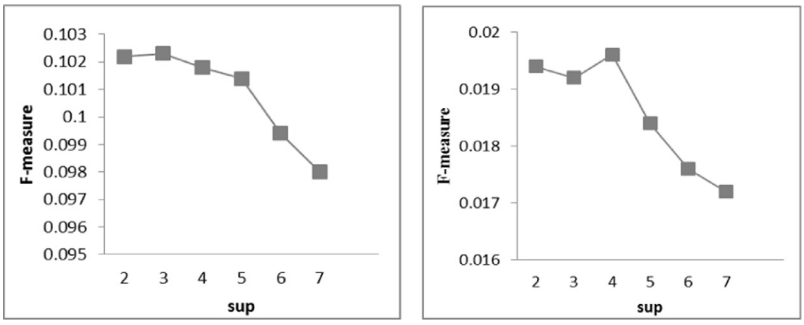
In Table 5 "+" means the result outperforms and "−" means the opposite. Compared to ontology-based method, we see that PatSum improves the performance by 0.07, 0.09, 0.07, 0.18, 0.18 and 0.18, respectively, compared with the state-of-the-art Yago-based summarization. Compared to the best method MSSF of term-based methods, PatSum improves the performance by 0.04, 0.04, 0.04, 0.18, 0.18 and 0.18, respectively. In addition, in Table 5, we have the following observations:

(1) Although ItemSum exploits frequent itemsets, PatSum is better than ItemSum . Usually a general pattern occurs more frequently than a specific pattern. PatSum takes advantage of this property and achieves better performance than ItemSum. The promising results can be explained that PatSum can deal with the problem of low frequency. We calculate sentence weights with respect to the distribution of closed patterns rather than their distributions in the document collection. In Section 4.5, we will show the results when using frequent itemsets based on our strategy. We can see that PatSum using frequent itemsets outperforms ItemSum, which clearly shows that the low frequency information is very important for multi-document



(a) ROUGE-2 F-measure (b) ROUGE-4 F-measure

Fig. 4. Impact of λ on the pattern-based method performance under *sup* = 4.



(a) ROUGE-2 F-measure (b) ROUGE-4 F-measure

Fig. 5. Impact of *sup* on the pattern-based method performance under λ = 0.5.

summarization. In addition, there are a lot of redundant patterns among frequent itemsets compared to closed patterns.

(2) PatSum performs better than Yago, and Yago outperforms most of term-based methods (DUC2004 peers, TexLexAn, Baseline). Because term-based methods have the problems of polysemy and synonymy, Yago captures the semantic information of terms. As a document collection usually comes from the same topic, Yago relies on Wikipedia utilizes too many resources and ignores the context information of the current document collection. Yago is restricted to specific application domains because it cannot attain the semantic meanings for terms that do not belong to its ontology. Meanwhile, the construction of ontology can be prohibitively expensive. On the other hands, PatSum can effectively capture the correlations among the words, and also do not require additional resources. Therefore, PatSum outperforms Yago.

(3) PatSum outperforms all term-based methods including TexLexAn, Baseline, and MSSF. Among the term-based methods, MSSF has the best performance which greedily chooses the sentences using submodular functions. But, compared to term-based methods, PatSum outperforms all these methods, because term-based methods suffer from the problems of polysemy and synonymy.

4.4. Parameter setting

The user-specified parameters (λ and *sup*) could affect the performance of PatSum. Therefore, for analyzing their impact on PatSum, we carry out the experiments with different parameters. The experiments are divided into two categories: one by varying the value of λ in the range [0.2, 0.7] under *sup* = 4; the other by varying the value of *sup* in the range [2, 7] under λ = 0.5.

In Fig. 4, the performance of PatSum is shown, as the value of the parameter λ changes. When increasing the value of λ, the performance increases first, and declines finally. The function of λ controls the tradeoff between content coverage and non-redundancy. If the value of λ is a small number, the performance of PatSum has receded dramatically, because PatSum does not reduce much information redundancy through Formula 6. If the λ value surpasses a certain threshold value (e.g., 0.5), it is difficult to remove the redundancy among sentences. As both coverage and non-redundancy are critical for summarization, the best performance through the experiments is achieved by setting λ equals 0.5 under *sup* = 4 for a good trade-off between coverage and non-redundancy.

Fig. 5 shows the results of pattern-based method by varying the value of minimal support (*sup*). We can see that the best performance is achieved when *sup* = 4. Furthermore, the results remain relatively stable when λ ranges between 2 and 7. Since our strategy can solve the problem of low frequency, it means that PatSum is hardly influenced by the minimum support. When *sup* is sufficiently large, the results are lower than the original results, because only few patterns are generated when support is greater than 6.

4.5. Different representation

In our model, we represent all sentences using closed patterns, and compute their scores. We have conducted some experiments with frequent itemsets, closed itemsets and frequent patterns to study whether they influence the performance of the algorithm. Table 6 illustrates the experimental results, and the best results of each representation are shown in bold font.

We can see that all these strategies are quite effective compared with the state-of-the-art methods. The method using frequent patterns outperforms the method using frequent itemsets, and PatSum outperforms closed itemsets. The reason is that, term order is useful for the summarization, which is not considered by itemsets.

Table 6
Performance with different conditions.

Summarizer	Parameter	ROUGE-2			ROUGE-4		
		Recall	Precision	F-measure	Recall	Precision	F-measure
Frequent itemsets	$\lambda = 0.4$, sup $= 6$	0.098	0.097	0.098	0.018	0.018	0.018
	$\lambda = 0.4$, sup $= 7$	0.099	0.098	0.098	0.017	0.017	0.017
	$\lambda = 0.5$, sup $= 6$	**0.010**	**0.010**	**0.010**	**0.019**	**0.019**	**0.019**
	$\lambda = 0.5$, sup $= 7$	0.097	0.097	0.097	0.018	0.018	0.018
	$\lambda = 0.6$, sup $= 6$	0.093	0.092	0.092	0.018	0.018	0.018
	$\lambda = 0.6$, sup $= 7$	0.090	0.090	0.090	0.016	0.016	0.016
Frequent patterns	$\lambda = 0.4$, sup $= 4$	0.098	0.098	0.098	0.018	0.018	0.018
	$\lambda = 0.4$, sup $= 5$	0.099	0.098	0.099	0.018	0.018	0.018
	$\lambda = 0.5$, sup $= 4$	**0.010**	**0.010**	**0.010**	**0.019**	**0.019**	**0.019**
	$\lambda = 0.5$, sup $= 5$	0.098	0.098	0.098	0.019	0.019	0.019
	$\lambda = 0.6$, sup $= 4$	0.096	0.096	0.096	0.019	0.019	0.019
	$\lambda = 0.6$, sup $= 5$	0.096	0.096	0.096	0.018	0.018	0.018
Closed itemsets	$\lambda = 0.4$, sup $= 4$	0.099	0.098	0.098	0.018	0.018	0.018
	$\lambda = 0.4$, sup $= 5$	0.098	0.098	0.098	0.018	0.018	0.018
	$\lambda = 0.5$, sup $= 4$	**0.101**	**0.101**	**0.101**	**0.019**	**0.019**	**0.019**
	$\lambda = 0.5$, sup $= 5$	0.099	0.099	0.099	0.019	0.019	0.019
	$\alpha\lambda = 0.6$, sup $= 4$	0.098	0.098	0.098	0.019	0.019	0.019
	$\lambda = 0.6$, sup $= 5$	0.099	0.099	0.099	0.020	0.020	0.020
PatSum	**$\lambda = 0.3$, sup $= 5$**	**0.102**	**0.102**	**0.102**	**0.020**	**0.020**	**0.020**

Meanwhile, these two methods using closed itemsets or patterns are superior to the other two methods. This suggests that removing the redundant patterns is critical for summarization. Taking these two aspects into consideration, PatSum chooses closed patterns for summarization.

5. Conclusion

Term-based methods suffer from the problems of polysemy and synonymy, and ontology-based approaches are often restricted in specific application domains. In this paper, we propose a novel method for multi-document summarization based on closed patterns, namely pattern-based summarization (PatSum), which simultaneously considers content coverage and non-redundancy. Compared to term-based methods, PatSum can capture more contextual semantic information than individual terms. Compared to ontology-based methods, PatSum not only can capture the correlations among the words, but also do not require additional information. Experimental results have shown that pattern-based summarization outperforms not only term-based methods and all participating systems on DUC2004 datasets, but also ontology-based method. Multi-document summarization can extract a particular individual's opinions in the form of closed patterns, from this individual's documents shared in social networks, hence provides a useful tool for further analyzing the individual's influence in group activities. In the future, we plan to extend our work to query-based multi-document summarization and multi-document update summarization, with applications to behavior and influence analysis in social networks.

Acknowledgments

This research is partially supported by the National 973 Program of China (no. 2013CB329604), the National Natural Science Foundation of China (no. 61229301), and the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT) of the Ministry of Education, China (no. IRT13059).

References

[1] R. Agrawal, R. Srikant, Fast algorithms for mining association rules in large databases, in: Proceedings of the Twentieth International Conference on Very Large Data Bases (VLDB '94), 1994, pp. 478–499.
[2] A. Algarni, Y. Li., Mining specific features for acquiring user information needs, Advances in Knowledge Discovery and Data Mining, 7818, Lecture Notes in Computer Science, 2013, pp. 532–543.
[3] R.M. Alguliev, R.M. Aliguliyev, N.R. Isazade, Multiple documents summarization based on evolutionary optimization algorithm, Expert Syst. Appl. 40 (5) (2013) 1675–1689.
[4] E. Baralis, L. Cagliero, A. Fiori, S. Jabeen, Multi-document summarization exploiting frequent itemsets, in: Proceedings of the ACM Symposium on Applied Computing, 2012, pp. 782–786.
[5] E. Baralis, L. Cagliero, S. Jabeen, A. Fiori, S. Shah, Multi-document summarization based on the Yago ontology, Expert Syst. Appl. 40 (17) (2013) 6976–6984.
[6] D. Bollegala, N. Okazaki, M. Ishizuka, A bottom-up approach to sentence ordering for multi-document summarization, Inf. Process. Manag. 46 (1) (2010) 89–109.
[7] S. Brin, L. Page, The anatomy of a large-scale hypertextual Web search engine, Comput. Netw. ISDN Syst. 30 (1) (1998) 107–117.
[8] J.G. Carbonell, J. Goldstein, The use of MMR, diversity-based re-ranking for reordering documents and producing summaries, in: Proceedings of the Twenty-first Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, 1998, pp. 335–336.
[9] J. Conroy, D. O'Leary, Text summarization via hidden markov models, in: Proceedings of the Twenty-fourth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2001, pp. 406–407.
[10] T.G. Dietterich, Approximate statistical test for comparing supervised classification learning algorithms, Neural Comput. 10 (7) (1998) 1895–1923.
[11] T.D.T. Do, A. Termier, A. Laurent, B. Negrevergne, B. Omidvar-Tehrani, S. Amer-Yahia, PGLCM: Efficient parallel mining of closed frequent gradual itemsets, Knowl. Inf. Syst. 43 (3) (2015) 497–527.
[12] G. Erkan, D.R. Radev, LexRank: Graph-based lexical centrality as salience in text summarization, J. Artif. Intell. Res. 22 (1) (2004) 457–479.
[13] M. Flores-Garrido, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, AGraP: An algorithm for mining frequent patterns in a single graph using inexact matching, Knowl. Inf. Syst. 44 (2) (2015) 1–22.
[14] Y. Gong, X. Liu, Generic text summarization using relevance measure and latent semantic analysis, in: Proceedings of the Twenty-fourth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2001, pp. 19–25.
[15] J. Han, J. Pei, Y. Yin, Mining frequent patterns without candidate generation, in: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '00), 2000, pp. 1–12.
[16] L. Hennig, W. Umbrath, R. Wetzker, An ontology-based approach to text summarization, in: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT '08), 2008, pp. 291–294.
[17] A. Kogilavani, B. Balasubramanie, Ontology enhanced clustering based summarization of medical documents, Int. J. Recent Trends Eng. 1 (1) (2009) 546–549.
[18] J. Li, Lei Li, Tao Li., Multi-document summarization via submodularity, Appl. Intell. 37 (3) (2012) 420–430.
[19] Y.J. Li, S.M. Chung, J.D. Holt, Text document clustering based on frequent word meaning sequences, Data Knowl. Eng. 64 (1) (2008) 381–404.
[20] L. Li, D. Wang, C. Shen, T. Li., Ontology-enriched multi-document summarization in disaster management, in: Proceedings of the Thirty-third International ACM SIGIR Conference on Research and Development in Information Retrieval, 2010, pp. 819–820.

[21] C.Y. Lin, ROUGE: A package for automatic evaluation summaries, in: Proceedings of the 2004 Workshop on Text Summarization Branches Out, Barcelona, Spain, 2004, pp. 74–81.
[22] J.H. Lee, S. Park, C.M. Ahn, D. Kim, Automatic generic document summarization based on non-negative matrix factorization, Inf. Process. Manag. 45 (1) (2009) 20–34.
[23] D.R. Radev, H. Jing, M. Styś, Centroid-based summarization of multiple documents, Inf. Process. Manag. 40 (6) (2004) 919–938.
[24] T. Ma, X. Wan, Multi-document summarization using minimum distortion, in: Proceedings of the 2010 IEEE International Conference on Data Mining, Sydney, Australia, 2010, pp. 354–363.
[25] N.R. Mabroukeh, C.I. Ezeife, A taxonomy of sequential pattern mining algorithms, ACM Comput. Surv. 43 (1) (2010) 1–41.
[26] I. Mani, M.T. Maybury, Advances in Automatic Text Summarization, MIT Press, 1999.
[27] V. Nastase, Topic-driven multi-document summarization with encyclopedic knowledge and spreading activation, in: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, 2008, pp. 763–772.
[28] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, M.-C. Hsu, Prefixspan: Mining sequential patterns efficiently by prefix projected pattern growth, in: Proceedings of the Twenty-ninth IEEE International Conference on Data Engineering (ICDE), 2001, p. 0215.
[29] H. Saggion, T. Poibeau, Automatic text summarization: Past, present and future, Multi-source, Multilingual Information Extraction and Summarization, Springer Berlin Heidelberg, 2013, pp. 3–21.
[30] M. Saravanan, B. Ravindran, Automatic identification of rhetorical roles using conditional random fields for legal document summarization, J. Artif. Intell. Law (2008) 481–488.
[31] K. Sarkar, Syntactic trimming of extracted sentences for improving extractive multi-document summarization, J. Comput. 2 (7) (2010) 177–184.
[32] D. Shen, J. Sun, Q. Yang, Z. Chen, Document summarization using conditional random fields, in: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence, Hyderabad, India, 2007, pp. 2862–2867.
[33] Y. Song, W. Ng, K.W.T. Leung, Q Fang, SFP-Rank: Significant frequent pattern analysis for effective ranking, Knowl. Inf. Syst. 43 (3) (2015) 529–553.
[34] F.M. Suchanek, G. Kasneci, G. Weikum, Yago: A core of semantic knowledge, in: Proceedings of the Sixteenth International Conference on World Wide Web, 2007, pp. 697–706.
[35] TexLexAn. Texlexan: An open-source text summarizer. http://texlexan.sourceforge.net/ (retrieved March 2014).
[36] D. Wang, T. Li, C. Ding, Weighted feature subset non-negative matrix factorization and its applications to document understanding, in: Proceedings of the IEEE Tenth International Conference on Data Mining, 2010, pp. 541–550.
[37] D. Wang, S. Zhu, T. Li, Y. Chi, Y. Gong, Integrating document clustering and multi-document summarization, ACM Trans. Knowl. Discov. Data 5 (3) (2011) 14–26.
[38] J. Wang, J. Han, BIDE, Efficient mining of frequent closed sequences, in: Proceedings of the Twentieth International Conference on Data Engineering, 2004, pp. 79–90.
[39] Wikipedia. Wikipedia website. http://www.wikipedia.org (last accessed 01.03.2013), 2013.
[40] C. Wu, C. Liu, N.C. Debnath, Ontology-based text summarization for business news articles, in: Proceedings of the ISCA Eighteenth International Conference on Computers and Their Applications, Ed., 2003, pp. 389–392.
[41] S. Wu, Y. Li, Y. Xu., Deploying approaches for pattern refinement in text mining, in: Proceedings of the IEEE Sixth International Conference on Data Mining, 2006, pp. 1157–1161.
[42] F. Xie, X. Wu, X. Zhu, Document-specific key phrase extraction using sequential patterns with wildcards, in: Proceedings of the IEEE Tenth International Conference on Data Mining (ICDM), 2014, pp. 1055–1060.
[43] G. Xu, Z. Wu, G. Li, E. Chen, Improving contextual advertising matching by using Wikipedia thesaurus knowledge, Knowl. Inf. Syst. 43 (3) (2015) 599–631.
[44] X. Yan, J. Han, R. Afhar, CloSpan: Mining closed sequential patterns in large datasets, in: Proceedings of the SIAM International Conference on Data Mining, 2003, pp. 166–177.
[45] Z. Yang, K. Cai, J. Tang, et al., Social context summarization, in: Proceedings of the Thirty-fourth International ACM SIGIR Conference on Research and Development in Information Retrieval, 2011, pp. 255–264.
[46] D.M. Zajic, B.J. Dorr, J. Lin, in: Single-document and multi-document summarization techniques for email threads using sentence compression, 44, 2008, pp. 1600–1610.
[47] M. Zaki, Spade: An efficient algorithm for mining frequent sequences, Mach. Learn. 42 (1-2) (2001) 31–60.
[48] W. Zhang, T. Yoshida, X. Tang, Q. Wen, Text clustering using frequent itemsets, Knowl. Based Syst. 23 (5) (2010) 379–388.
[49] N. Zhong, Y. Li, S. Wu., Effective pattern discovery for text mining, IEEE Trans. Knowl. Data Eng. 24 (1) (2012) 30–44.
[50] J. Zhu, C. Wang, X. He, et al., Tag-oriented document summarization, in: Proceedings of the Eighteenth International Conference on World Wide Web, 2009, pp. 1195–1196.