Available online at www.sciencedirect.com
ScienceDirect
Information Processing and Management 43 (2007) 1643–1662
INFORMATION PROCESSING & MANAGEMENT
www.elsevier.com/locate/infoproman

Document concept lattice for text understanding and summarization

Shiren Ye, Tat-Seng Chua *, Min-Yen Kan, Long Qiu

Department of Computer Science, School of Computing, National University of Singapore, Singapore 117543, Singapore

Received 18 July 2006; received in revised form 16 February 2007; accepted 8 March 2007
Available online 22 May 2007

Abstract

We argue that the quality of a summary can be evaluated based on how many concepts in the original document(s) that can be preserved after summarization. Here, a concept refers to an abstract or concrete entity or its action often expressed by diverse terms in text. Summary generation can thus be considered as an optimization problem of selecting a set of sentences with minimal answer loss. In this paper, we propose a document concept lattice that indexes the hierarchy of local topics tied to a set of frequent concepts and the corresponding sentences containing these topics. The local topics will specify the promising sub-spaces related to the selected concepts and sentences. Based on this lattice, the summary is an optimized selection of a set of distinct and salient local topics that lead to maximal coverage of concepts with the given number of sentences. Our summarizer based on the concept lattice has demonstrated competitive performance in Document Understanding Conference 2005 and 2006 evaluations as well as follow-on tests.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Text summarization; Document concept lattice; Concept; Semantic

1. Introduction

Text summarization is the process of distilling the most important information from sources to produce an abridged version for a particular users and tasks (Mani & Maybury, 1999). It has been applied to news articles, group meeting transcripts, and web pages.

In this paper, we review and detail our approach to automatic, multi-document extractive summarization. Such summarization methods simplify the problem of summarization into the problem of selecting a representative subset of the sentences in the original documents. This is in contrast to abstractive summarization, which may compose novel sentences, unseen in the original sources. However, abstractive approaches require deep natural language processing such as semantic representation, inference and natural language generation, which have yet to reach a mature stage today.

* Corresponding author.
E-mail addresses: yesr@comp.nus.edu.sg (S. Ye), chuats@comp.nus.edu.sg (T.-S. Chua), kanym@comp.nus.edu.sg (M.-Y. Kan), qiul@comp.nus.edu.sg (L. Qiu).

0306-4573/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2007.03.010

In both extractive and abstractive summarization, a key consideration is how to properly represent the knowledge contained in the input document. This task of document understanding (Dang, 2005) is a core issue in text summarization. In our approach, we model such information using a simple, shallow lexical representation, which we term *concepts*. From our perspective, concepts model key facts and answers to important questions that the reader asks. Here, a concept are words that are relevant to abstract or concrete entities and their actions. Our system also attempts to link together different lexical realizations of the same latent concepts (i.e., synonymy) using a process of semantic equivalence discovery.

Once words that represent unified concepts in the documents are linked, we represent the sources as a *document concept lattice* (DCL). This data structure organizes the set of all concepts present in the input into a direct acyclic graph, where nodes represent overlapping sets of concepts. Nodes in the DCL store the frequency of its set of concepts within the input, and contain pointers to subsumed nodes. At the base level of the lattice, sentence nodes store the inventory of concepts in each sentence, along with its word count.

Following our work in the Document Understanding Conference (DUC) 2005 and 2006 (Ye, Qiu, Chua, & Kan, 2005; Ye, Qiu, Chua, & Kan, 2006), our summarization algorithm uses the DCL to select a globally optimum sentence set that represents as many concepts as possible with the fewest words. This is done by computing a fitness metric for a candidate summary, which we term a summary's *representative power*. Due to the compact representation of many candidate summaries in the DCL, it is computationally efficient to evaluate the representative power of many summary candidates. Our algorithm selects the set that results in highest representative power for the output summary. Experimental results from DUC evaluations and follow-on tests indicate that our DCL-based approach is competitive with the state-of-the-art. It should be noted that our work obtains its high performance without the use of common heuristic factors such as position and BUG.

The contributions of our work are:

- We motivate our approach by considering an automatic summarization evaluation framework where summaries are judged on answer loss.
- Motivated by our evaluation metric on answer loss, we propose a novel document model, the document concept lattice, which indexes sentences with respect to their coverage of overlapping concepts. The DCL represents partial overlapping subsets of concepts at different levels of granularities and succinctly encodes an exponential number of candidate summaries.
- The DCL is then used as a base data structure to implement a theoretically-sound summarization algorithm that selects an optimal combination of representative sentences from the DCL encoding.
- The tests on DUC 2005 and 2006 indicate that both concept and DCL have significant contributions to the performance of text summarization.

The rest of this paper is organized according to the above contributions. Following a description of our evaluation framework, we discuss the notion of a concept – our representation for the source text's information – and our algorithm for concept detection. We then address the principles behind the DCL data structure and its construction in Section 4. Section 5 depicts the algorithm for constructing the summary based on DCL. We report the experiment results based on DUC 2005 and 2006 corpus and discuss our extended experiments in Section 6.

2. What is a good summary?

In this section, we propose a theoretical summary evaluation that focuses on summary content. We introduce this evaluation criterion as a way of motivating our summarization approach.

How does one judge the content of a text? Evaluating the quality of a summary is a fundamental problem in summarization. Recently, several well-known measures have been developed and used. These include responsiveness measures such as ROUGE (Lin & Och, 2004), Pyramid (Passonneau, Nenkova, McKeown, & Sigelman, 2005), Basic Elements (Hovy, Lin, & Zhou, 2005), correlation (Dang, 2005); readability measures such as grammaticality, non-redundancy, referential clarity, focus, as well as structure and coherence (Dang, 2005).

These measures are based on the comparisons between system-crafted and human-crafted summaries or evaluated by human accessors directly.

Along a similar line as other responsiveness measures, we propose that summaries might be evaluated with respect to their content: the ratio of content preserved from the source document. Once a concrete definition of content is defined, this measure becomes an objective criterion, sidestepping the problems of subjectivity such as language-related issues. In our approach, we equate content with answers to factual questions about the source documents. Akin to factoid question answering in TREC (Voorhees, 2001), we say that answers to basic *who, what, where, when, why* and *how* questions make up the content that should be preserved in summaries. Then, if a list of such answers are given, one can calculate the *Answer Loss* (AL) of a compression C of the uncompressed source texts S as

$$\mathrm{AL}(C,S) = (|A(C)|/|A(S)|) \tag{1}$$

where $|A(C)|$ and $|A(S)|$ are the numbers of answers in C and S, respectively[1]. Such loss is an objective measure of a summary's task-oriented fitness. It is equivalent to measuring the factual recall of a summary based on a list of answers to questions derived from the source document set. Query-based summarization can be handled in a similar way, in which the source texts are first filtered for relevance to the query, and only answers that appear in the filtered set would be considered in the denominator of Eq. (1). This approach assumes that an answer existing in a summary will always be extractable by the reader, which we think is reasonable.

However, all answers do not contribute equally in judging the quality of a summary. Answers appear frequently in the document set might be considered to be more crucial in understanding the source texts. This is because texts often exhibit such repetition on purpose, in order to aid a reader's understanding of the text. We thus need to weight the importance of each sentence based on which answers in contains and the importance of each contained answer.

However, in a summary we have limited space and repetition comes at the premium of space that could be used to introduce other answers. Hence, we feel that the set of sentences in a good summary should have two properties: (i) each sentence should collect as many answers as possible; and (ii) the number of overlapping answers among different sentences should be small. These properties are described *as redundancy and diversity* in Katz's G model (Katz, 1996) and are used as basis in many current summarization approaches.

3. Concepts as answers

We have yet not discussed how to create a list of answers from source documents. Such question and answer (QA) pairs can be manually defined, as is done in the manual evaluation schemes, such as the Pyramid approach (Passonneau et al., 2005). Manual compilation has the advantage of getting a clean set of QA pairs but introduces subjectivity and scalability issues. Thus, we propose to create the answer list automatically, making the process scalable while it may introduce some errors.

If we set out to build a summarization system that minimizes answer loss, we need a definition of what an answer is and then an algorithm to minimize the loss. We tackle the first problem in this section, by equating answers with a certain set of open-class words, which we term concepts.

From our own informal study of question and answers in both summarization and QA systems, we observe that answers usually describe abstract or concrete entities and their actions. These entities and actions may be expressed using a diverse set of lexical items and/or phrases. These set of open-class of words we denote as (*semantic*) *concepts*, after the convention started in TRECVID (Naphade & Smith, 2004).

Definition. A *(semantic)* concept is a set of synonymous or equivalent words or phrases that describes an abstract or concrete entity or its actions. Usually, these words and phrases are limited to the set of open-class words of nouns, verbs, adjectives and adverbs, as well as noun or verb phrases.

The words and phrases that make up concepts are easily obtainable from running a part-of-speech tagger on the source text and chunking sequences of such open-class words together. From this viewpoint, WordNet

[1] The duplicate answers, such as repeated ones in diverse sentences, will be counted out as well.

(Miller, Beckwith, Fellbaum, Gross, & Miller, 1990) can be considered as a collection of concepts. By focusing on concepts, we are reducing the problem of finding answers to specific factual questions (a hard problem) and approximating it with finding concepts (an easier problem).

Closely related to our definition of a concept is the notion of a basic element (BE) proposed by Hovy et al. (2005). A BE is a minimal semantic unit which is broken down from reference sentences, which is usually represented by the form of tri-tuples ⟨`head`, `modifier`, `relation`⟩ to reflect the relations between heads and modifiers. Furthermore, Hovy et al. considered summary content units (SCUs) in the Pyramid evaluation method (Passonneau et al., 2005) and various n-grams in ROUGE (Lin & Och, 2004) as different versions of BEs. They also argued that the smaller BEs tend to facilitate the automation in the procedure of unit identification, and matching of diverse equivalent expressions.

Our semantic concepts can be thought of as extending the argument for using smaller units to represent the source documents. We simplify BEs by ignoring the relationship encoded grouping words into their tuples. This simplification allows for ease of processing, but comes with a price: we make a strong assumption that relationships between words are stable across input documents. For instance, our representation of concepts does not distinguish between "John killed Smith" and "Smith killed John" as they both generate the same unordered set of concepts, namely {Smith, John, killed}. We feel that this is a worthwhile trade-off as (i) relations are largely captured by extracting verbs and nominalized actions, and that (ii) systems that do encode and filter their representation of actions have yet to systematically demonstrate a quantitative advantage. In practice in multi-document summarization, we have found that sentences that have the same inventory of concepts rarely express different semantics between concepts.

The set of concepts in different source sentences often exhibit partial overlap rather than being subsumed or being equivalent (Radev, Jing, & Budzikowska, 2004). A key aspect of using smaller units for representing the source text is that it enables us to determine the degree of the intersection between sentences. Compared with more complex forms such as SCUs with more terms and even mini-structures, the techniques for detecting units consisting of fewer terms is mature, requiring minimal human effort. This is pivotal for large-scale summarization testing and evaluation.

In order to visualize semantic concepts more clearly, we present an example take from the DUC 2005, selected the Cluster *d*324*e*. Fig. 1 shows an article in which concepts are underlined. This article discusses post Falkland War Argentine – British relations. Here, sentences (1)–(4) mention the Falkland Islands War between Britain and Argentina. Although their syntactic structures vary, they contain a stable and overlapping set of concepts that represent key entities and their actions in the story. This is consistent with our observations that even though the authors may perceive an idea from different perspectives, and narrate it using various terms (such as synonyms, aliases or words with different part-of-speech) and using diverse sentential and phrasal constructs, their concepts still belong to a finite set and partially overlap in most cases.

No	*Sentence*
1	Argentine-British relations since the Falkland Islands War in 1982 have gradually improved.
2	Thirteen years after the war between Britain and Argentina over the Falkland Islands, ...
3	Argentina was still obsessed with the Falkland Islands even in 1994, 12 years after its defeat in the 74-day war with Britain.
4	Argentina and British fought over Falkland islands in 1982.
5	Commercial relations have continued to improve between UK and Argentina.
6	Mr Douglas Hurd, Britain foreign secretary, is to visit Argentina early next year.
7	Britain lifted military protection zones around the Falklands in 1990, 8 years after the Argentina-British war over the area.
8	Mr Hurd's visit to Argentina is the first by a cabinet minister since the Falklands conflict indicating improved diplomatic relations between UK and Argentina.

Fig. 1. Some sentences from cluster at DUC 2005 (Cluster *d*324*e*).

Based on the discussion above, concepts are considered as the terms that remain after removing closed-class words in the source sentences. We check WordNet for entries that match sequences of words as multi-word concepts, such as *Falkland Islands* in the above example. Words that do not appear in WordNet are counted as individual concepts.

3.1. Similarity computation for concept unification

Synonymy and lexical variation are prevalent in natural texts. We need a mechanism to unify these variations into a canonical, controlled form such that accurate frequency counts for concepts can be established for the source text.

In order to handle variations in terms used to describe the same concept, the first step is to identify the equivalence relations among terms about identical concepts. Following the idea of lexical chains (Morris & Hirst, 1991; Silber & McCoy, 2000) and concept links (Ye et al., 2005), we propose a method for fast semantic equivalence discovery. Our method handles synonymy and hyponymy usingWordNet, and morphological variation using stemming.

Specifically, we calculate semantic similarity between candidate concepts by considering the following relations in addition to standard stemming: (i) synonyms or hyponyms (such as *war–battle*); (ii) derivational morphological variations (such as *decision–decide*, *Argentine–Argentina*); and (iii) inflectional morphological variations (such as *relations–relation*).

We now pause to discuss the details on our usage of WordNet for computing similarity. According to Lesk's hypothesis (Lesk, 1986), word senses that are related can be characterized by their shared words in their definition. In our work, the definition of a sense of a concept consists of information from WordNet: its *synset* (a set of synonyms of the sense), *gloss* (the explanation of the sense with possibly specific examples), direct *hypernyms* (is-a relation) and *meronyms* (has-a relation). For two concepts c_i and c_j with sense definitions $SS_1^i, SS_2^i, \ldots, SS_m^i$ and $SS_1^j, SS_2^j, \ldots, SS_n^j$, respectively, their semantic similarity is

$$\mathrm{Sim}(c_i, c_j) = \max_{\substack{1 \leqslant x \leqslant m \\ 1 \leqslant y \leqslant n}} \frac{\sum_{z=0}^{k} |\mathrm{Overlap}^z(SS_x^i, SS_y^j)|^2}{|SS_x^i| \times |SS_y^j|}. \tag{2}$$

Here $\mathrm{Overlap}^z(SS_x^i, SS_y^j)$ is defined as the zth shared segment in SS_x^i and SS_y^j. In Eq. (2), shared segments are not allowed to be properly subsumed in any other shared segments.

For sample sentences shown in Fig. 1, we can group 15 distinctive terms into 6 sets of concepts (A–F in Table 1) by means of semantic equivalence discovery. For example, British, Britain and UK will be mapped together since their senses are close according to Eq. (2). In total, we can obtain 16 sets of equivalent or repeated concepts, as shown in Table 1. Compared with the fact that 61 distinctive words appear in the sample sentences, these 16 concepts are more effective since we can represent them in a more compact feature space, where diverse terms with identical meaning can be unified.

Table 1
Equivalent concepts (A–F) and repeated concepts (G–P) appearing in sentences in Fig. 1

ID	Concept	ID	Concept
A	Argentine Argentina	I	Island
B	British Britain UK	J	1982
C	War fought conflict military	K	Year
D	Diplomatic foreign	L	Mr
E	Secretary minister	M	Hurd
F	Area zone	N	Visit
G	Improve	O	Relation
H	Falkland islands	P	Falklands

Here NP *Falkland islands* is considered as a concept as we can find it in WordNet; Hurd and Falklands are also considered as two concepts since they are unknown words (missing in WordNet). In total, 16 concepts instead of 61 distinct words found in these sentences can be used to represent the contents.

Here, the computational cost is $O(n^2)$ since computing $\mathrm{Sim}(c_i, c_j)$ requires a scan through all possible pairs of senses for all pairs of concepts. This is unreasonable to compute online, as there are over 150 k entries in WordNet. To reduce the run-time computational cost, we pre-compute the semantic similarity between all pairs of WordNet entries (concepts in our discussion) offline and store non-zero valued pairs in a hash table. This reduces the computation to a O(1) lookup. Also, as most pairs of concepts have no similarity, the size of this hash table is acceptable (238,728 records in total) and can be stored in main memory.

This similarity measure (Eq. (2)) focuses on discovering semantic equivalence relations among the diverse terms which are used to present identical and close concepts. This is different from dependency and proximity similarity measure proposed by Lin (1998). Lin's measure emphasizes linear proximity and dependency relationships, which is less restrictive than equivalence relation and does not serve our purpose well. For example, the pair of *water* and *fish* will emerge in Lin's measure, but they are not equivalent concepts under our framework.

Closely related is research on using lexical chains for summarization, such as Barzilay and Elhadad (1997), Silber and McCoy (2000), Schiffman, Nenkova, and McKeown (2002), Doran, Stokes, Carthy, and Dunnion (2004), Bosma (2005). Our approach examines the similarity of all pairs of open-class terms based on synset, gloss, direct hypernym and meronym information in WordNet.

At the end of this section, let us summarize the difference between our concepts in DCL and bag-of-word in vector space model (VSM): (1) Intuitively, compared with stop-word removing used in the latter, closed-class terms filtering used in the former is more capable of grasping the key information in text. In our system, the number of closed-class terms that can be detected reaches 500 s, which is much larger than that of stop-words commonly used. (2) Semantic equivalence discovery in the former can deal with synonyms and conjugations, which is powerful than stemming in the latter. We notice that the number of concepts is about 10–20% less than that of words in VSM. (3) In view of underlying philosophy, concepts are designed to grasp the existing entities and actions in the document cluster rather than naive assumption in VSM.

4. Document concept lattice

Given the notion of semantic concepts, the extractive summarization approach is equivalent to picking a set of sentences that represent as many salient concepts as possible. Sentences that contain more important concepts should be picked, but sentences that are picked earlier may reduce the utility of other sentences that partially overlap in terms of concepts, if they are picked later. According to Eq. (1), finding the optimal summary is thus equivalent to a knapsack problem, which is NP-hard: pick a given number of sentences so that the answer loss in summary is minimal. However, it is exponentially difficult to examine all possible combinations of sentences in the search process.

In order to overcome the problem of computational complexity, we introduce a data structure that helps to summarize the overlap between concepts shared among sentences. This enables us to succinctly represent overlapping concepts and compute an optimal combination in a reasonable online running time.

Most summarization systems to date do not select the set of summary sentences as a single step. Rather, most systems use some variation of maximal marginal relevance (MMR) Carbonell and Goldstein (1998) and cross-sentence informational subsumption by Radev et al. (2004), in which the summary is built up incrementally, sentence by sentence. These MMR-based systems typically use a method to choose the kth sentence using some measure of the sentence's *redundancy* and *diversity* with the set of sentences already flagged as part of the summary. The sentence with the highest score is included and the process is iterated greedily until the summary length is satisfied.

A problem is that this greedy process may lead the summarization system to pick a local optimum instead of a global one. To illustrate this phenomena, we construct an example document with seven sentences containing a set of artificial answers (without loss of generality, these answers could be words, BEs, or semantic concepts, depending on the system), shown in Fig. 2. Most MMR based systems (Barzilay & Elhadad, 1997; Carbonell & Goldstein, 1998; Radev et al., 2004) would differentiate these sentences into three groups $\{1,2,3,4\}$ (each containing unique terms with respect to each other), $\{5,6\}$ (sharing two groups of two terms from $\{1,4\}$), and $\{7\}$ (sharing one term from each sentence $\{1,4\}$). For this example document, we can also compute the answer loss ratio of any summary easily. Each of the seven sentence has four terms, amounting to

Sentence ID	answers	answer loss
1	A B C D	21/28
2	E F G H	21/28
3	I J K L	21/28
4	M N O P	21/28
5	C D I J	18/28
6	G H M N	18/28
7	D G J M	16/28
5 & 7	C D G I J M	12/28
1 & 7	A B C D G J M	12/28
5 & 6	C D G H I J M N	8/28
1 & 5 & 6 & 7	ABCDGHIJKLMN	4/28
1 & 2 & 3 & 4	ABCDEFGHIJKLMNOP	0/28

Fig. 2. An artificial sample consisting of seven sentences. Here each sentence contains four concepts, and there exist 16 distinctive concepts in total.

28 answers. The answer loss is then the fraction of answers not represented in the summary, weighted by frequency. For example, oicking any sentence in group 1 would result in a summary that represents 2 answers that occur once, 1 answer that appears twice, and 1 answer that appears thrice or $7 = (2 \times 1) + 2 + 3$, giving an answer loss of 21/28.

When the answer length is set to a single sentence, MMR-based systems would select sentence 7 as the summary. This preserves as many answers as possible given the constraint of a single sentence, which is fine. However, when the length is set to 2, they simply add the next optimal sentence to the already-selected sentence 7. This results in an answer loss of 12/28, however better selection can be made by selecting the two sentences from the second group, $\{5, 6\}$. This results in a more competitive answer loss of 8/28 as compared to 12/28. This problem is exacerbated in the 4 sentence summary scenario. MMR systems would select a summary equivalent to the $\{1, 5, 6, 7\}$ with an answer loss of 4/28, while a selection of $\{1, 2, 3, 4\}$ would cover all answers. To properly compute the optimal summary requires a global selection strategy rather than a local, iterative one.

In contrast to other methods, our summarization method uses a global selection strategy that seeks to minimize answer loss. We believe this is a key differentiating factor and contributes to our relative improvement in performance over peer systems. In DUC 2005 and 2006 tasks, 10–15 sentences are necessary to satisfy the 250 word length requirement. At these longer summary lengths, selecting less important and non-overlapping sentences may exhibit some key details of the topic. In contrast, greedy selection of highest ranked sentences may cause a system to pick sentences that are too general (such as titles or section headers) that will contribute little in the final summary.

Our analysis of summarization as an analogy to answer preservation bears a resemblance to rhetorical structure theory (RST) (Mann & Thompson, 1988). We borrow the concept of nuclearity (Marcu, 1997, 1998) to depict the general/special degree of sentences tied to their spans. In RST, a rhetorical relation typically holds between two contiguous text spans, of which one span (the nucleus) is more central to the writer's intention than the other (the satellite). The most nuclear sentence of an RST analysis is one which is most central to the author's purpose (Bosma, 2005).

If we equate nuclearity to concept frequency, then we can build an RST-like tree structure, as shown in the hanging structure in Fig. 3. The high-frequent concepts appear in more sentences, and thus associate these sentences together. Furthermore, the concept sets with diverse frequencies will overlap each other and form a hierarchy.

While an RST approach to summarization is well motivated, it is difficult to build such trees for single documents without explicit textual cues, and even more problematic to build a tree for a set of multiple source documents. As such, introduce a new data structure, the document concept lattice, that compactly represents such hanging structures. The DCL describes the hierarchy of partial overlapping concepts present in the

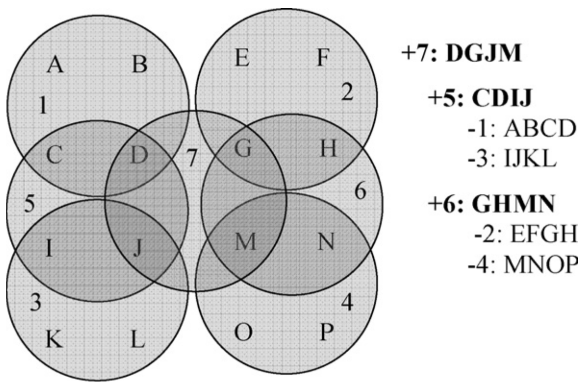


Fig. 3. Venn diagram and hanging structure for artificial sample in Fig. 1.

source sentences. A key difference from RST trees is that the DCL is directed acyclic graph, in which children nodes can be descended from multiple parents.

A DCL for the sample artificial document is shown in Fig. 4. At the leaves of the DCL, source sentence nodes (in gray) show the set of concepts present in each sentence and its salience (computed as the sum of the frequencies of each answer). The internal nodes of the DCL encode maximal partial overlaps between descendent nodes. The bottom node represents the involved concept space. These nodes also show their concept inventory and the number of source sentences that contain the set of concepts.

As the DCL encodes superset and subset relations, a few characteristics should be clear from observation: (1) that the semantic concepts closer to the root of the DCL are the more frequently occurring concepts; (2) that all of the concepts of a node are covered by any and all of its descendant nodes; (3) that a node covers all of the sentences that are covered by its descendants.

In a DCL, shared concepts are hierarchically organized into internal nodes. The DCL structure is similar to the data cube representation for structured data found in database research, used to summarize measures according to specific dimensions, such as the total sale in the specified regions, seasons and categories (Stumme, Wille, & Wille, 1998). From these observations, we can see that it is possible to represent frequent (nuclear) concepts in the upper parts of the DCL by using its descendant nodes, possibly in a way that results in non-overlapping choice of concepts.

For example, we can represent nodes 2.1–2.4, using the nodes at the lower level 1.1–1.4, which covers all of the same concepts. This is equivalent to selecting the nodes in the middle of the DCL only. In this case,

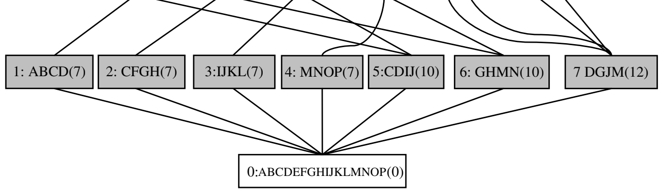


Fig. 4. The lattice structure for indexing and categorizing existing concepts.

sentences 5 and 6 capable of representing nodes 1.1–1.4, despite their lower frequency of representation compared to sentence 7. This two sentence summary has the globally optimum answer loss. This model addresses the weakness of the greedy model discussed earlier. The DCL representation easily allows us to select sentences 5 and 6 for a two sentence summary.

4.1. Constructing the document concept lattice

In contrast to other methods (Lin & Hovy, 2000a, 2000b; Harabagiu & Lacatusu, 2005), our approach employs a simple representation of topics as open-class words, and phrases. Our innovation is in pairing this simple representation with the DCL, which hierarchically organizes these semantic concepts into a partial ordering lattice structure. We now describe how to build the lattice from source documents.

For ease of reference, we denote the set of concepts in the source documents using letters as shown in Table 1. The construction of a DCL shown in Fig. 5 can be carried out as follows. (1) Repeated concepts are identified from the parsing each sentence to construct the leaf nodes (e.g., nodes 1–8 for the eight sentences given in Fig. 1). (2) The maximal common concepts are identified and create first level of internal nodes (nodes $1.x$). For example, the maximal common concepts for nodes 1 and 4 are $ABCHJ$ in node 1.1 rather than AB in node 4.1 or ABC in node 3.1. If there are more than one possible higher level internal nodes that can be built, a node ϕ with blank concept are used to cover all upper nodes. (3) Other internal nodes are recursively generated from the existing leaf nodes and internal nodes until no new internal node can be generated. (4) Finally, a new node containing all concepts is introduced at the base.

A formal description and construction algorithm of a lattice can be found in Wang, Dubitzky, Düntsch, and Bell (1999) and Nguifo, Duquenne, and Liquiere (2003). However, the computation cost of constructing a complete DCL is very exponentially expensive since almost every possible concept combination needs to be examined. Fortunately, we are only interested in nodes with high-frequency concepts. Nodes with infrequent concepts (satellite concepts) tend to be about trivial information that can be ignored. For instance, we can simplify nodes 1.1, 1.2 and 2.1 as a single node, where the statistical information relating to the low-frequency concepts K and J is ignored. Here, we can use association rules (Agrawal & Srikant, 1994) to locate and mine sets of frequent concepts, as shown in Fig. 6.

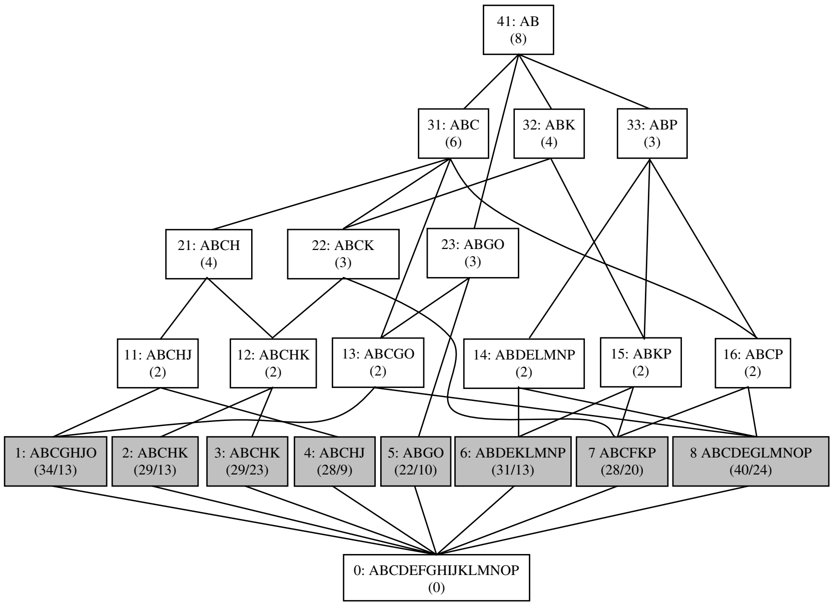


Fig. 5. Document concept lattice derived from concepts identified in sentences in Fig. 1. The nodes with a gray background are leaf nodes that represent source sentences. The links between node describe their partial ordering. The number in parentheses in both node types differs: in internal nodes, it denotes the frequency of its represented concepts; while in leaf nodes, the two numbers are the frequency of its concepts and the number of words in the sentence.

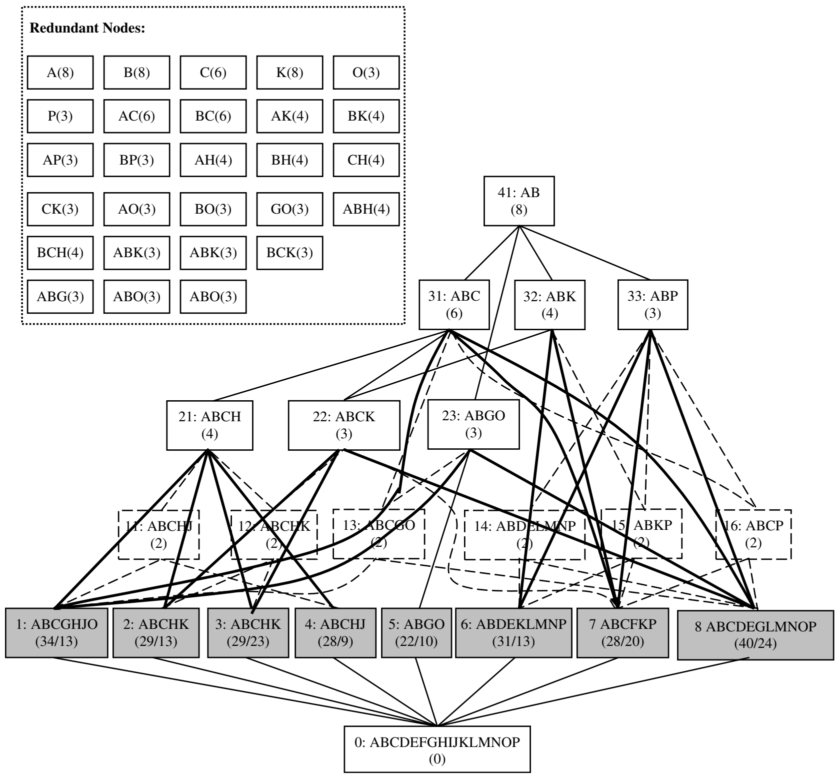


Fig. 6. The construction of DCL for sentences in Fig. 1 using associations rules Here, the threshold of minimal frequency of concept set is set to 3. The derived nodes with concept frequency less than 3 and the corresponding links are removed (marked by dashed).

Meanwhile, the concept sets with fewer concepts whose frequency is equal to a larger concept set will be ignored during the procedure of DCL construction. For instance, concept sets for *A* and *B* will be excluded, since they appear 8 times as the larger concept set *AB* does. Hence, just the largest frequent concept sets are taken into account. Finally, it is convenient to build a concise version of DCL based on the frequent concept sets according their relations of set inclusion.

As a document concept model, DCL has the following properties:

(1) In a derived node, the local topic, represented by the a set of existing concepts, defines a sub-space in the concept space of the document cluster. The covered sentences appear to argue the involved local topic, and even have the identical meaning. For instance, all of the sentences in node 21 describe a concrete event of *the war between Britain and Argentine* with respect to the local topic of *ABCH*.
(2) If a set of high-frequency concepts are found adjacent to each other, they will most likely be noun phrases or verb phrases tied to particular entities or actions. Actually, they tend to become the true concepts which we defined in Section 3. In cases where the concepts appear in isolation, they might be the alias or informal expressions (e.g., *Falklands*) of the canonical concept names (e.g., *Falkland Islands*).
(3) In a pair of derived nodes with partial ordering relation, the upper node, spanning a larger local topic, will cover more sentences than the lower node does. The representatives of the lower node will represent the upper node as well, since it must contain the concepts existing in the larger local topic. Consequently, if we select the representatives of a set of non-overlapping nodes[2] in the middle of DCL as the observation, all upper nodes covering these non-overlapping nodes will be represented as well.

[2] Derived nodes X and Y are non-overlapping iff neither $X \preceq Y$ nor $Y \preceq X$.

(4) When we zoom down through DCL, we can find more details, where more smaller local topics (derived nodes) exists. The derived nodes in higher layers contain fewer concepts, but the frequency of these concepts are higher. Hence, if we select a set of derived nodes in the higher layer as the observation, fewer representative sentences will merge and vice versa. It suggests that we can utilize representative sentences with different nuclearities from local topics in diverse layers in DCL to build a series of summaries with various compression ratios according to the pre-defined length.

5. Summarization using the DCL

Based on this lattice structure, we can extract a summary customized to a desired length. For example, if we need to generate a longer summary, (assuming that we only draw one sentence per node to satisfy diversity criteria), we can use a set of non-overlapping frequent nodes with a finer granularity for summary extraction. We first describe our metric for assessing sentence relevance and then detail the algorithm to compute the optimal summary using the DCL.

In DCL, the chosen representative sentence should not only narrate the predicates and arguments related to the internal node, but is also expected to eliminate uncertainty among the unselected sentences. Consequently, we try to choose representative sentences as ones whose concepts frequently occur in other sentences as well. Considering the various contributions of different concepts, we further employ inverse document frequency (IDF[3]) from a large text collection to better weight the frequency counts to obtain the representative power (RP) of sentence S as follows:

$$\mathrm{RP}(S) = \sum_{C_i \in C(S)} \{|C_i(\mathbb{D})| \cdot \log(N/df_{C_i})\}, \tag{3}$$

where N is the number of documents in the corpus, $|C_i(\mathbb{D})|$ is the frequency of concept C_i in the document cluster $\mathbb{D}$, and df_{C_i} is the number of documents containing concept C_i. The same formula is used to compute the significance of internal nodes as well.

In summarization tasks such as DUG, summary length is judged based on word count rather than sentence count. As such, we need to adjust RP for sentence length. We thus introduce the ratio of RP and word count that we called the *representative power adjusted for sentence length*:

$$\mathrm{RP}_{sl}(S) = \mathrm{Sig}(S)/\log(\mathrm{word\_count}(S)). \tag{4}$$

In the example given in Fig. 5, RP_{sl} of node 4 exceeds other leaf nodes, for its RP_{sl} is 35/9 if we ignore IDF. As Eq. (4) will prefer shorter sentences rather than longer ones. However, previous work (Nobata & Sekine, 2004; Erkan & Radev, 2004b) indicate that longer sentences serve better in summaries, thus we include a log factor to lessen the impact of word_count.

For query-based summarization, we need to consider whether the selected sentences are relevant to the user's intention expressed by their query. To achieve a query-biased RP measure, we modify RP_{sl} to account for concept overlap (Ye et al., 2005) between the query and each node, namely:

$$\mathrm{RP}(s) = \mathrm{Sig}(s)/\log(\mathrm{word\_count}(s)) + \lambda \cdot \mathrm{sim}(s, q), \tag{5}$$

where λ is the weight of users' bias (empirically set to 0.2–0.5 in our experiment), and sim(s, query) is the sum of semantic similarities between the concepts in node s and query q defined in Eq. (2) (the concepts existing in the query are modified by TFIDF as well).

Constructing a DCL for a set of source document enumerates and indexes all concepts into a DAG of internal nodes and pointers to sentences in the leaf nodes. We utilize dynamic programming to explore the search space specified by DCL in three steps: (i) selecting a set of salient internal nodes, say $\Omega = \{N_1, \ldots, N_{|\Omega|}\}$; (ii) picking representative sentences (i.e., covered base nodes with highest RPs) from each internal node in Ω; and (iii) examining the combinations of the selected sentences to output the best combination leading to minimal

[3] We use a large-score balanced corpus available at http://elib.cs.berkeley.edu/docfreq to get the term document frequency, to retrieve the concept distribution.

answer loss. Here, the collection of summary candidates COL comes from the product of representatives in distinct derived nodes, namely,

$$\mathrm{COL} = \{B_1 \times \ldots B_{|\Omega|}\} \tag{6}$$

where B_i belongs to the representatives of derived node $N_i(1 \leqslant i \leqslant |\Omega\|)$. The number of representatives in each internal node is called the *search beam width* (SBW). The leaf nodes in each combination should not share any internal node in Ω to satisfy diversity. The summary candidate whose total $\mathrm{RP}_{\mathrm{sl}}$ exceeds all others in COL is the output summary, since it will lead to minimal answer loss under our evaluation framework.

For the first step (selecting Ω), we can simplify it as the search for a sufficient non-overlapping set of derived nodes with maximal RPs as the promising sub-spaces for exploitation. As it is unnecessary to explore the overlapping derived nodes according to the properties of DCL, we navigate from the root of DCL and collect a set of derived nodes with largest RPs by means of: (i) selecting the children of root; or (ii) recursively splitting the children with maximal RPs into descendent derived nodes. A series of observations with increasing derived nodes will be generated during this sub-space exploration. For instance, in the samples shown in Fig. 5, the series of the observations should be $\{41\}$, $\{31,32,23,33\}$, $\{21,22,23,14,15,16\}$[4], etc. If the required size of Ω varies from 1 to 5, Ω will be constructed from the observation series and comprise the sets such as $\{41\}$, $\{31,32\}$, $\{31,32,23\}$, $\{31,32,23,33\}$ and $\{21,22,23,14,15\}$. When the size of Ω increases, the derived nodes in the observation will be further away from the root of DCL, which implies that more non-overlapping but smaller local topics with high RPs will be examined.

For the second step (selecting representatives), we can suppose that only the base nodes with top n (i.e., BWS) RPs are promising and can play representative roles of its covering derived nodes. Here, BWS is set to 1–3 in our experiments. Otherwise, there will be a large computation cost during the examination of the sentence combinations.

For the third step, we need to examine combinations of sentences from distinct derived nodes in the observation $BWS^{|\Omega|}$ times (Cartesian product of representative lists). If the number of selected sentences in a summary is about 10 (for example, the summaries in DUC 2005 and 2006, and definition QA in TREC crafted by human are about 10 sentences), we need to examine 2^{10}–3^{10}(=1024–59,049) combinations, which is computational feasible. Here, we need to pay attention to the problem that one selected sentence (base node) may be covered by more than one derived node in Ω.

Fig. 7 uses a simple example to demonstrate the procedure of summary generation based on DCL. In observation with nodes 11–13, suppose that the representatives of derived node 11 include base nodes 1 and 2 ($\mathrm{RP}(1) > \mathrm{RP}(2)$). If node 1 is selected to represent derived node 11, then the combination such as base nodes $\{1,6\}$ will become one of the summary candidates, as base node 1 could represent derived nodes 11 and 12, while base node 6 could represent derived node 13. On the contrary, if node 2 is considered as the representative of derived node 11, then the combination such as base nodes $\{2,4\}$ will become one of the summary candidates, as base node 2 could represent derived node 13 as well. It implies that (i) the dynamic program can explore promising sentence combinations from various local topics; and (ii) it is necessary to conduct such exploration since a better summary with larger RP may be detected when $\mathrm{BWS} > 1$.

According to the above discussion, we outline the algorithm of the summary generation based on DCL called *DCLSummarizer2*, as shown in Algorithm 1. In fact, the algorithm *DCLSummarizer* proposed by Ye et al. (2006) can be considered as a special case with $\mathrm{BWS} = 1$.

```
Algorithm 1 DCLSummarizer2
Input: $DCL, |\mathbb{A}|, BWS$
Output: summary $\mathbb{A}$
  1: $len \leftarrow |\mathbb{A}|$
  2: $COL \leftarrow \phi$
  3: recursively generate the series of observations until the number of derived nodes in the last observa-
     tion is larger than $len$
```

[4] Actually, the nodes in each observation should be sorted by their RPs. Here, we just roughly sort them for legibility.

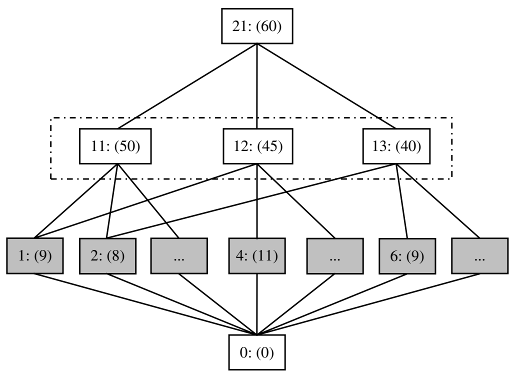


Fig. 7. A simple example for the generation of sample candidates. Here, the nodes with light or dark background are derived nodes or based nodes, respectively. The numbers in parentheses are their RPs. Derived nodes 11–13 surrounded by the dash dot rectangle constitute the current observation.

```
4: $\Omega \leftarrow$ {top $len$ derived nodes in the last observation}
5: denote the base node covered by the $i$th derived node in $\Omega$ with the $j$th maximal RP by $B_i^j (1 \leqslant i \leqslant len, 1 \leqslant j \leqslant BWS)$
6: construct $len \times BW\ S$ sets of summary candidates using base nodes such as $\langle B_1^j \times \cdots \times B_{len}^j \rangle, (1 \leqslant j \leqslant BWS)$, where $B_i^j$ whose derived node also covers any previously selected base node will be ignored. We then put them into $COL$
7: select the candidate in $COL$ with the maximal sum of RPs as $\mathbb{A}$
8: **if** the sentence number of $\mathbb{A}$ is equal to $|\mathbb{A}|$ **then**
9:    return $\mathbb{A}$
10: **else**
11:    $len \leftarrow len + 1$; **goto** 1
12: **end if**
```

In algorithm *DCLSummarizer2*, the input includes the expected sentence number of output summary $|\mathbb{A}|$ and BWS in each derived node. Steps 3–4 will generate the set of non-overlapping derived nodes Ω with maximal RPs. These nodes might lie in diverse layers in DCL. The collection COL containing $len \times$ BWS sets of summary candidates will be generated in step 6. During the construction of each summary candidate using a set of representatives with top RPs covered by distinct derived nodes in Ω, the selection of the remaining representatives must constitute an optimal policy with regard to those previously selected. Namely, if there exists a selected representative covered by the derived node under the current consideration in Ω, it is unnecessary to add any representative for this node due to sentence diversity. The algorithm employs a dynamic program to achieve this function. The summary with maximal sum of RPs in COL might not gather sufficient sentences (base nodes) when one representative is covered by more than one derived node in Ω. We may thus need to enlarge the size of Ω to allow more local topics to be considered if necessary (in steps 8–12).

5.1. Comparison between DCL and sentence clustering for summarization

In sentence clustering for extractive generic text summarization, such as Hatzivassiloglou, Klavans, Holcombe, Barzilay, and Kan (2001), Nomoto and Matsumoto (2001), Erkan and Radev (2004b), Radev et al. (2004), and so on, the systems collect representative sentences from a set of sentence clusters, where clustering parameters and configuration usually need to be carefully tuned. As compared with the sentence clustering approaches, our DCL approach has the following characteristics.

First, most clustering methods use similarity metric based on the cosine distance of terms. This might worsen the problem and affect the quality of clustering. DCL focuses on concepts, which are terms in open class after the expression variations (such as synonymy and alias) have been removed. DCL is more semantic-oriented and use only reliable features.

Second, since the distribution of some sentences in vector space (such as VSM) is sparse, the number of sentences per cluster vary widely. As a brief, clustering is often not accurate; clusters may contain spurious sentences, and the cluster size may exaggerate its importance (Blair-Goldensohn et al., 2004). It is thus difficult to estimate a good cluster number to be used and predict the quality of clustering, although we can ignore some odd sentences using heuristic rules. DCL can handle these problems naturally, since it does not need the exact number of clusters, and all involved local topics are organized in a hierarchy. We can also change the level of the observation with various derived nodes which will render the summaries with different lengthes.

Third, we may use a hierarchical clustering algorithm to segment the sentences (Kummamuru, Lotlikar, Roy, Singal, & Krishnapuram, 2004), where every sentence and every sub-cluster will appear in a special (super)-cluster. This is too restrictive in some cases. For example, a long sentence or sub-topic may cross two topics. We cannot say which assignment is more reasonable. In DCL, a lower node could have more than one upper node; and one sentence (base node) could be covered by multiple derived nodes. This is consistent with our common sense, namely, users can sum up a document according to their observation and requirement. For example, there are two kinds of summaries, specific or general, for a document cluster (Dang, 2005). DCL can exhibit more flexibility than clustering does, where users can interpret it freely.

Fourth, the clustering-based approaches usually pick up more than one sentence from each cluster as the number of sentences required in the summary is larger than that of clusters. Although MMR (Carbonell & Goldstein, 1998) or other strategies such as Cluster-based relative utility and Cross-sentence informational subsumption (Radev et al., 2004) might be used for redundancy penalty among the selected sentences, they tend to achieve a local optimization for sentence selection rather than a global optimization. DCL, on the other hand, will synchronously select a batch of representative sentences from distinct and salient local topics, where all selected sentences will work as a team to facilitate the minimal answer loss according to our evaluation framework.

6. Experiments

6.1. The system and test data

We built the test platform, named *OnModer*, based on the systems that we developed to participate DUC 2005 and 2006 evaluation. Unlike most existing approaches, we did not used heuristics such as sentence position, length, centroid, title overlap and even cue phrases, although these heuristics have been reported to have large contributions to system performance (Erkan & Radev, 2004a, 2004b; Hatzivassiloglou et al., 2001; Nobata & Sekine, 2004). Here, we focus on examining the efficiency of our proposed DCL model for summarization.

The input to *OnModer* comprises a cluster of relevant documents and an optional topic. At the preprocessing phase, *OnModer* ignores the document boundaries in the document cluster. It takes all the documents as a single document which it delimits into sentences for further analysis. The topic is treated similarly: only its sentences boundaries, if any, are detected. No other features of the topic are collected.

The main components of *OnModer* focus on the functions related to concept identification and semantic equivalence discovery, the construction of DCL as well as the algorithm of summary generation. *OnModer* summarizes the cluster according to the following workflow:

(1) After the *tokenizer* delimits numbers, words, and punctuations under the given format, a *sentence delimiter* detects and annotates sentence boundaries.
(2) A shallow parser, *NLProcessor* (available at http://www.infogistics.com/textanalysis.html), outputs the part-of-speech of all words. We thus obtain all open-class terms.
(3) We use the method described in Section 3 to detect semantic equivalence for all existing concepts. Here the threshold for semantic equivalence is set to 0.35 according to our initial tuning to achieve better ROUGE scores.

(4) We build a DCL as described in Section 4.1, where the minimal frequency of concepts is set to 3. The sparse concepts with frequency of 1–2 are ignored to save computation cost.
(5) We utilize the algorithm *DCLSummarizer2* (Section 5) to generate the desired summary.

The reasons of using corpora from DUC 2005 and 2006 to test *OnModer* are twofold. First, many active teams who work on summarization participated in DUC, and we can compare the performance of *OnModer* with the leading systems directly. Second, a number of valuable hand-crafted summaries are available, and we can use ROUGE to automatically evaluate the performance of *OnModer*. The corpus of DUC 2005 consists of 50 document clusters from *Financial Times of London and Los Angeles Times*. The average number of documents and size per cluster are about 31.9 and 144k, respectively. The corpus of DUC 2006 consists of 50 document clusters from the *AQUAINT* corpus, comprising newswire articles from the *Associated Press and New York Times* (1998–2000) and *Xinhua News Agency* (1996–2000). The average number of documents and size per cluster is 24.6 and 110k, respectively.

Incidently, since the length of summaries in DUC 2005 and 2006 is specified by word number (250 words) rather than sentence number, we can estimate the possible number of sentences (e.g., 7–10) as the parameter for our algorithm *DCLSummarizers* and then examine whether the number of words in the output summary is proper. We also ignore the granularity of summary (general or specific) described in DUC 2005 for simplicity as there is no such information in DUC 2006. Thus, there is no significant difference between DUC 2005 corpus and DUC 2006 corpus. We collectively call them DUC corpora.

6.2. The testing

Fig. 8 presents the results of all submitted systems for DUC 2005 and 2006 using the performance metrics of ROUGE-2 and ROUGE-SU4 recall. We can see that our DCL-based system achieve the best ROUGE scores (ROUGE-2 Recall = 7.17% and ROUGE-SU4 Recall = 13.16%) in DUC 2005 (Dang, 2005), and the second best ROUGE scores (ROUGE-2 Recall = 8.99% and ROUGE-SU4 Recall = 14.75%) in DUC 2006 based on the DCL with BWS = 1 (Dang, 2006). In this paper, we will report follow-on tests to better examine the efficiency and effectiveness of our proposed DCL model.

6.2.1. Answer loss

The significance and RP used in DCL for the selection of salient derived nodes and their representative sentences are based on our evaluation framework. We use randomly selected sentences as the baseline to verify the answer loss in the selected sentences based on DCL. We randomly select 10 clusters from DUC corpora for this test. We utilize DCL to generate summaries with various $|\mathbb{A}|/|\mathbb{D}|$ and then compare them with the baseline with the same sentence number.

Fig. 9 illustrates the answer loss in DCL-based sentences and baseline with various sentence numbers, where ROUGE recalls are obtained under BDS = 1. We find that there are much answer loss in the baseline, where $L(\mathbb{A}/\mathbb{D})$ is nearly linear to $|\mathbb{A}|/|\mathbb{D}|$. On the other hand, the sentences generated by DCL could achieve very low answer loss even though $|\mathbb{A}|/|\mathbb{D}|$ is very small. For instance, when 8–12 sentences ($|\mathbb{A}|/|\mathbb{D}| = 1\%$) are selected, the sentences generated by DCL could achieve an $L(\mathbb{A}/\mathbb{D})$ of about 45%, but the answer loss in the baseline reaches about 98%. This implies that DCL could collect sentences with abundant repeated concepts and generate summary with minimal answer loss.

6.2.2. The effectiveness of different text representations

Besides concepts, we could also employ other text representations, such as the original words and open-class terms, to build a lattice to generate local topics and index sentences. In order to examine the performance of DCL with different text representations, we utilize the same 10 randomly-selected clusters as above to perform the test.

When the original words, open-class terms and concepts are used to build DCLs, we notice that the ROUGE scores of DCL (BWS = 1) increases gradually, while the average dimensions of original words, open-class terms and concepts existing in each document cluster decreases gradually (see Table 2). However,

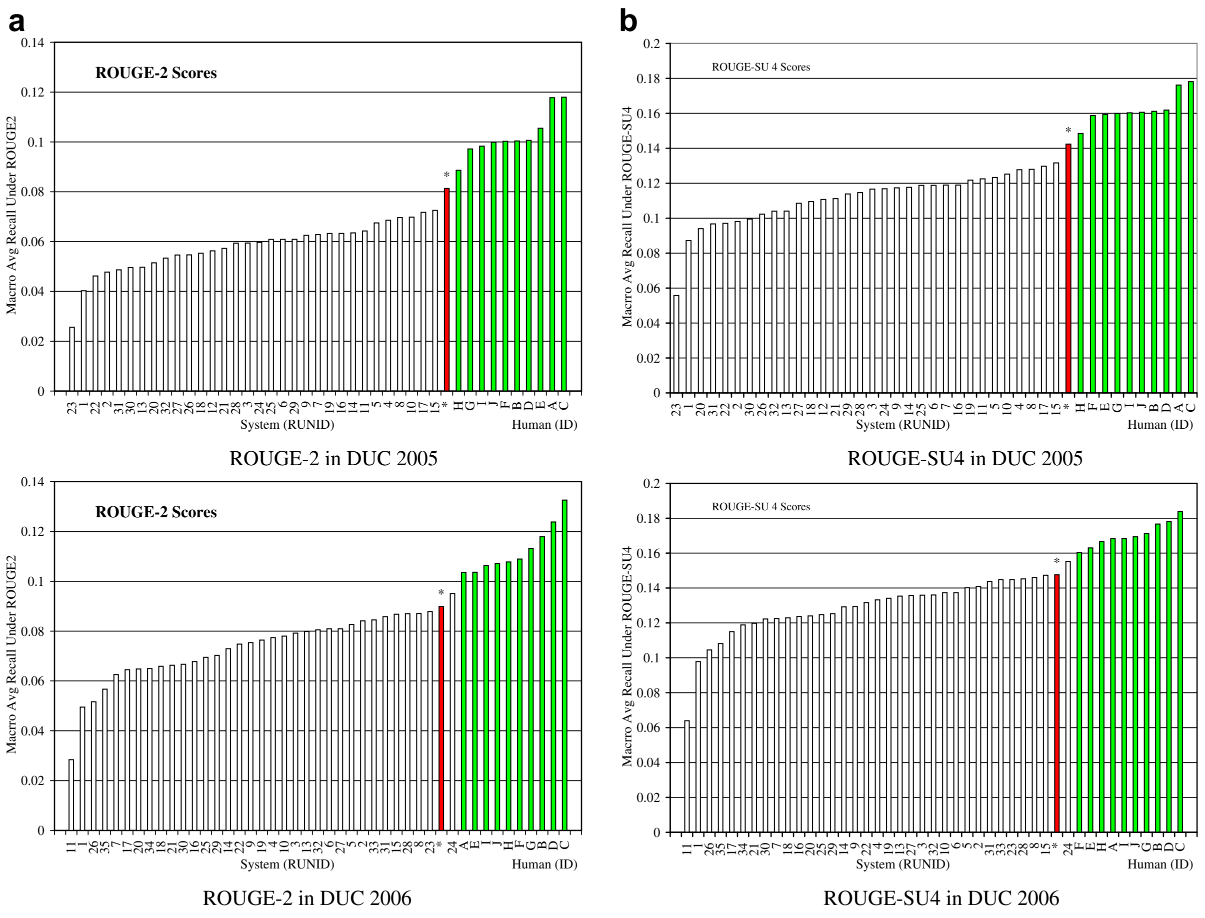


Fig. 8. Multiple comparison of systems in DUC 2005 and 2006 based on ROUGE scores. The column marked by "*" are the scores of our system. The columns with light background are scores of peer systems, and the columns with dark background are scores of human summaries.

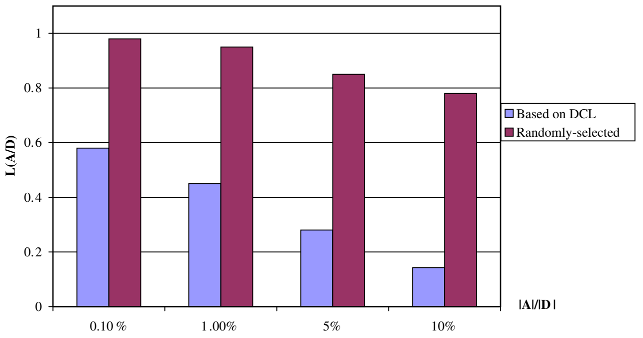


Fig. 9. DCL contribution in terms of answer loss.

the number of derived nodes and the average frequency per derived node in DCL built by concepts are the largest. This may indicate that there is more coherence among the sentences generated when concepts are used to render the underlying entities and their actions. Furthermore, this leads to better performance of summarization.

Table 2
The features of DCL represented by original words, open-class words and concepts

	By original words	By open-class words	By concepts
Dimension	893.8	824.3	743.7
Number of derived nodes	1978.3	1894.2	2340.0
Avg. frequency per derived node	3.65	3.97	6.03
ROUGE-2 Recall (%)	6.21	6.73	8.87
ROUGE-SU4 Recall (%)	10.31	11.84	14.68

In order to fairly compare these text representations, we ignore all words and concepts whose occurrences are less than 3.
We ignore the stop words in the original words as well.

6.2.3. The performance of DCL using different settings

To comprehensively evaluate the DCL-based summarization, we utilize the whole DUC corpora to test the system.

Table 3 lists the ROUGE recalls of DCL with various BWS (=1–3), as well as the average of human-crafted summaries and the best performing systems from DUC 2005 and 2006. We find that the performance of DCL is competitive: the ROUGE scores of DCL with BWS = 1 were in the second best among the DUC 2006 participants (with ROUGE-2 and ROUGE-SU4 scores of 8.99% and 14.76%, respectively), and the method based on concept link achieved the best ROUGE scores (with ROUGE-2 and ROUGE-SU4 of 8.13% and 14.24% respectively) in DUC 2005. For follow-on tests with settings such as BWS = 2 and 3, DCL obtains even better ROUGE scores. In particular, when BWS is set to 3, the DCL system achieves the best performance.

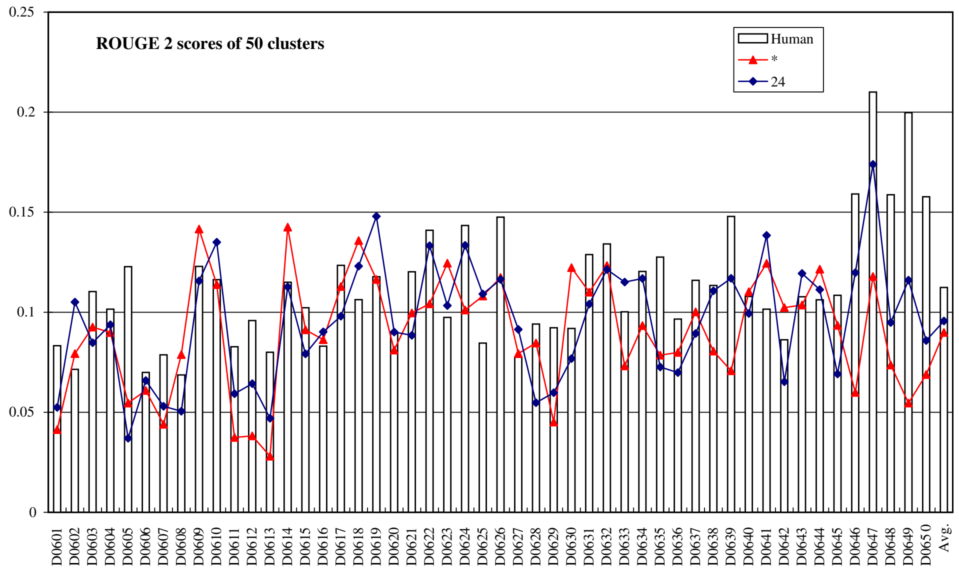
At the same time, both our previous system in DUC 2005 (Ye et al., 2005) and DCL use the concepts as text representation. The difference is that the former system collects the sentences for a summary, one at a time, without global optimization where sentences are ranked by a modified MRR; while the latter investigates the combinations of a set of representatives from distinct local topics globally. The comparison of the ROUGE scores between NUS3 and DCL on DUC 2005 corpus suggests that the global optimization approach can achieve better performance. Meanwhile, when we change the sentence number of a summary in some document clusters, we observe that the selected sentences existing in shorter summaries may not appear in longer ones. The reason for this is that such sentences is too nuclear to be covered by smaller local topics in the observation for a shorter summary. It seems that this phenomenon never happens in the most existing summarization approaches based on sentence-sorting.

6.3. Discussion

When we look into DUC corpora, we find that the performance of summarization in each cluster vary largely. For instance, as shown in Fig. 10, the variations of ROUGE scores of humans, DCL with BWS = 1 and the best system (Jagarlamudi, Pingali, & Varma, 2006) in different clusters in DUC 2006 corpus are large, and the estimated standard deviations reach 0.030, 0.029 and 0.029 in ROUGE-2, respectively. The difference between humans and machines per cluster are large as well. For example, the average ROUGE-2 difference per cluster between the human and our system is 0.032, although the difference of average ROUGE-2 scores of all clusters is 0.0225. In about 10 clusters, the machine-crafted summaries outperform the human-crafted

Table 3
The overall performance of DCL on DUC corpora measured by ROUGE recall

	DUG 2005		DUG 2006	
	ROUGE-2 (%)	ROUGE-SU4 (%)	ROUGE-2 (%)	ROUGE-SU4 (%)
Avg. of human-crafted	10.26	16.24	11.26	17.06
Best system	8.13	14.24	9.51	15.50
DCL with BWS = 1	8.43	14.57	8.99	14.76
DCL with BWS = 2	8.71	14.93	9.78	15.30
DCL with BWS = 3	8.79	15.10	9.82	15.83



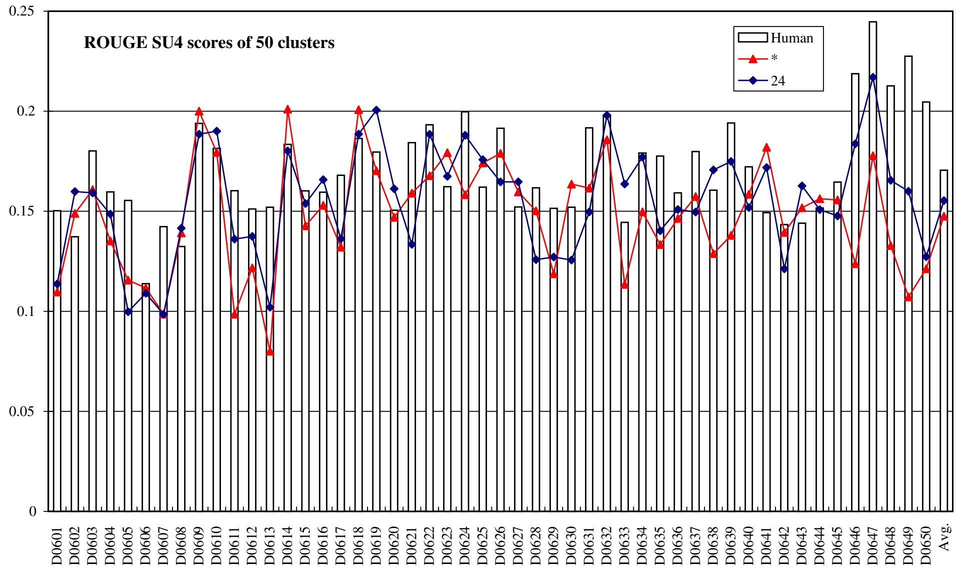


Fig. 10. ROUGE scores in 50 clusters in DUC 2006. The columns labeled by “Human” are the average scores of 10 human summaries (Since every person crafted only about 20 clusters, all persons are considered as a virtual one with average scores), and the lines labeled “24” and “*” are the ROUGE scores of the best performing system and our system, respectively.

ones in terms of ROUGE scores. However, it seems that we cannot say that the quality of the machine-crafted summaries is better than that of the human-crafted ones in these clusters. It may just imply that the current technologies are far from reliable summarization as human-crafted summaries still cannot achieve good consistency among each other. Analyzing the reasons among the cases with lowest and highest performance may be helpful for further research.

At present, we just consider the open-class terms as concepts. However, pronouns also refer to the entities and their actions depicted in the story. We guess that if pronouns can be replaced by the corresponding phrases, more sharing concepts and the sentences with stronger coherence tend to be detected. In future investigation, we would like to use deep-parsing to handle coreference and we expect it to contribute to the overall performance of DCL.

As we have previously discussed, DCL and its algorithm for summarization are based on the evaluation framework, which pursues a compression version of the original document(s) with minimal answer loss. The significant advantage of this evaluation framework is that it does not need human-crafted summary involved. We also would like to investigate its relationship to other classic evaluation methods such as ROUGE, Pyramid and Basic Elements in the future.

7. Conclusion

Following our work in DUC 2005 and 2006, we proposed a document concept lattice (DCL) model and the corresponding algorithm for summarization. DCL indexes all existing sentences in the concept space so that the sentences for a good summary can be optimized from promising sub-spaces defined by local topics. Here, each concept refers to an entity and its action possibly expressed by different terms, and we unify these diverse terms to form the identical concept by semantic equivalence discovery. Every derived node in DCL specifies a local topic defined by a set of high-frequency concepts and covers the sentences sharing such concepts. All derived nodes render overlapping local topics with various granularity, which will form a lattice under partial ordering relation as a whole. The sub-spaces of the observation for summary generation consist of a set of non-overlapping local topics with maximal representative powers retrieved from DCL. We design a summarization algorithm based on DCL that explores the combinations of representative sentences from the distinct local topics in the observation. The output summary having the maximal sum of representative powers is selected from the promising summary candidates. The experiment results of our participation of DUC 2005 and 2006 evaluations, along with follow-on tests, demonstrate that DCL is a competitive model for summarization as compared with existing technologies based on sentence-clustering and sentence-sorting.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In J. B. Bocca, M. Jarke, & C. Zaniolo (Eds.), *VLDB'94 Proceedings of the 20th international conference on very large data bases* (pp. 487–499). San Francisco, CA, USA: Morgan Kaufmann.

Barzilay, R., & Elhadad, M. (1997). Using lexical chains for text summarization. In *Proceedings of the intelligent scalable text summarization workshop (ISTS'97)*. Madrid, Spain: ACL.

Blair-Goldensohn, S., Evans, D., Hatzivassiloglou, V., McKeown, K., Nenkova, A., & Passonneau, R. (2004). Columbia University at DUC 2004, *DUC*.

Bosma, W., (2005). Extending answers using discourse structure. In: Saggion, H., & Minel, J.-L. (Eds.), *RANLP workshop on crossing barriers in text summarization research*. Incoma Ltd., Borovets, Bulgaria, pp. 2–9, ISBN 954-90906-8-X.

Carbonell, J., & Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*. pp. 335–336.

Dang, H.T. (2005). Overview of DUC 2005. In *DUC*. URL http://duc.nist.gov/.

Dang, H.T. (2006). Overview of DUC 2006 (draft). In *DUC*.

Doran, W., Stokes, N., Carthy, J., & Dunnion, J. (2004). Comparing lexical chain-based summarisation approaches using an extrinsic evaluation. In *Global WordNet Conference (GWC)*.

Erkan, G., & Radev, D. R. (2004b). The University of Michigan at DUC 2004. In *DUC*.

Erkan, G., & Radev, D. R. (2004a). LexRank: Graph-based lexical centrality as salience in text summarization. *Artificial Intelligence Research, 22*, 457–479.

Harabagiu, S., & Lacatusu, F. (2005). Topic themes for multi-document summarization. In *SIGIR'05: Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 202–209). New York, NY, USA: ACM Press.

Hatzivassiloglou, V., Klavans, J. L., Holcombe, M. L., Barzilay, R., Kan, M.-Y., & McKeown., K. R. (2001). Simfinder: A flexible clustering tool for summarization. In *Workshop on summarization in NAACL*.

Hovy, E., Lin, C.-Y., & Zhou, L. (2005). Evaluating DUC 2005 using basic elements. In *DUC*.

Jagarlamudi, J., Pingali, P., & Varma, V. (2006). Query independent sentence scoring approach to DUC 2006. In *DUC*.

Katz, S. (1996). Distribution of content words and phrases in text and language modelling. *Natural Language Engineering, 2*(1), 15–60.

Kummamuru, K., Lotlikar, R., Roy, S., Singal, K., & Krishnapuram, R. (2004). A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *WWW'04: Proceedings of the 13th international conference on world wide web* (pp. 658–665). New York, NY, USA: ACM Press.

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *SIGDOC*. pp. 24–26.

Lin, D. (1998). An information-theoretic definition of similarity. In *15th International conference on machine learning* pp. 296–304.

Lin, C.-Y., & Hovy, E. (2000a). The automated acquisition of topic signatures for text summarization, Vol. 1. Saarbrüken, pp. 495–500.

Lin, C.-Y., & Och, F. J. (2004). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *ACL*. URL http://www.isi.edu/~cyl/ROUGE/.

Lin, C.-Y., & Hovy, E. (2000b). The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th conference on computational linguistics* (pp. 495–501). Morristown, NJ, USA: Association for Computational Linguistics.

Mani, I., & Maybury, M. T. (1999). *Advances in automatic text summarization*. The MIT Press.

Mann, W., & Thompson, S. (1988). Rhetorical structure theory: Towards a functional theory of text organization. *Text, 8*(3), 243–281.

Marcu, D. (1997). The rhetorical parsing of unrestricted natural language texts. In P. R. Cohen & W. Wahlster (Eds.), *Proceedings of the thirty-fifth annual meeting of the association for computational linguistics and eighth conference of the European chapter of the association for computational linguistics* (pp. 96–103). Somerset, NJ: Association for Computational Linguistics.

Marcu, D. (1998). To build text summaries of high quality, nuclearity is not sufficient, Stanford, CA, USA, pp. 1–8 (March 23–25).

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990). Introduction to wordnet: An on-line lexical database*. *International Journal of Lexicography, 3*(4), 235–244.

Morris, J., & Hirst, G. (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computer Linguistics, 17*(1), 21–48.

Naphade, M. R., & Smith, J. R. (2004). On the detection of semantic concepts at trecvid. In *MULTIMEDIA'04: Proceedings of the 12th annual ACM international conference on multimedia* (pp. 660–667). New York, NY, USA: ACM Press.

Nguifo, E. M., Duquenne, V., & Liquiere, M. (2003). Introduction – Concept latticebased theory, methods and tools for knowledge discovery in databases: Applications. *Applied Artificial Intelligence, 17*(3), 177–180.

Nobata, C., & Sekine, S. (2004). CRL/NYU summarization system at DUC-2004. In *DUC*.

Nomoto, T., & Matsumoto, Y. (2001). A new approach to unsupervised text summarization. In *SIGIR* (pp. 26–34). New York, NY, USA: ACM Press.

Passonneau, R. J., Nenkova, A., McKeown, K., & Sigelman, S. (2005). Applying the pyramid method in DUC 2005. In *DUC*.

Radev, D., Jing, H., & Budzikowska, M. (2004). Centroid-based summarization of multiple documents. *Information Processing and Management, 40*, 919–938.

Schiffman, B., Nenkova, A., & McKeown, K. (2002). Experiments in multi-document summarization. In *HLT*.

Silber, H. G., & McCoy, K. F. (2000). An efficient text summarizer using lexical chains. In *INLG'00: Proceedings of the first international conference on natural language generation* (pp. 268–271). Morristown, NJ, USA: Association for Computational Linguistics.

Stumme, G., Wille, R., & Wille, U. (1998). Conceptual knowledge discovery in databases using formal concept analysis methods. *Lecture Notes in Computer Science*, 450–458.

Voorhees, E. M. (2001). Question answering in TREC. In *CIKM '01: Proceedings of the tenth international conference on information and knowledge management* (pp. 535–537). New York, NY, USA: ACM Press.

Wang, H., Dubitzky, W., Düntsch, I., & Bell, D. (1999). Lattice machine approach to automated casebase design: Marrying lazy and eager learning. In *IJCAI*. pp. 254–259.

Ye, S., Qiu, L., Chua, T.-S., & Kan, M.-Y. (2005). NUS at DUC 2005: Understanding documents via concept links. In *DUC*.

Ye, S., Qiu, L., Chua, T.-S., & Kan, M.-Y. (2006). NUS at DUC 2006: Document concept lattice for summarization. In *DUC*.