Single Document Summarization based on Nested Tree Structure

Yuta Kikuchi[†] **Tsutomu Hirao**[‡] **Hiroya Takamura**[†] **Manabu Okumura**[†] **Masaaki Nagata**[‡]
[†]Tokyo Institute of technology
4295, Nagatsuta, Midori-ku, Yokohama, 226-8503, Japan
{kikuchi,takamura,oku}@lr.pi.titech.ac.jp
[‡]NTT Communication Science Laboratories, NTT Corporation
2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0237, Japan
{hirao.tsutomu,nagata.masaaki}@lab.ntt.co.jp

Abstract

Many methods of text summarization combining sentence selection and sentence compression have recently been proposed. Although the dependency between words has been used in most of these methods, the dependency between sentences, i.e., rhetorical structures, has not been exploited in such joint methods. We used both dependency between words and dependency between sentences by constructing a nested tree, in which nodes in the document tree representing dependency between sentences were replaced by a sentence tree representing dependency between words. We formulated a summarization task as a combinatorial optimization problem, in which the nested tree was trimmed without losing important content in the source document. The results from an empirical evaluation revealed that our method based on the trimming of the nested tree significantly improved the summarization of texts.

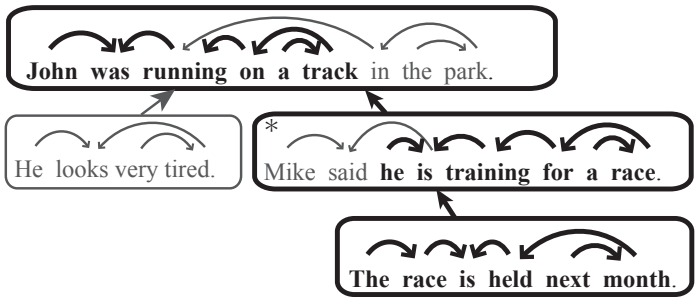
1 Introduction

Extractive summarization is one well-known approach to text summarization and extractive methods represent a document (or a set of documents) as a set of some textual units (e.g., sentences, clauses, and words) and select their subset as a summary. Formulating extractive summarization as a combinational optimization problem greatly improves the quality of summarization (McDonald, 2007; Filatova and Hatzivassiloglou, 2004; Takamura and Okumura, 2009). There has recently been increasing attention focused on approaches that jointly optimize sentence extraction and sentence compression (Tomita et al., 2009; Qian and Liu, 2013; Morita et al., 2013; Gillick and Favre, 2009; Almeida and Martins, 2013; Berg-Kirkpatrick et al., 2011). We can only extract important content by trimming redundant parts from sentences.

However, as these methods did not include the discourse structures of documents, the generated summaries lacked coherence. It is important for generated summaries to have a discourse structure that is similar to that of the source document. Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) is one way of introducing the discourse structure of a document to a summarization task (Marcu, 1998; Daumé III and Marcu, 2002; Hirao et al., 2013). Hirao et al. recently transformed RST trees into dependency trees and used them for single document summarization (Hirao et al., 2013). They formulated the summarization problem as a tree knapsack problem with constraints represented by the dependency trees.

We propose a method of summarizing a single document that utilizes dependency between sentences obtained from rhetorical structures and dependency between words obtained from a dependency parser. We have explained our method with an example in Figure 1. First, we represent a document as a **nested tree**, which is composed of two types of tree structures: a **document tree** and a **sentence tree**. The document tree is a tree that has sentences as nodes and head modifier relationships between sentences obtained by RST as edges. The sentence tree is a tree that has words as nodes and head modifier relationships between words obtained by the dependency parser as edges. We can build the nested tree by regarding each node of the document tree as a sentence tree. Finally, we formulate the problem of single document summarization as that of combinatorial optimization, which is based on the trimming of the nested tree.

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 315–320,
Baltimore, Maryland, USA, June 23-25 2014.



Source document
John was running on a track in the park.
He looks very tired.
Mike said he is training for a race.
The race is held next month.

Summary
John was running on a track.
he is training for a race. *
The race is held next month.

Figure 1: Overview of our method. The source document is represented as a nested tree. Our method simultaneously selects a rooted document subtree and sentence subtree from each node.

Our method jointly utilizes relations between sentences and relations between words, and extracts a rooted document subtree from a document tree whose nodes are arbitrary subtrees of the sentence tree.

Elementary Discourse Units (EDUs) in RST are defined as the minimal building blocks of discourse. EDUs roughly correspond to clauses. Most methods of summarization based on RST use EDUs as extraction textual units. We converted the rhetorical relations between EDUs to the relations between sentences to build the nested tree structure. We could thus take into account both relations between sentences and relations between words.

2 Related work

Extracting a subtree from the dependency tree of words is one approach to sentence compression (Tomita et al., 2009; Qian and Liu, 2013; Morita et al., 2013; Gillick and Favre, 2009). However, these studies have only extracted rooted subtrees from sentences. We allowed our model to extract a subtree that did not include the root word (See the sentence with an asterisk $*$ in Figure 1). The method of Filippova and Strube (2008) allows the model to extract non-rooted subtrees in sentence compression tasks that compress a single sentence with a given compression ratio. However, it is not trivial to apply their method to text summarization because no compression ratio is given to sentences. None of these methods use the discourse structures of documents.

Daumé III and Marcu (2002) proposed a noisy-channel model that used RST. Although their method generated a well-organized summary, no optimality of information coverage was guaranteed and their method could not accept large texts because of the high computational cost. In addition, their method required large sets of data to calculate the accurate probability. There have been some studies that have used discourse structures locally to optimize the order of selected sentences (Nishikawa et al., 2010; Christensen et al., 2013).

- The scare over Alar, a growth regulator
- that makes apples redder and crunchier
- but may be carcinogenic,
- made consumers shy away from the Delicious,
- though they were less affected than the McIntosh.

Figure 2: Example of one sentence. Each line corresponds to one EDU.

3 Generating summary from nested tree

3.1 Building Nested Tree with RST

A document in RST is segmented into EDUs and adjacent EDUs are linked with rhetorical relations to build an RST-Discourse Tree (RST-DT) that has a hierarchical structure of the relations. There are 78 types of rhetorical relations between two spans, and each span has one of two aspects of a nucleus and a satellite. The nucleus is more salient to the discourse structure, while the other span, the satellite, represents supporting information. RST-DT is a tree whose terminal nodes correspond to EDUs and whose nonterminal nodes indicate the relations. Hirao et al. converted RST-DTs into dependency-based discourse trees (DEP-DTs) whose nodes corresponded to EDUs and whose edges corresponded to the head modifier relationships of EDUs. See Hirao et al. for details (Hirao et al., 2013).

Our model requires sentence-level dependency. Fortunately we can simply convert DEP-DTs to obtain dependency trees between sentences. We specifically merge EDUs that belong to the same sentence. Each sentence has only one root EDU that is the parent of all the other EDUs in the sentence. Each root EDU in a sentence has the parent

$$\text{max.} \quad \sum_{i}^{n}\sum_{j}^{m_i} w_{ij} z_{ij}$$

$$\begin{aligned}
\text{s.t.} \quad & \textstyle\sum_i^n \sum_j^{m_i} z_{ij} \leq L; && (1)\\
& x_{\text{parent}(i)} \geq x_i; && \forall i \quad (2)\\
& z_{\text{parent}(i,j)} - z_{ij} + r_{ij} \geq 0; && \forall i,j \quad (3)\\
& x_i \geq z_{ij}; && \forall i,j \quad (4)\\
& \textstyle\sum_j^{m_i} z_{ij} \geq \min(\theta, \text{len}(i)) x_i; && \forall i \quad (5)\\
& \textstyle\sum_j^{m_i} r_{ij} = x_i; && \forall i \quad (6)\\
& \textstyle\sum_{j \notin R_c(i)} r_{ij} = 0; && \forall i \quad (7)\\
& r_{ij} \leq z_{ij}; && \forall i,j \quad (8)\\
& r_{ij} + z_{\text{parent}(i,j)} \leq 1; && \forall i,j \quad (9)\\
& r_{i\text{root}(i)} = z_{i\text{root}(i)}; && \forall i \quad (10)\\
& \textstyle\sum_{j \in \text{sub}(i)} z_{ij} \geq x_i; && \forall i \quad (11)\\
& \textstyle\sum_{j \in \text{obj}(i)} z_{ij} \geq x_i; && \forall i \quad (12)
\end{aligned}$$

Figure 3: ILP formulation ($x_i, z_{ij}, r_{ij} \in \{0, 1\}$)

EDU in another sentence. Hence, we can determine the parent-child relations between sentences. As a result, we obtain a tree that represents the parent-child relations of sentences, and we can use it as a document tree. After the document tree is obtained, we use a dependency parser to obtain the syntactic dependency trees of sentences. Finally, we obtain a nested tree.

3.2 ILP formulation

Our method generates a summary by trimming a nested tree. In particular, we extract a rooted document subtree from the document tree, and sentence subtrees from sentence trees in the document tree. We formulate our problem of optimization in this section as that of integer linear programming. Our model is shown in Figure 3.

Let us denote by w_{ij} the term weight of word ij (word j in sentence i). x_i is a variable that is one if sentence i is selected as part of a summary, and z_{ij} is a variable that is one if word ij is selected as part of a summary. According to the objective function, the score for the resulting summary is the sum of the term weights w_{ij} that are included in the summary. We denote by r_{ij} the variable that is one if word ij is selected as a root of an extracting sentence subtree. Constraint (1) guarantees that the summary length will be less than or equal to limit L. Constraints (2) and (3) are tree constraints for a document tree and sentence trees. r_{ij} in Constraint (3) allows the system to extract non-rooted sentence subtrees, as we previously mentioned. Function parent(i) returns the parent of sentence i and function parent(i, j) returns the parent of word ij. Constraint (4) guarantees that words are only selected from a selected sentence. Constraint (5) guarantees that each selected sentence subtree has at least θ words. Function len(i) returns the number of words in sentence i. Constraints (6)-(10) allow the model to extract subtrees that have an arbitrary root node. Constraint (6) guarantees that there is only one root per selected sentence. We can set the candidate for the root node of the subtree by using constraint (7). The $R_c(i)$ returns a set of the nodes that are the candidates of the root nodes in sentence i. It returned the parser's root node and the verb nodes in this study. Constraint (8) maintains consistency between z_{ij} and r_{ij}. Constraint (9) prevents the system from selecting the parent node of the root node. Constraint (10) guarantees that the parser's root node will only be selected when the system extracts a rooted sentence subtree. The root(i) returns the word index of the parser's root. Constraints (11) and (12) guarantee that the selected sentence subtree has at least one subject and one object if it has any. The $sub(i)$ and $obj(i)$ return the word indices whose dependency tag is "SUB" and "OBJ".

3.3 Additional constraint for grammaticality

We added two types of constraints to our model to extract a grammatical sentence subtree from a dependency tree:

$$z_{ik} = z_{il}, \quad (13)$$

$$\sum_{k \in s(i,j)} z_{ik} = |s(i,j)| x_i. \quad (14)$$

Equation (13) means that words z_{ik} and z_{il} have to be selected together, i.e., a word whose dependency tag is PMOD or VC and its parent word, a negation and its parent word, a word whose dependency tag is SUB or OBJ and its parent verb, a comparative (JJR) or superlative (JJS) adjective and its parent word, an article (a/the) and its parent word, and the word "to" and its parent word. Equation (14) means that the sequence of words has to be selected together, i.e., a proper noun sequence whose POS tag is PRP\$, WP%, or POS and a possessive word and its parent word and the words between them. The $s(i, j)$ returns the set of word indices that are selected together with word ij.

Table 1: ROUGE score of each model. Note that the top two rows are both our proposals.

	ROUGE-1
Sentence subtree	**0.354**
Rooted sentence subtree	0.352
Sentence selection	0.254
EDU selection (Hirao et al., 2013)	0.321
LEAD_{EDU}	0.240
LEAD_{snt}	0.157

4 Experiment

4.1 Experimental Settings

We experimentally evaluated the test collection for single document summarization contained in the RST Discourse Treebank (RST-DTB) (Carlson et al., 2001) distributed by the Linguistic Data Consortium (LDC) [1]. The RST-DTB Corpus includes 385 Wall Street Journal articles with RST annotations, and 30 of these documents also have one manually prepared reference summary. We set the length constraint, L, as the number of words in each reference summary. The average length of the reference summaries corresponded to approximately 10% of the length of the source document. This dataset was first used by Marcu et al. for evaluating a text summarization system (Marcu, 1998). We used ROUGE (Lin, 2004) as an evaluation criterion.

We compared our method (**sentence subtree**) with that of EDU selection (Hirao et al., 2013). We examined two other methods, i.e., **rooted sentence subtree** and **sentence selection**. These two are different from our method in the way that they select a sentence subtree. Rooted sentence subtree only selects rooted sentence subtrees [2]. Sentence selection does not trim sentence trees. It simply selects full sentences from a document tree[3]. We built all document trees from the RST-DTs that were annotated in the corpus.

We set the term weight, w_{ij}, for our model as:

$$w_{ij} = \frac{log(1 + tf_{ij})}{depth(i)^2}, \tag{15}$$

where tf_{ij} is the term frequency of word ij in a document and $depth(i)$ is the depth of sentence i within the sentence-level DEP-DT that we described in Section 3.1. For Constraint (5), we set θ to eight.

4.2 Results and Discussion

4.2.1 Comparing ROUGE scores

We have summarized the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) scores for each method in Table 1. The score for sentence selection is low (0.254). However, introducing sentence compression to the system greatly improved the ROUGE score (0.354). The score is also higher than that with EDU selection, which is a state-of-the-art method. We applied a multiple test by using Holm's method and found that our method significantly outperformed EDU selection and sentence selection. The difference between the sentence subtree and the rooted sentence subtree methods was fairly small. We therefore qualitatively analyzed some actual examples that will be discussed in Section 4.2.2. We also examined the ROUGE scores of two LEAD[4] methods with different textual units: EDUs (LEAD_{EDU}) and sentences (LEAD_{SNT}). Although LEAD works well and often obtains high ROUGE scores for news articles, the scores for LEAD_{EDU} and LEAD_{SNT} were very low.

4.2.2 Qualitative Evaluation of Sentence Subtree Selection

This subsection compares the methods of subtree selection and rooted subtree selection. Figure 4 has two example sentences for which both methods selected a subtree as part of a summary. The $\{\cdot\}$ indicates the parser's root word. The $[\cdot]$ indicates the word that the system selected as the root of the subtree. Subtree selection selected a root in both examples that differed from the parser's root. As we can see, subtree selection only selected important subtrees that did not include the parser's root, e.g., purpose-clauses and that-clauses. This capability is very effective because we have to contain important content in summaries within given length limits, especially when the compression ratio is high (i.e., the method has to generate much shorter summaries than the source documents).

[1] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002T07

[2] We achieved this by making $R_c(i)$ only return the parser's root node in Figure 7.

[3] We achieved this by setting θ to a very large number.

[4] LEAD methods simply take the first K textual units from a source document until the summary length reaches L.

Original sentence	:	John Kriz, a Moody's vice president, {said} Boston Safe Deposit's performance has been hurt this year by a mismatch in the maturities of its assets and liabilities.
Rooted subtree selection	:	John Kriz a Moody's vice president [{said}] Boston Safe Deposit's performance has been hurt this year
Subtree selection	:	Boston Safe Deposit's performance has [been] hurt this year
Original sentence	:	Recent surveys by Leo J. Shapiro & Associates, a market research firm in Chicago, {suggest} that Sears is having a tough time attracting shoppers because it hasn't yet done enough to improve service or its selection of merchandise.
Rooted subtree selection	:	surveys [{suggest}] that Sears is having a time
Subtree selection	:	Sears [is] having a tough time attracting shoppers

Figure 4: Example sentences and subtrees selected by each method.

Table 2: Average number of words that individual extracted textual units contained.

Subtree	Sentence	EDU
15.29	18.96	9.98

4.2.3 Fragmentation of Information

Many studies that have utilized RST have simply adopted EDUs as textual units (Mann and Thompson, 1988; Daumé III and Marcu, 2002; Hirao et al., 2013; Knight and Marcu, 2000). While EDUs are textual units for RST, they are too fine grained as textual units for methods of extractive summarization. Therefore, the models have tended to select small fragments from many sentences to maximize objective functions and have led to fragmented summaries being generated. Figure 2 has an example of EDUs. A fragmented summary is generated when small fragments are selected from many sentences. Hence, the number of sentences in the source document included in the resulting summary can be an indicator to measure the fragmentation of information. We counted the number of sentences in the source document that each method used to generate a summary[5]. The average for our method was 4.73 and its median was four sentences. In contrast, methods of EDU selection had an average of 5.77 and a median of five sentences. This meant that our method generated a summary with a significantly smaller number of sentences[6]. In other words, our method relaxed fragmentation without decreasing the ROUGE score. There are boxplots of the numbers of selected sentences in Figure 5. Table 2 lists the number of words in each textual unit extracted by each method. It indicates that EDUs are shorter than the other textual units. Hence, the number of sentences tends to be large.

[5]Note that the number for the EDU method is not equal to selected textual units because a sentence in the source document may contain multiple EDUs.

[6]We used the Wilcoxon signed-rank test ($p < 0.05$).

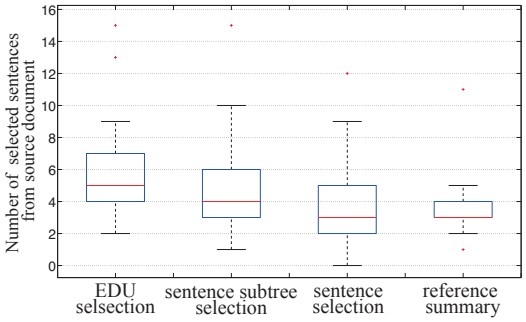


Figure 5: Number of sentences that each method selected.

5 Conclusion

We proposed a method of summarizing a single document that included relations between sentences and relations between words. We built a nested tree and formulated the problem of summarization as that of integer linear programming. Our method significantly improved the ROUGE score with significantly fewer sentences than the method of EDU selection. The results suggest that our method relaxed the fragmentation of information. We also discussed the effectiveness of sentence subtree selection that did not restrict rooted subtrees. Although ROUGE scores are widely used as evaluation metrics for text summarization systems, they cannot take into consideration linguistic qualities such as human readability. Hence, we plan to conduct evaluations with people[7].

We only used the rhetorical structures between sentences in this study. However, there were also rhetorical structures between EDUs inside individual sentences. Hence, utilizing these for sentence compression has been left for future work. In addition, we used rhetorical structures that were manually annotated. There have been related studies on building RST parsers (duVerle and Prendinger, 2009; Hernault et al., 2010) and by using such parsers, we should be able to apply our model to other corpora or to multi-document settings.

[7]For example, the quality question metric from the Document Understanding Conference (DUC).

References

Miguel Almeida and Andre Martins. 2013. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *ACL*, pages 196–206, August.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *ACL*, pages 481–490, Portland, Oregon, USA, June.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *SIGDIAL*, pages 1–10.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *NAACL:HLT*, pages 1163–1173.

Hal Daumé III and Daniel Marcu. 2002. A noisy-channel model for document compression. *ACL*, pages 449–456.

David duVerle and Helmut Prendinger. 2009. A novel discourse parser based on support vector machine classification. In *IJCNLP*, pages 665–673.

Elena Filatova and Vasileios Hatzivassiloglou. 2004. A formal model for information selection in multi-sentence text extraction. In *COLING*.

Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *INLG*, pages 25–32.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *ILP*, pages 10–18.

Hugo Hernault, Helmut Prendinger, David duVerle, and Mitsuru Ishizuka. 2010. Hilda: A discourse parser using support vector machine classification. *Dialogue & Discourse*, 1(3):1–30.

Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. 2013. Single-document summarization as a tree knapsack problem. In *EMNLP*, pages 1515–1520.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization - step one: Sentence compression. In *National Conference on Artificial Intelligence (AAAI)*, pages 703–710.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, pages 74–81.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, pages 243–281.

Daniel Marcu. 1998. Improving summarization through rhetorical parsing tuning. In *In Proc. of the 6th Workshop on Very Large Corpora*, pages 206–215.

Ryan T. McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *ECIR*, pages 557–564.

Hajime Morita, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2013. Subtree extractive summarization via submodular maximization. In *ACL*, pages 1023–1032.

Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Genichiro Kikui. 2010. Opinion summarization with integer linear programming formulation for sentence extraction and ordering. In *COLING*, pages 910–918.

Xian Qian and Yang Liu. 2013. Fast joint compression and summarization via graph cuts. In *EMNLP*, pages 1492–1502.

Hiroya Takamura and Manabu Okumura. 2009. Text summarization model based on the budgeted median problem. In *CIKM*, pages 1589–1592.

Kohei Tomita, Hiroya Takamura, and Manabu Okumura. 2009. A new approach of extractive summarization combining sentence selection and compression. *IPSJ SIG Notes*, pages 13–20.