Provable Fast Greedy Compressive Summarization with Any Monotone Submodular Function

Shinsaku Sakaue Tsutomu Hirao Masaaki Nishino Masaaki Nagata
NTT Communication Science Laboratories
{sakaue.shinsaku, hirao.tsutomu}@lab.ntt.co.jp
{nishino.masaaki, nagata.masaaki}@lab.ntt.co.jp

Abstract

Submodular maximization with the greedy algorithm has been studied as an effective approach to *extractive summarization*. This approach is known to have three advantages: its applicability to many useful submodular objective functions, the efficiency of the greedy algorithm, and the provable performance guarantee. However, when it comes to *compressive summarization*, we are currently missing a counterpart of the extractive method based on submodularity. In this paper, we propose a fast greedy method for compressive summarization. Our method is applicable to any monotone submodular objective function, including many functions well-suited for document summarization. We provide an approximation guarantee of our greedy algorithm. Experiments show that our method is about 100 to 400 times faster than an existing method based on integer-linear-programming (ILP) formulations and that our method empirically achieves more than 95%-approximation.

1 Introduction

Automatic document summarization continues to be a seminal subject of study in natural language processing and information retrieval (Luhn, 1958; Edmundson, 1969; Cheng and Lapata, 2016; Peyrard and Eckle-Kohler, 2017). Owing to the recent advances in data collection, the size of document data to be summarized has been exploding, which has been bringing a drastic increase in the demand for fast summarization systems.

Extractive summarization is a widely used approach to designing fast summarization systems. With this approach, we construct a summary by extracting some sentences from the original document(s). The extractive approach is not only fast but also has the potential to achieve state-of-the-art ROUGE scores (Lin, 2004), which was revealed by Hirao et al. (2017b). In many existing methods, sentences are extracted by solving various subset selection problems: for example, the *knapsack problem* (McDonald, 2007), *maximum coverage problem* (Filatova and Hatzivassiloglou, 2004; Takamura and Okumura, 2009a), *budgeted median problem* (Takamura and Okumura, 2009b), and *submodular maximization problem* (Lin and Bilmes, 2010). Of particular interest, the method based on submodular maximization has three advantages: (1) Many objective functions used for document summarization are known to be *monotone* and *submodular* (Lin and Bilmes, 2011; J Kurisinkel et al., 2016); examples of such functions include the *coverage function*, *diversity reward function*, and ROUGE. Therefore, the method can deliver high performance by using monotone submodular objective functions that are suitable for the given tasks. (2) The efficient greedy algorithm is effective for the submodular maximization problem, which provides fast summarization systems. (3) Theoretical performance guarantees of the greedy algorithm can be proved; for example, a $\frac{1}{2}(1 - e^{-1})$-approximation guarantee can be obtained.

Although the above extractive methods successfully obtain summaries with high ROUGE scores, they have the following shortcoming: A long sentence typically has redundant parts, which means a summary constructed simply by extracting some sentences often includes many redundant parts. As a result, if the limitation placed on summary length is tight, the extractive approach cannot yield an informative summary.

Compressive summarization is known to be effective in overcoming this problem. With this approach, a summary is constructed with some

Proceedings of NAACL-HLT 2018, pages 1737–1746
New Orleans, Louisiana, June 1 - 6, 2018.

compressed sentences, and thus we can obtain a concise and informative summary. To make compressed sentences, the dependency-tree-based approach (Filippova and Strube, 2008) is often used, which is advantageous in that each compressed sentence preserves its original dependency relations. Specifically, given a set of dependency trees constructed for sentences in the original documents, a summary is obtained by extracting some rooted subtrees; each subtree corresponds to a compressed sentence. Different from the extractive summarization, the dependency relations in each sentence must be taken into account, and hence the aforementioned extractive methods cannot be applied to compressive summarization. A number of methods have been proposed for compressive summarization (Berg-Kirkpatrick et al., 2011; Almeida and Martins, 2013; Morita et al., 2013; Kikuchi et al., 2014; Hirao et al., 2017a). These methods formulate summarization as a type of combinatorial optimization problem with a tree constraint, and they obtain summaries by solving the problem. Unfortunately, the existing methods have two drawbacks: (1) The class of objective functions to which they are applicable is limited; for example, they work only with the linear function or coverage function. As a result, the performance of these methods cannot be improved by elaborating the objective functions. (2) They contain costly procedures as their building blocks: integer-linear-programming (ILP) solvers, dynamic programming (DP) algorithms, and so on. Therefore, they are not fast enough to be applied to large-scale document data. In a nutshell, compressive summarization is currently missing a fast method that is applicable to a wide variety of objective functions.

1.1 Our Contribution

In this paper, we propose a submodularity-based greedy method for compressive summarization. Our method is, so to speak, a compressive counterpart of the greedy method for extractive summarization (Lin and Bilmes, 2010). Similar to the extractive method, our method has the three key advantages:

1. Our method works with any monotone submodular objective function, a wide class of useful objective functions, examples of which include the coverage function, ROUGE, and many others (Lin and Bilmes, 2011; J Kurisinkel et al., 2016).

2. Our method is faster than existing compressive summarization methods since it employs the efficient greedy algorithm. Specifically, given a set, V, of all textual units contained in the document data and a summary length limitation value, L, our method requires at most $O(L|V|)$ objective function evaluations. Experiments show that our method is about 100 to 400 times faster than the ILP-based method implemented with `CPLEX`.

3. A theoretical guarantee of our method can be proved; specifically, a $\frac{1}{2}(1 - e^{-1/\lambda})$-approximation guarantee can be obtained, where λ is a parameter defined from given document data (a definition is shown later). This result generalizes the $\frac{1}{2}(1 - e^{-1})$-approximation of the greedy algorithm for submodular maximization with a knapsack constraint (Leskovec et al., 2007). In experiments, our method achieved more that 95%-approximation. Furthermore, our method attained ROUGE_1 scores comparable to those of the ILP-based method.

1.2 Related Work

There are many existing methods for compressive summarization (Berg-Kirkpatrick et al., 2011; Almeida and Martins, 2013; Morita et al., 2013; Kikuchi et al., 2014; Hirao et al., 2017a), and they attempt to create summaries by solving optimization problems with a tree and length constraints. Unfortunately, these methods accept only a few objective functions.

A common approach is to use ILP formulations. Berg-Kirkpatrick et al. (2011) formulate the problem as an ILP with the coverage objective function, which is solved by using an ILP solver. Almeida and Martins (2013) also employs an ILP formulation and solves the problem via an algorithm based on *dual decomposition*, which runs faster than an ILP solver.[1] These ILP-based methods are optimal in terms of objective function values. However, it is hard to apply them to large-scale document data since to solve ILPs often takes long computation time.

[1]Their method was observed to be about 25 times faster than `GLPK`, a commonly used free ILP solver. On the other hand, `CPLEX`, which is a commercial ILP solver used in our experiments, was observed to be about 3 to 20 times faster than `GLPK`, and our method is about 100 to 400 times faster than `CPLEX`. Consequently, our method is estimated to be about 12 to 320 times faster than their method.

In an attempt to uncover the potential power of dependency-tree-based compressive summarization, Hirao et al. (2017a) solved ILPs with the ROUGE objective function with an ILP solver. Their method obtains summaries by directly maximizing the ROUGE score for given reference summaries (i.e., any other methods cannot achieve higher ROUGE scores than their method). The resulting summaries, called *oracle summaries*, were revealed to attain substantially high rouge scores, which implies that there remains much room for further research into compressive summarization.

A greedy method with a DP algorithm (Morita et al., 2013) is probably the closest one to our idea. Their method iteratively chooses compressed sentences in a greedy manner, for which a DP algorithm is employed. Thanks to the submodularity of their objective function, their method enjoys a $\frac{1}{2}(1-e^{-1})$-approximation guarantee. However, because of the costly DP procedure, their method is less scalable than the standard greedy methods such as the extractive method (Lin and Bilmes, 2010) and ours. Moreover, it is applicable only to objective functions that are designed for their problem settings; for example, it cannot use ROUGE as an objective function.

1.3 Overview of Our Approach

A high-level sketch of our approach is as follows: As in many existing works, we formulate the compressive summarization task as a combinatorial optimization problem with a tree constraint, which we call the *submodular tree knapsack problem* (STKP). STKP is generally NP-hard; in fact, it includes the knapsack problem and maximum coverage problem as special cases. Unfortunately, as we will see later, a naive greedy algorithm for STKP does not offer any approximation guarantee in general. The main difficulty with STKP is that its tree constraint is too complex. To avoid dealing with the complex constraint directly, we transform STKP into a special case of the *submodular cost submodular knapsack problem* (SCSKP) (Iyer and Bilmes, 2013). For general SCSKP, no approximation guarantee has been proved. Fortunately, in our case, a $\frac{1}{2}(1-e^{-1/\lambda})$-approximation can be proved by exploiting the structure of the resulting SCSKP. Thus we obtain a fast greedy method for compressive summarization, which works with various monotone submodular objective functions and enjoys an approximation guarantee.

2 Submodularity

Given finite set V (e.g., a set of chunks), set function $g : 2^V \to \mathbb{R}$ is said to be *submodular* if $g(A \cup B) + g(A \cap B) \leq g(A) + g(B)$ holds for any $A, B \subseteq V$. We define $g(A \mid B) \coloneqq g(A \cup B) - g(B)$. The submodularity is also characterized by the following *diminishing return property*: $g(\{v\} \mid A) \geq g(\{v\} \mid B)$ for any $A \subseteq B$ and $v \in V \backslash B$. Set function g is *monotone* if $g(A) \leq g(B)$ for any $A \subseteq B$. In this paper, we focus on monotone submodular functions such that $g(\emptyset) = 0$. The submodularity and monotonicity are a natural fit for document summarization; intuitively, the marginal gain, $g(\{v\} \mid S)$, of adding new chunk $v \in V$ to summary $S \subseteq V$ is small if S already has many chunks (submodularity), and a summary becomes more informative as it gets more chunks (monotonicity). In fact, as in (Lin and Bilmes, 2011), many objective functions well-suited for document summarization have submodularity and monotonicity; examples of such functions include the coverage function, diversity reward function, and ROUGE, to name a few.

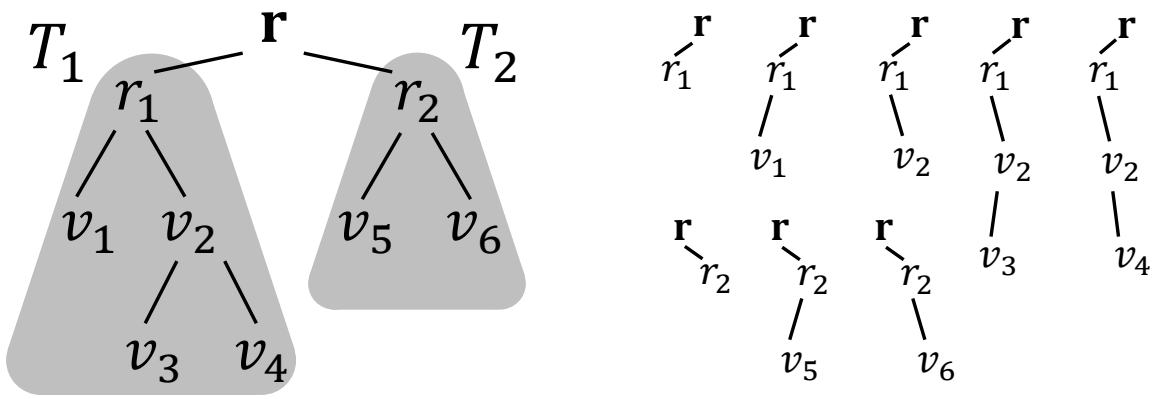
3 Problem Statements

We formulate the summarization task as the following subtree extraction problem called STKP hereafter. In what follows, we let $[M] \coloneqq \{1, \ldots, M\}$ for any positive integer M.

We attempt to summarize document data consisting of N sentences. Each sentence forms a dependency tree, which can be constructed by using existing methods (e.g., (Filippova and Strube, 2008; Filippova and Altun, 2013)). For convenience, we call the dependency tree of a sentence the *sentence tree*. The i-th sentence ($i \in [N]$) yields sentence tree $T_i = (V_i, E_i)$ rooted at $r_i \in V_i$, where V_i is a set of textual units (e.g., words or chunks) contained in the i-th sentence, and edges in E_i represent their dependency relations. We define a *document tree* with a dummy root vertex $\mathbf{r}$ as $\mathbf{T} \coloneqq (\{\mathbf{r}\} \cup V, E)$, where V and E are vertex and edge sets, respectively, defined as follows:

$$V \coloneqq \bigcup_{i \in [N]} V_i, \quad E \coloneqq \bigcup_{i \in [N]} \{E_i \cup \{(\mathbf{r}, r_i)\}\}.$$

Namely, V is the set of all textual units contained in the document data, and edges in E represent the dependency relations as well as the relations between $\mathbf{r}$ and r_i, with which the multiple sentence



(a) Document tree $\mathbf{T} = (\{\mathbf{r}\} \cup V, E)$

(b) Set of all paths rooted at $\mathbf{r}$

Figure 1: Illustration of the problem reformulation. The left figure is a document tree rooted at $\mathbf{r}$; it consists of two sentence trees, T_1 and T_2, rooted at r_1 and r_2, respectively. We have $V = \{r_1, r_2, v_1, \ldots, v_6\}$. The right figure shows $\mathcal{P}$, the set of all paths rooted at $\mathbf{r}$. Note that $|V| = |\mathcal{P}|$ holds. With our method, the greedy algorithm is performed over $\mathcal{P}$, which requires at most $O(|V|)$ objective function evaluations in each iteration.

trees form a single document tree. Figure 1 (a) illustrates an example of a document tree.

Given document tree $\mathbf{T}$, a summary preserves the original dependency relations if it forms a subtree rooted at $\mathbf{r}$ in $\mathbf{T}$. Therefore, our aim is to find a rooted subtree of $\mathbf{T}$ that includes informative textual units. For each $v \in V$, the length of v is denoted by $\ell_v \geq 0$; for example, ℓ_v is the number of words or characters in chunk v. If $S \subseteq V$ is a subset of the textual units included in an obtained summary, its total length must be less than or equal to the given length limitation value $L \geq 0$; namely, the following knapsack constraint must be satisfied: $\sum_{v \in S} \ell_v \leq L$. The quality of summary S is evaluated by a monotone submodular function g. Consequently, compressive summarization is formulated as STKP:

$$\begin{aligned} \underset{S \subseteq V}{\text{maximize}} \quad & g(S) \qquad (1) \\ \text{subject to} \quad & \sum_{v \in S} \ell_v \leq L, \\ & S \cup \{\mathbf{r}\} \text{ forms a subtree in } \mathbf{T}. \end{aligned}$$

At first glance, it may seem that the following naive greedy approach works well for this problem: Starting from root $\mathbf{r}$, we sequentially add the most beneficial child to the current solution until the knapsack constraint is violated. Unfortunately, the approximation ratio of this method can become arbitrarily bad since it may miss beneficial vertices that are far from $\mathbf{r}$; if such missed vertices are more beneficial than those added to the solution by a considerable margin, the resulting approximation ratio is almost equal to zero. To avoid this difficulty, we reformulate STKP in the next section.

4 Proposed Method

We observed that the naive greedy algorithm does not work well for STKP (1) due to the complex tree constraint. We circumvent this difficulty by transforming STKP into a special case of the submodular cost submodular knapsack problem (SCSKP). We then provide a greedy algorithm for SCSKP. An approximation guarantee of the greedy algorithm is also presented.

4.1 Problem Reformulation

We show that STKP can be transformed into SCSKP. Let $\mathcal{P}$ be a set of all paths that connect $v \in V$ to $\mathbf{r}$. Note that there is a one-to-one correspondence between $v \in V$ and $p \in \mathcal{P}$ that connects v to $\mathbf{r}$, and hence $|\mathcal{P}| = |V|$. We define $V_p \subseteq V$ as the set of vertices that are included in $p \in \mathcal{P}$, and we let $V_X \coloneqq \bigcup_{p \in X} V_p$ for any $X \subseteq \mathcal{P}$. If $X \subseteq \mathcal{P}$, then $V_X \cup \{\mathbf{r}\}$ forms a subtree in $\mathbf{T}$. Conversely, if $S \cup \{\mathbf{r}\}$ forms a subtree in $\mathbf{T}$ ($S \subseteq V$), there exists $X \subseteq \mathcal{P}$ such that $V_X = S$. Thus STKP (1) can be transformed into the following maximization

Algorithm 1 Greedy

$U \leftarrow \mathcal{P}$, $X \leftarrow \emptyset$
while $U \neq \emptyset$ **do**
  $p = \mathrm{argmax}_{p' \in U} \frac{f(p'|X)}{c(p'|X)}$
  if $c(X + p) \leq L$ **then**
    $X \leftarrow X + p$
  end if
  $U \leftarrow U - p$
end while
$\hat{p} = \mathrm{argmax}_{p' \in \mathcal{P}} f(p')$
return $Y = \mathrm{argmax}_{X' \in \{X, \hat{p}\}} f(X')$

problem on $\mathcal{P}$:

$$\begin{aligned} \underset{X \subseteq \mathcal{P}}{\text{maximize}} \quad & f(X) \coloneqq g(V_X) \\ \text{subject to} \quad & c(X) \coloneqq \sum_{v \in V_X} \ell_v \leq L. \end{aligned} \tag{2}$$

We here suppose that $c(p) \leq L$ holds for all $p \in \mathcal{P}$; any $p \in \mathcal{P}$ violating this condition can be removed in advance since no feasible solution includes such p. The set functions f and c are monotone submodular functions defined on $\mathcal{P}$ (see the Appendix), and thus the above problem is SCSKP. Figure 1 illustrates how to transform STKP into SCSKP.

4.2 Greedy Algorithm

We provide a greedy algorithm for SCSKP (2). In what follows, given any $X, Y \subseteq \mathcal{P}$, we define the binary operators $+$ and $-$ on $\mathcal{P}$ as

$$\begin{aligned} X + Y &\coloneqq \{p \in \mathcal{P} : p \in X \text{ and/or } p \in Y\}, \\ X - Y &\coloneqq \{p \in \mathcal{P} : p \in X \text{ and } p \notin Y\}. \end{aligned}$$

Namely, they are the union and subtraction of two subsets defined on $\mathcal{P}$. We sometimes abuse the notation and regard $p \in \mathcal{P}$ as a subset of $\mathcal{P}$; for example, we let $X + p = X + \{p\}$ for any $X \subseteq \mathcal{P}$ and $p \in \mathcal{P}$. Furthermore, we define $f(X \mid Y) \coloneqq f(X + Y) - f(Y)$ and $c(X \mid Y) \coloneqq c(X + Y) - c(Y)$ for any $X, Y \subseteq \mathcal{P}$.

Algorithm 1 presents a concise description of the greedy algorithm for SCSKP (2). In practice, function evaluations in the above greedy algorithm can be reduced by using the technique provided in (Leskovec et al., 2007) with some modifications. The resulting greedy algorithm requires at most $O(L|V|)$ function evaluations.

Different from the naive greedy algorithm explained in Section 3, the above greedy algorithm is performed on the set of all rooted paths, $\mathcal{P}$. Thus, even if beneficial vertices are far from $\mathbf{r}$, rooted paths that include such beneficial vertices are considered as candidates to be chosen in each iteration. As a result, we get the following performance guarantee for Algorithm 1; we define λ_i as the number of leaves in T_i for $i \in [N]$, and we let $\lambda \coloneqq \max_{i \in [N]} \lambda_i$.

Theorem 1. *If $Y \subseteq \mathcal{P}$ is the output of Algorithm 1 and $X^* \subseteq \mathcal{P}$ is an optimal solution for SCSKP (2), then we have $f(Y) \geq \frac{1}{2}(1 - e^{-1/\lambda}) f(X^*)$.*

Proof. See the Appendix. □

In other words, Algorithm 1 enjoys a $\frac{1}{2}(1 - e^{-1/\lambda})$-approximation guarantee. Notably, if the values of λ_i ($i \in [N]$) are bounded by a small constant for all N sentences, the performance guarantee does not deteriorate no matter how many sentences are in the document data. This implies that our method works effectively for summarizing large-scale document data that comprises many sentences.

4.3 Relation with Existing Work

We first see some existing results. For submodular maximization with a size constraint (i.e., $|S|$ must be at most a certain value), the greedy algorithm has been proved to achieve $(1 - e^{-1})$-approximation (Nemhauser et al., 1978). Khuller et al. (1999) studied the maximum coverage problem with a knapsack constraint, and proved that the greedy algorithm achieves $(1 - e^{-1/2})$-approximation. They also showed that $(1 - e^{-1})$-approximation can be obtained by executing the greedy algorithm $O(|V|^3)$ times, and this result was generalized to the case with a submodular objective function (Sviridenko, 2004). The greedy algorithm for submodular maximization with a knapsack constraint is known to achieve $\frac{1}{2}(1 - e^{-1})$-approximation (Leskovec et al., 2007). Lin and Bilmes (2010) stated that $(1 - e^{-1/2})$-approximation can be obtained with the greedy algorithm, but a mistake in their proof was pointed out by Morita et al. (2013).[2]

Unlike the above problem settings, submodular maximization with a tree constraint has only a few literatures. Krause et al. (2006) studied submodular maximization over a graph with a knapsack and tree constraints, but their algorithm, called *pSPIEL*,

[2]Probably, this mistake can be fixed with the techniques used in (Khuller et al., 1999).

requires a complicated preprocessing step and imposes some assumptions on the problem, which do not hold in most summarization tasks. Iyer and Bilmes (2013) addressed SCSKP, a more general problem setting. Their algorithm is, however, more expensive than the greedy algorithm, and it only achieves a *bi-criterion* approximation guarantee (i.e., not only the objective value but also the magnitude of constraint violation is approximated); if we use this algorithm for document summarization, a resulting summary may violate the length limitation.

We turn to the relation between our result and the existing ones. We consider submodular maximization with a knapsack constraint. This problem can be formulated as an STKP on a *star graph*, whose vertex and edge sets are $\{\mathbf{r}, r_1, \dots, r_N\}$ and $\{(\mathbf{r}, r_1), \dots, (\mathbf{r}, r_N)\}$, respectively (i.e., every leaf corresponds to an element in $V = \{r_1, \dots, r_N\}$). In this case, we have $\lambda = 1$, and thus we obtain a $\frac{1}{2}(1 - e^{-1})$-approximation guarantee, matching the result of (Leskovec et al., 2007).[3]

5 Objective Functions

As presented in (Lin and Bilmes, 2011), many objective functions used for document summarization are known to be monotone and submodular. Below we list examples of the functions that will be used in the experiments.

Coverage Function

To use the coverage function is a simple but powerful approach to document summarization, and so it appears in many existing works (e.g., (Filatova and Hatzivassiloglou, 2004; Takamura and Okumura, 2009a; Berg-Kirkpatrick et al., 2011)). Let M be the number of distinct words in the document data, and suppose that they are indexed with $j \in [M]$. We let w_j ($j \in [M]$) be the weight value of the j-th word. Given summary $S \subseteq V$, the coverage function $\mathrm{COV}(S)$ is defined as follows:

$$\mathrm{COV}(S) := \sum_{j=1}^{M} w_j z_j,$$

where $z_j \in \{0, 1\}$ is a binary decision variable that indicates whether the j-th word is included in S or not; more precisely, $z_j = 1$ if and only if at least one textual unit in S contains the j-th word.

[3] We also tried to obtain an approximation guarantee that corresponds to the $(1 - e^{-1/2})$-approximation (Khuller et al., 1999; Lin and Bilmes, 2010), but it was not straightforward to apply their techniques to our case.

Coverage Function with Rewords

A summary obtained with the above coverage function often consists of many overly-compressed sentences, which typically leads to low readability. Morita et al. (2013) addressed this problem by adding a positive reward term to the coverage function. Given summary S, let $b_{r_i} \in \{0, 1\}$ ($i \in [N]$) be a binary decision variable that indicates whether r_i, the root node of sentence tree T_i, is included in S or not. Note that, if $S \cup \{\mathbf{r}\}$ forms a rooted subtree in $\mathbf{T}$, we have $b_{r_i} = 1$ if and only if at least one textual unit in the i-th sentence appears in S. With these additional variables, the modified coverage function can be written as

$$\mathrm{COVR}(S) := \mathrm{COV}(S) + \gamma \left(\sum_{v \in S} \ell_v - \sum_{i=1}^{N} b_{r_i} \right),$$

where $\gamma \geq 0$ is a parameter that controls the rate of sentence compression. The value of $\sum_{i=1}^{N} b_{r_i}$ is equal to the number of sentences whose textual unit(s) is used in S. Therefore, a summary that consists of fewer sentences tends to get a higher objective value, thus enhancing readability.

ROUGE

ROUGE (Lin, 2004) is widely used for summarization evaluation, and it is known to be highly correlated with human evaluation. Furthermore, ROUGE is known to be monotone and submodular (Lin and Bilmes, 2011). Specifically, given K reference summaries $R_1, \dots, R_K \subseteq V$ and function $\mathrm{C}_e(S)$, which counts the number of times that n-gram e occurs in summary $S \subseteq V$, the ROUGE_n score function is defined as

$$\mathrm{ROUGE}_n(S) := \frac{\sum_{k=1}^{K} \sum_{e \in R_k} \min\{\mathrm{C}_e(S), \mathrm{C}_e(R_k)\}}{\sum_{k=1}^{K} \sum_{e \in R_k} \mathrm{C}_e(R_k)}.$$

6 Experiments

We applied our method to compressive summarization tasks with the three kinds of objective functions: the coverage function, the one with rewards, and ROUGE_1. To benchmark our method, we also applied the ILP-based method to the tasks. These two methods were compared in terms of achieved approximation ratios, ROUGE_1 scores, and running times.

Objective function	Method	Approximation ratio	ROUGE$_1$	Time (ms)
Coverage	Greedy	0.964	**0.347**	**1.34**
	ILP	**1.00**	0.346	231
Coverage with rewards	Greedy	0.967	**0.334**	**1.44**
	ILP	**1.00**	0.332	552
ROUGE$_1$	Greedy	0.985	0.468	**0.759**
	ILP (oracle)	**1.00**	**0.494**	92.1

Table 1: Approximation ratios, ROUGE$_1$ scores, and running times for our method (Greedy) and the ILP-based method (ILP); the average values over the 50 topics are presented. The two methods are applied to compressive summarization tasks with three types of objective functions: Coverage, Coverage with rewards, and ROUGE$_1$. Summaries obtained with the ILP-based method and ROUGE$_1$ objective function are oracle summaries.

6.1 Settings

In the following experiments, we regard V as the set of all chunks in the document data. For each chunk $v \in V$, we let ℓ_v be the number of words contained in v, and we set the length limitation, L, to 100. For the coverage function and the one with rewards, the weight values w_j ($j \in [M]$) were estimated by logistic regression (Yih et al., 2007) trained on the DUC-2003 dataset. For the coverage function with rewards, we set the parameter, γ, to 0.9.

The experiments were conducted on the DUC-2004 dataset for multiple document summarization evaluation, which is a commonly used benchmark dataset. The dataset consists of 50 topics, each of which has 10 newspaper articles. The dependency trees for this dataset were obtained as follows: We first applied the Stanford parser (de Marneffe et al., 2006) to all sentences in the dataset in order to obtain dependency relations between words. We then applied Filippova's rules (Filippova and Strube, 2008; Filippova and Altun, 2013) to the obtained relations so as to construct trees that represent dependency relations between chunks. To obtain summaries with high readability, we treated a set of chunks connected with certain relations (e.g., subject–object) as a single chunk.

Our algorithm was implemented in C++ and compiled with `GCC version 4.8.5`. The ILP-based method solved ILPs with `CPLEX ver. 12.5.1.0`, a widely used commercial ILP solver. The details of ILP formulations for the three objective functions are presented in the Appendix. All experiments were conducted on a Linux machine (CPU: Intel Xeon E5-2620 v4 2.10GHz and 32GB RAM).

6.2 Results

Table 1 summarizes the comparisons of the achieved approximation ratios, ROUGE$_1$ scores and running times. The ILP-based method are always optimal in terms of objective values (i.e., 100%-approximation is attained), and our method achieved more than 95%-approximation. We observed that the maximum number, λ, of leaves in a sentence tree was about 22 on average, which leads to a 2.2%-approximation guarantee of our algorithm. Therefore, our method empirically performs much better than the theoretical guarantee; this is often the case with the greedy algorithm for submodular maximization problems, in particular when the problems have complex constraints. The ROUGE$_1$ scores of our method are comparable to those of the ILP-based method. With the coverage function and the one with rewards, it happened that our method attained slightly higher ROUGE$_1$ scores than those of ILP-based methods;[4] note that this result is possible since the objective values and ROUGE$_1$ scores are not completely correlated. The results on approximation ratios and ROUGE$_1$ scores imply that our method compares favorably with the ILP-based method in terms of empirical performance. With regard to the running times, our method substantially outperformed the ILP-based method. Specifically, our method was about 170, 380, and 120 times faster than the ILP-based one for the coverage function, the one with rewards, and the ROUGE$_1$ objective function, respectively.

Table 2 shows examples of the summaries obtained by our method and the ILP-based method; both methods used the coverage function with rewards as an objective function. We see that

[4] Similar results were observed in (Takamura and Okumura, 2009a).

Greedy: Yeltsin suffered from disease and had a heart attack followed by multiple bypass surgery in the months. Russian President Boris Yeltsin cut short a trip to Central Asia on Monday due to a respiratory infection that revived questions about his health and ability to lead Russia through a sustained economic crisis. Doctors insisted that Yeltsin fly home ahead of schedule. The prime minister reiterated Wednesday that Yeltsin has plans to resign early elections. Russia's Constitutional Court opened hearings Thursday on whether Boris Yeltsin can seek a term. Sources in Primakov's office said the cancellation was due to concerns.
ILP: Russian President Boris Yeltsin cut short a trip to a respiratory infection that revived questions about his health and ability to lead Russia through a economic crisis. Yeltsin was spending outside Moscow his spokesman Dmitry Yakushkin told reporters. Doctors insisted Monday that Yeltsin fly home from Central Asia ahead of schedule because he was suffering. Yeltsin falls ill speculation arises. The prime minister reiterated Wednesday that Yeltsin has plans to resign early elections. Russia's Constitutional Court opened hearings Thursday on whether Boris Yeltsin can seek a term. Sources in Primakov's office said the cancellation was due to concerns.

Table 2: Summaries obtained with our greedy method (upper) and the ILP-based method (lower) for topic:D31032. To obtain these summaries, both methods used the coverage function with rewards as an objective function.

both methods successfully created informative summaries that preserve original dependency relations. The readability of obtained summaries is unfortunately not high enough. Note that not only our method but also most compressive summarization methods suffer this problem; in fact, there is little difference between the two summaries obtained with our method and the optimal ILP-based method with regard to readability. To conclude, the empirical performance of our method matches that of the ILP-based method, while running about 100 to 400 times faster.

7 Conclusion and Discussion

We proposed a fast greedy method for compressive summarization. Our method works with any monotone submodular objective function; examples of such functions include the coverage function, ROUGE, and many others. The $\frac{1}{2}(1 - e^{-1/\lambda})$-approximation guarantee of our method was proved, which generalizes the $\frac{1}{2}(1 - e^{-1})$-approximation for submodular maximization with a knapsack constraint. Experiments showed that our greedy method empirically achieves more than 95%-approximation and that it runs about 100 to 400 times faster than the ILP-based method implemented with `CPLEX`. With the coverage function and its variant, our method attained as high ROUGE_1 scores as the ILP-based method.

As mentioned above, current compressive summarization systems often fail to achieve high readability, and one possible approach to this problem is to develop better objective functions. Since our method is applicable to various monotone submodular objective functions and can find almost optimal solutions efficiently, our method would be helpful in testing the performance of newly proposed objective functions. Thus we believe that our method is useful for advancing the study into compressive summarization.

Interestingly, STKP can be seen as a variant of *DR-submodular* maximization (Soma and Yoshida, 2017), which is a submodular maximization problem defined over integer lattice. The constraint that appears in DR-submodular maximization is somewhat easier to deal with than that of our problem; exploiting this, Soma and Yoshida (2017) developed a polynomial-time algorithm that achieves roughly $\frac{1}{2}$-approximation. The techniques studied in this field may be useful to develop better algorithms for STKP, which we leave for future work.

References

Miguel Almeida and Andre Martins. 2013. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 196–206. http://www.aclweb.org/anthology/P13-1020.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 481–490. http://www.aclweb.org/anthology/P11-1049.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 484–494. https://doi.org/10.18653/v1/P16-1046.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*. European Language Resources Association, pages 449–454. http://www.aclweb.org/anthology/L06-1260.

Harold P. Edmundson. 1969. New methods in automatic extracting. *Journal of the ACM* 16(2):264–285. https://doi.org/10.1145/321510.321519.

Elena Filatova and Vasileios Hatzivassiloglou. 2004. A formal model for information selection in multi-sentence text extraction. In *Proceedings of the 20th International Conference on Computational Linguistics*. http://www.aclweb.org/anthology/C04-1057.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1481–1491. http://www.aclweb.org/anthology/D13-1155.

Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *Proceedings of the 5th International Natural Language Generation Conference*. Association for Computational Linguistics, pages 25–32. http://www.aclweb.org/anthology/W08-1105.

Tsutomu Hirao, Masaaki Nishino, and Masaaki Nagata. 2017a. Oracle summaries of compressive summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, pages 275–280. https://doi.org/10.18653/v1/P17-2043.

Tsutomu Hirao, Masaaki Nishino, Jun Suzuki, and Masaaki Nagata. 2017b. Enumeration of extractive oracle summaries. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, pages 386–396. http://www.aclweb.org/anthology/E17-1037.

Rishabh Iyer and Jeff Bilmes. 2013. Submodular optimization with submodular cover and submodular knapsack constraints. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. Curran Associates Inc., pages 2436–2444. http://dl.acm.org/citation.cfm?id=2999792.2999884.

Litton J Kurisinkel, Pruthwik Mishra, Vigneshwaran Muralidaran, Vasudeva Varma, and Dipti Misra Sharma. 2016. Non-decreasing sub-modular function for comprehensible summarization. In *Proceedings of the NAACL Student Research Workshop*. Association for Computational Linguistics, pages 94–101. https://doi.org/10.18653/v1/N16-2014.

Samir Khuller, Anna Moss, and Joseph S. Naor. 1999. The budgeted maximum coverage problem. *Information Processing Letters* 70(1):39–45. https://doi.org/10.1016/S0020-0190(99)00031-9.

Yuta Kikuchi, Tsutomu Hirao, Hiroya Takamura, Manabu Okumura, and Masaaki Nagata. 2014. Single document summarization based on nested tree structure. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, pages 315–320. https://doi.org/10.3115/v1/P14-2052.

Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon Kleinberg. 2006. Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*. ACM, pages 2–10. https://doi.org/10.1145/1127777.1127782.

Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. 2007. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pages 420–429. https://doi.org/10.1145/1281192.1281239.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Proceedings of Workshop on Text Summarization Branches Out*. pages 74–81. http://www.aclweb.org/anthology/W04-1013.

Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 912–920. http://www.aclweb.org/anthology/N10-1134.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Lin-*

guistics: Human Language Technologies. Association for Computational Linguistics, pages 510–520. http://www.aclweb.org/anthology/P11-1052.

Hans P. Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development* 2(2):159–165. https://doi.org/10.1147/rd.22.0159.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European Conference on IR Research*. Springer-Verlag, pages 557–564. http://dl.acm.org/citation.cfm?id=1763653.1763720.

Hajime Morita, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2013. Subtree extractive summarization via submodular maximization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 1023–1032. http://www.aclweb.org/anthology/P13-1101.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming* 14(1):265–294. https://doi.org/10.1007/BF01588971.

Maxime Peyrard and Judith Eckle-Kohler. 2017. Supervised learning of automatic pyramid for optimization-based multi-document summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 1084–1094. https://doi.org/10.18653/v1/P17-1100.

Tasuku Soma and Yuichi Yoshida. 2017. Non-monotone DR-submodular function maximization. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. pages 898–904. https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14483.

Maxim Sviridenko. 2004. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters* 32(1):41–43. https://doi.org/10.1016/S0167-6377(03)00062-2.

Hiroya Takamura and Manabu Okumura. 2009a. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the ACL*. Association for Computational Linguistics, pages 781–789. http://www.aclweb.org/anthology/E09-1089.

Hiroya Takamura and Manabu Okumura. 2009b. Text summarization model based on the budgeted median problem. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*. ACM, pages 1589–1592. https://doi.org/10.1145/1645953.1646179.

Wen-tau Yih, Joshua Goodman, Lucy Vanderwende, and Hisami Suzuki. 2007. Multi-document summarization by maximizing informative content-words. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann Publishers Inc., pages 1776–1782. http://dl.acm.org/citation.cfm?id=1625275.1625563.