NAIST-IS-DD1261010

Doctoral Dissertation

Automatic Summarization on Various Domains with Combinatorial Optimization and Machine Learning

Hitoshi Nishikawa

August 12, 2013

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Hitoshi Nishikawa

Thesis Committee:

Professor Yuji Matsumoto	(Supervisor)
Professor Satoshi Nakamura	(Co-supervisor)
Associate Professor Masashi Shimbo	(Co-supervisor)
Assistant Professor Kevin Duh	(Co-supervisor)

Automatic Summarization on Various Domains with Combinatorial Optimization and Machine Learning*

Hitoshi Nishikawa

Abstract

The kind and number of texts which can be targets of automatic summarization are increasing. This is because the number of computerized texts is increasing due to the growth of the use of the Internet. It has caused novel problems in a field of automatic summarization. There are three major problems: readability, diversity of content, and speed. The aim of this study is to propose novel methods of automatic summarization and to address these problems.

The main contribution of this study is to present a solution for each problem. First, we propose a novel automatic summarization model which extracts and orders sentences simultaneously to the problem about readability, and show a result that the readability of a summary is improved by formulating and solving the model as an instance of integer linear programming. Second, we propose a transfer learning method for the problem about diversity of content and show that the content of a summary is improved by leveraging the data from different domains through transfer learning. Finally, we propose a novel summarization model, the redundancy-constrained knapsack problem, and a novel decoding method based on the Lagrange heuristic, and show that the proposed model and decoding method can generate a good summary far faster than a state-of-the-art summarization model.

In this dissertation, we first clarify the goal and significance of this study. Then, as preliminaries, we overview integer linear programming and structured learning. Since all tasks of automatic summarization introduced in this dissertation are described as an instance of combinatorial optimization problems, we briefly explain integer linear programming as a powerful way to model and solve the combinatorial optimization

*Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD1261010, August 12, 2013.

problems. Additionally, we touch on structured learning as a way to estimate the parameters in tasks of automatic summarization. Also, we survey studies of automatic summarization and clarify the position of this study in the field. After these preliminaries, we elaborate the aforementioned three problems. First, we deal with a set of evaluative texts such as reviews as a target of automatic summarization and show that the proposed method can improve the readability of a summary. Second, we address a task to summarize contact center dialogues and show that the content of a summary can be improved by the proposed method. Finally, we show that the speed of summarizing multiple documents is accelerated by our proposals. At the end of this dissertation, we show future direction based on these results.

Keywords:

Natural Language Processing, Automatic Summarization, Combinatorial Optimization, Integer Linear Programming, Structured Learning

組み合わせ最適化と機械学習による多様な分野に対する自動要約*

西川 仁

内容梗概

自動要約の対象となるテキストの種類，量が増加しつつある．これはインターネットの普及に伴い電子化されたテキストが増加したためであり，テキストの多様性が増すにつれて自動要約において新しい課題が生じている．課題は大きく分けて3つ存在する．第一に可読性の問題であり，第二に内容性の問題であり，最後に要約速度の問題である．本研究の目的は，自動要約において新しく生じたこれらの課題に対して，新しい自動要約の方法を提案し，これを解決することである．

本研究の主たる貢献は，これらの問題それぞれに対して解決の方法を示した点にある．まず，可読性の問題に対しては，文の抽出と順序付けを同時に行う要約モデルを新しく提案し，これを整数計画問題として表現し解くことによって，文の抽出と順序付けを別個に行う既存の手法に比べて可読性が改善された要約を生成できることを示す．次に，内容性の問題に対しては，転移学習を利用し異なる分野のテキストを教師事例として用いることで内容性が改善された要約を生成できることを示す．最後に，要約速度の問題に対しては，新しい要約モデルである冗長性制約付きナップサックモデルと，ラグランジュヒューリスティックに基づく高速なデコーディングアルゴリズムを提案し，良好な要約を高速に生成できることを示す．

本論文ではまず，本研究の目的，意義を明らかにする．次に，準備として，整数計画と構造学習について概観する．本論文中で扱われる自動要約の課題はすべて組み合わせ最適化問題の一種として表現されるため，組み合わせ最適化問題を表現しこれを解くための強力な枠組みである整数計画について簡単に説明する．また，自動要約の課題に含まれるパラメータを推定するための方法として，構造学習についても説明する．そののち，自動要約研究について広く概観し，本研究の位置づけを明らかにする．これらの準備ののち，上述した3つの問題について

*奈良先端科学技術大学院大学 情報科学研究科 情報処理学専攻 博士論文, NAIST-IS-DD1261010, 2013年8月12日.

詳述する．第一の可読性に関する問題では，評価文書集合を要約の対象として取り上げ，提案する方法により要約の可読性が改善することを示す．第二の内容性に関する問題では，コンタクトセンタログを要約の対象として取り上げ，提案する方法により要約の内容性が改善することを示す．第三の要約速度に関する問題では，新聞記事集合と評価文書集合を要約の対象として取り上げ，提案する方法により要約速度が改善することを示す．最後に，これらの結果を踏まえ，更なる研究の方向性を示す．

キーワード

自然言語処理, 自動要約, 組み合わせ最適化, 整数計画問題, 構造学習

Acknowledgments

主指導教官の松本裕治教授に感謝いたします．松本先生からは，研究においてだけでなく，人間的にも大きな影響を受けました．ありがとうございました．

お忙しい中，副指導教官になっていただいきました中村哲教授，新保仁准教授，Kevin Duh 助教に感謝いたします．先生方からは本論文に関し様々なご助言を賜りました．ありがとうございました．現在，名古屋大学におられる関浩之教授は中間発表の際に様々なご助言をくださりました．ありがとうございました．また秘書の北川祐子さんは遠隔地で研究活動を進める私に様々なご支援をくださいました．ありがとうございました．

学部および修士課程では慶應義塾大学の石崎俊教授にご指導をいただき，現在，湘南工科大学の内山清子准教授，嘉悦大学の岡本潤専任講師，現在の同僚でもある NTT メディアインテリジェンス研究所の東中竜一郎主任研究員からは公私ともに様々なご支援を頂戴しました．ありがとうございました．

本研究は日本電信電話株式会社にて行われたものです．研究の推進においては社内の様々な方のご支援を頂戴しました．特に，私が現在の職場の一員となったときのグループリーダであり，現在は岡山県立大学におられる菊井玄一郎教授をはじめ，松尾義博主幹研究員，牧野俊朗主幹研究員，今村賢治主任研究員，長谷川隆明主任研究員，齋藤邦子主任研究員，平尾努研究主任，平野徹研究員，小林のぞみ研究員からは多大なご支援を賜りました．菊井さんには，私が修士の際に現在の職場に実習生として来たときに初めてお会いしましたが，菊井さんから得た強い印象が卒業後も自然言語処理の研究を続ける動機となりました．大学に移られた後も事あるごとに有益な助言をくださいました．ありがとうございました．長谷川さんは，右も左もわからない私を粘り強く研究者として教育してくださいました．長谷川さんのご指導がなければ本論文は書かれなかったものと思います．ありがとうございました．平尾さんからは，西川がまさに自動要約の研究を始めようとしているとき，様々なご助言を頂戴しました．京都まで平尾さんのご助言を頂戴しに伺ったときのことを，昨日のことのように覚えています．あの時の厳しい，それでいて暖かい励ましがここまで自動要約の研究を続ける原動力になりました．ありがとうございました．

泉朋子研究員，丹羽健太研究員，山室健研究員，数原良彦研究員，現在， NTT コミュニケーションズ株式会社の福冨隆朗氏をはじめとする，日本電信電話株式会社の同期の皆様には公私ともに大変お世話になりました．ありがとうございました．優秀な同期から受けた多くの刺激がなければ本論文が完成することはなかったと思います．

自動要約の研究に携わり，幸いなことに，自然言語処理の分野でご活躍されている数多くの研究者の皆様から研究についてご助言を頂戴することができました．特に，東京工業大学の高村大也准教授，飯田龍助教，笹野遼平助教，はこだて未来大学の藤田篤准教授，首都大学東京の小町守准教授，東北大学の渡邉陽太郎助教，国立情報学研究所の宮尾祐介准教授，横野光研究員，株式会社プリファードインフラストラクチャーの海野裕也氏，株式会社ミクシィの木村俊也氏， LINE 株式会社の佐藤敏紀氏からは学会や研究会で示唆に富んだ様々なご助言を頂戴しました．ありがとうございました．

父・昭男と母・緑は常に私に機会を与えることを惜しまず，その結果の一つが本論文として結実しました．父と母の献身の偉大さに，家庭を持った今，あらためて圧倒されています．感謝します．妹・愛は，私が重大な選択に直面した際にいつも示唆に富んだ助言をくれました．感謝します．妻・伸枝は，自身多忙にもかかわらず，本論文を執筆する私を常に気遣ってくれました．妻の暖かい気遣いは本論文の執筆にあたって心の支えとなりました．最大の感謝を捧げます．

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

In recent years, with the growth of the use of the Internet, there are a huge number of computerized documents. By utilizing information contained in these documents, individuals and organizations can achieve a better decision. For example, individual consumers can consult reviews about some sort of product written by other consumers as to whether they should buy it. Companies can leverage reviews about their products written by the consumers to improve the products and marketing strategy. However, it is almost impossible to read all documents given their sheer number.

Automatic summarization is a technique to summarize a set of computerized documents so as to alleviate such information overload. The use of automatic summarization is expected to allow all possible documents to be efficiently utilized. However, there are documents which have newly become targets of automatic summarization. The property of those is difficult to be treated with previous approaches.

1.2 Research Objectives and Tasks

There are three major problems to be solved for summarizing the documents which have newly arisen: readability, diversity of content, and speed.

Readability A dominant approach to automatic summarization is extractive summarization [50, 88]. It extracts a set of sentences in input documents, and then orders

extracted sentences appropriately. Finally, it outputs the extracted sentences as a summary.

Since a summary is generated by reproducing expressions in the original input documents, the quality of an output summary largely depends on the original expressions. The documents as targets in the previous studies of automatic summarization such as newswire articles and scientific literature, are written by experts such as journalists and researchers. Basically, these texts are well-written and readable, and hence a summary generated from those can reap their original quality.

In contrast to newswire articles and scientific literature, other documents which have newly arisen on the Internet due to the recent growth of the use of the Internet, such as reviews, are written mostly by people who aren ’t experts of writing documents, and hence output summaries suffer from the lack of readability. Therefore improving the readability of a summary is needed.

To address this problem, we propose a novel summarization model that extracts and orders sentences simultaneously. We will show that the readability of a summary can be improved by our proposal.

Diversity of Content The second problem is diversity of content. In contrast to the documents previously assumed as targets of automatic summarization such as newswire articles and scientific literature, the documents which have recently arisen on the Internet such as reviews and tweets have a great variety in their contents.

This causes an increase in the cost to prepare training data of summarization. Most of recent approaches in automatic summarization leverage the training data to estimate their parameters required to summarize the input documents. An increase in the diversity of input documents forces the training data to cover wider aspects assumed to occur in input documents in practical situations, and hence plenty of training data must be prepared to summarize these documents.

To address this problem, we leverage a transfer learning technique to estimate parameters. By the use of transfer learning, we can leverage the training data in domains other than the target domain, and hence we can estimate the required parameters from the relatively small portion of training data.

Speed The last problem is the time required to summarize input documents. The number of documents which have newly arisen as targets of automatic summarization is far larger than the number of documents previously targeted, such as newswire ar-

ticles. Most of previously proposed and widely-used summarization algorithms are computationally complex, and hence the time required to summarize the input documents is increasing with an increase in the number of those.

To address this problem, we propose a novel summarization model for multi-document summarization, the redundancy-constrained knapsack model, and a novel decoding method for that model.

1.3 Thesis Overview

The rest of this dissertation is organized as follows. As preliminaries, we will provide a brief introduction of integer linear programming and structured learning in Chapter 2. These techniques will be leveraged in following chapters. Then, we overview the basic techniques of automatic summarization proposed in previous studies in Chapter 3. In Chapter 4, we present a joint model for sentence extraction and ordering as a solution for the deterioration of readability. In Chapter 5, we present a transfer learning method to address the diversity of content. In Chapter 6, we present a fast decoding method with the Lagrange heuristic for multi-document summarization. We summarize the dissertation and set forth further directions in Chapter 7.

Chapter 2

Preliminaries

In this chapter, as preliminaries to the following chapters, we briefly explain two basic techniques: integer linear programming and structured learning. Integer linear programming, hereinafter abbreviated as ILP, are used to express models of summarization introduced in the following chapters and to leverage generic algorithms, such as branch-and-bound method, to solve the problems. Structured learning is used to estimate the parameters of the models.

2.1 Integer Linear Programming

ILP is a mathematical method to locate the optimal solution of a given mathematical model. Since ILP is based on linear programming, in this section, we firstly describe linear programming, hereinafter abbreviated as LP, then describe ILP. After explaining ILP, we explain branch-and-bound method, a basic technique to solve ILP.

2.1.1 Linear Programming

Linear programming is a way to locate the optimal solution in a given linear program. Linear program is a problem that can be expressed in the following form [41]:

$$\begin{aligned} \text{maximize} \quad & \mathbf{c}^\top \mathbf{x}, \\ \text{subject to} \quad & A\mathbf{x} \leq \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned} \tag{2.1}$$

where $\mathbf{x}$ indicates the vector of variables, $\mathbf{c}$ and $\mathbf{b}$ are vectors of coefficients, A is a matrix of coefficients, and $(\cdot)^\top$ is the matrix transpose. Vector $\mathbf{c}$ and Vector $\mathbf{b}$, and Matrix A are given; Vector $\mathbf{x}$ consists of variables to be determined. Formula $\mathbf{c}^\top\mathbf{x}$ is called the objective function. The objective function is linear, and hence the program is called linear program (i.e., if the objective function is quadratic, it is called quadratic program). The objective function is the dot product of the vector of variables to be determined, $\mathbf{x}$, and coefficients, $\mathbf{c}$. Hence the value of the vector $\mathbf{c}$ affects the value of the vector $\mathbf{x}$ strongly. For example, if the i -th element of the vector $\mathbf{c}$, c_i, has a large value, the i -th element of the vector $\mathbf{x}$, x_i, should also have a large value to maximize the objective function. The inequalities $A\mathbf{x} \leq \mathbf{b}$ defines the space of solutions. These inequalities specify a convex polytope over which the objective function is to be optimized.

To solve LP, the simplex method and interior-point method are widely used [41].

2.1.2 Integer Linear Programming

ILP is based on linear programming, though, the range of solutions of ILP is restricted to integer. That is, ILP is an instance of LP with an integer constraint and is described as follows:

$$\begin{aligned} \text{maximize} \quad & \mathbf{c}^\top\mathbf{x}, \\ \text{subject to} \quad & A\mathbf{x} \leq \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \\ & \mathbf{x} \in \{\mathbb{Z}\}^n, \end{aligned} \tag{2.2}$$

where n is the dimension of the vector $\mathbf{x}$, and $\mathbb{Z}$ means integer, and hence the elements of the vector $\mathbf{x}$ is restricted to integer. While in LP the space of solution is over Euclidean space $\mathbb{R}^n$, in ILP the space of solution is restricted to be over the integer lattice in the integer hull of the original space of LP.

Since the space of solution is restricted in the integer lattice, simplex method and interior-point method cannot locate the optimal solution efficiently. To solve ILP, branch-and-bound method is widely used.

2.1.3 Branch-and-Bound Method

The branch-and-bound method is a generic algorithm to solve combinatorial optimization problems. The method consists of two operations: branch, which is to build a new hypothesis in a search tree, and bound, which is to trim the hypothesis that has no hope to be the optimal solution. By excluding the hypotheses that never be the optimal solution, the method can efficiently locate the optimum solution. To exclude such hypotheses, the method must identify a lower bound and upper bound. A lower bound is the value that the value of the optimal solution is either more than or equal to. An upper bound is the value that the value of the optimal solution is either less than or equal to. If the value of the upper bound of hypotheses expanded from a hypothesis is less than the value of the lower bound, the hypotheses never be the optimal solution, and hence they not have to be checked.

The lower bound is calculated by some heuristics, such as the greedy method. Throughout the search process in the branch-and-bound method, the lower bound is calculated many times, and hence the lower bound have to be calculated quickly. Since the greedy method can locate the approximate solution quickly, it is widely used to locate the lower bound [41].

The upper bound can be calculated by relaxing the integer constraint of ILP. The value of the optimal solution of an instance of LP is larger than the value of the optimal solution of an instance of LP with the integer constraint (i.e., ILP). Hence, the upper bound can easily be calculated by solving LP made by relaxing the integer constraint using simplex method or interior-point method. The simplex method and interior-point method are fast enough to be used in the branch-and-bound method.

By identifying the range in which the value of the optimal solution lies, the branch-and-bound method can efficiently locate the optimal solution.

2.1.4 Integer Linear Programming in Natural Language Processing

In most of natural language processing tasks, the space of solutions is discrete, and hence ILP can be used to solve the tasks.

Germann et al. [26] formulated a decoding task of IBM Model 4 [12] as an instance of ILP. The speed of decoding IBM Model 4 with ILP is slower than other heuristics, such as the stack decoder and greedy algorithm, but ILP solver can output the optimal

solution. They compared the optimal solution from an ILP solver with other heuristics to analysis search error.

Roth and Yih [70] performed named entity recognition and relation extraction between named entities using ILP. Results of named entity recognition and relation extraction have a dependency relationship and ILP can model this dependency. They showed that the joint inference of named entity recognition and relation extraction with ILP improved the accuracy of both named entity recognition and relation extraction.

Barzilay and Lapata [8] formulated an aggregation task in natural language generation as an instance of ILP. The aggregation task is to divide a set of linguistic structures into subsets. Each subset is to be generated as a single sentence. The task can be formulated as an instance of the set partitioning problem and the problem requires global inference, and hence Barzilay and Lapata leveraged ILP to solve the problem.

As shown above, ILP offers a way to solve complex problems that have to take into account relations among many variables.

2.2 Structured Learning

In this section, we briefly explain about structured learning. The goal of learning is to induce a mapping,

$$F : \mathscr{X} \mapsto \mathscr{Y}, \tag{2.3}$$

where $\mathscr{X}$ is a set of input and $\mathscr{Y}$ is a set of output. In the case of structured learning, each element of $\mathscr{Y}$ has some sort of structure such as sequence, tree, and graph. To infer the structure from the input, the problem of inference is a task that finding the best solution on function $f : (\mathbf{x}, \mathbf{y}) \mapsto \mathbb{R}$, where $\mathbf{x} \in \mathscr{X}$ is an instance of input, $\mathbf{y} \in \mathscr{Y}$ is an instance of the output, and $\mathbb{R}$ is a real number. The inference is formulated as

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \mathscr{Y}}{\operatorname{argmax}} f(\mathbf{x}, \mathbf{y}). \tag{2.4}$$

A linear model is assumed here, which is $f(\mathbf{x}, \mathbf{y}) = \mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y})$, where $\mathbf{w}$ is a parameter vector, $\phi(\mathbf{x}, \mathbf{y})$ is a feature vector, and $\mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y})$ is a dot product of $\mathbf{w}$ and $\phi(\mathbf{x}, \mathbf{y})$. Hence, learning is to estimate parameter vector $\mathbf{w}$ given feature function ϕ.

In this section, two kinds of ways to estimate parameter vector $\mathbf{w}$ are introduced. In Section 2.2.1, structured perceptron [17] is explained. Then, in section 2.2.2, passive-aggressive algorithm [18], which is a refinement of structured perceptron, is explained.

Passive-aggressive algorithm is used to estimate a parameter vector in the later chapters: Chapter 4 and Chapter 5.

2.2.1 Structured Perceptron

Structured perceptron is proposed by Collins [17], estimating the parameters of hidden Markov models. Since perceptron is originally created as a classifier [69], Collins extended it to handle outputs with structure, i.e., sequence of labels.

Given training samples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1\ldots n}$, the learning of structured perceptron updates parameter vector $\mathbf{w}$ iteratively. The algorithm of structured perceptron is shown in Algorithm 1.

Algorithm 1 Structured Perceptron

1: $\mathbf{w} \leftarrow \mathbf{0}$
2: **for** $t = 1$ to T **do**
3: $\quad i = rand(1, n)$
4: $\quad \hat{\mathbf{y}} = argmax_{\mathbf{y}} \mathbf{w}^\top \phi(\mathbf{x}_i, \mathbf{y})$
5: $\quad$ **if** $\hat{\mathbf{y}} \neq \mathbf{y}_i$ **then**
6: $\quad\quad \mathbf{w} = \mathbf{w} + \phi(\mathbf{x}_i, \mathbf{y}_i) - \phi(\mathbf{x}_i, \hat{\mathbf{y}})$
7: $\quad$ **end if**
8: **end for**
9: **return** $\mathbf{w}$

First of all, parameter vector $\mathbf{w}$ is initialized (Line 1). Updating the parameter is performed T times (Line 2). Function $rand(1, n)$ returns an integer from 1 to n and the algorithm chooses one training samples from training examples (Line 3). An inference is performed given input $\mathbf{x}$ and parameter vector $\mathbf{w}$ (Line 4). If the output of the inference, $\hat{\mathbf{y}}$, is not correct, i.e., not the same as the correct output, $\mathbf{y}$ (Line 5), parameter vector $\mathbf{w}$ is updated (Line 6). After the iterations, the algorithm outputs parameter vector $\mathbf{w}$ (Line 9).

2.2.2 Passive-Aggressive Algorithm

In contrast to structured perceptron that has no explicit loss function, passive-aggressive algorithm [18] draw on a loss function to estimate the parameter.

Passive-aggressive algorithm updates the parameter vector with following formula

$$\begin{aligned} \mathbf{w}^{new} &= \underset{\mathbf{w}}{\operatorname{argmin}} \tfrac{1}{2}||\mathbf{w}-\mathbf{w}^{old}||^2, \\ \text{s.t. } & \mathbf{w}^\top \phi(\mathbf{x},\mathbf{y}) - \mathbf{w}^\top \phi(\mathbf{x},\hat{\mathbf{y}}) \geq \ell(\hat{\mathbf{y}};\mathbf{y}), \end{aligned} \tag{2.5}$$

where $\ell(\hat{\mathbf{y}};\mathbf{y})$ is a loss function measuring how wrong $\hat{\mathbf{y}}$ against correct output $\mathbf{y}$. Unlike the case of structured perceptron, which merely see whether output $\hat{\mathbf{y}}$ is the same as correct output $\mathbf{y}$, passive-aggressive algorithm takes into account how different these outputs are. To derive the update formula from Equation 2.5, the Lagrange function is introduced:

$$\mathscr{L}(\mathbf{w},\lambda) = \frac{1}{2}||\mathbf{w}-\mathbf{w}^{old}||^2 + \lambda(\mathbf{w}^\top \phi(\mathbf{x},\hat{\mathbf{y}}) - \mathbf{w}^\top \phi(\mathbf{x},\mathbf{y}) + \ell(\hat{\mathbf{y}};\mathbf{y})), \tag{2.6}$$

where $\lambda \geq 0$ is the Lagrange Multiplier. By partially differentiating the above equation with respect to $\mathbf{w}$, following gradient are obtained:

$$\nabla_{\mathbf{w}}\mathscr{L}(\mathbf{w},\lambda) = \mathbf{w}-\mathbf{w}^{old} + \lambda(\phi(\mathbf{x},\hat{\mathbf{y}}) - \phi(\mathbf{x},\mathbf{y})). \tag{2.7}$$

Since $\nabla_{\mathbf{w}}\mathscr{L}(\mathbf{w},\lambda)$ must be 0, by substituting the above $\mathbf{w}$ into Equation 2.6 and partially differentiating it with respect to λ, following gradient is obtained:

$$\begin{aligned} \nabla_\lambda \mathscr{L}(\lambda) = \quad & - \quad \lambda(\phi(\mathbf{x},\hat{\mathbf{y}}) - \phi(\mathbf{x},\mathbf{y}))^2 \\ & + \quad \mathbf{w}^{old}(\phi(\mathbf{x},\hat{\mathbf{y}}) - \phi(\mathbf{x},\mathbf{y})) \\ & + \quad \ell(\hat{\mathbf{y}};\mathbf{y})). \end{aligned} \tag{2.8}$$

$\nabla_\lambda \mathscr{L}(\lambda)$ also must be 0. By Equation 2.7 and 2.8, finally following update formula is obtained:

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \lambda(\phi(\mathbf{x},\mathbf{y}) - \phi(\mathbf{x},\hat{\mathbf{y}})), \tag{2.9}$$

$$\lambda = \frac{\ell(\hat{\mathbf{y}};\mathbf{y}) - \mathbf{w}^{old}\cdot\phi(\mathbf{x},\mathbf{y}) + \mathbf{w}^{old}\cdot\phi(\mathbf{x},\hat{\mathbf{y}})}{||\phi(\mathbf{x},\mathbf{y}) - \phi(\mathbf{x},\hat{\mathbf{y}})||^2}. \tag{2.10}$$

That is, Equation 2.13 means that the bigger the loss $\ell(\hat{\mathbf{y}};\mathbf{y})$, the greater the updates.

Algorithm 2 Passive-Aggressive Algorithm

1: $\mathbf{w} \leftarrow \mathbf{0}$
2: **for** $t = 1$ to T **do**
3: $\quad i = rand(1, n)$
4: $\quad \hat{\mathbf{y}} = argmax_{\mathbf{y}} \mathbf{w}^\top \phi(\mathbf{x}_i, \mathbf{y})$
5: $\quad \mathbf{w} = \mathbf{w} + \lambda(\phi(\mathbf{x}_i, \mathbf{y}_i) - \phi(\mathbf{x}_i, \hat{\mathbf{y}}))$
6: **end for**
7: **return** $\mathbf{w}$

Given training samples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1...n}$, as the learning of structured perceptron, the learning of passive-aggressive algorithm updates parameter vector $\mathbf{w}$ iteratively. The algorithm of passive-aggressive algorithm is shown in Algorithm 2.

First of all, parameter vector $\mathbf{w}$ is initialized (Line 1). Updating the parameter is performed T times (Line 2). Function $rand(1, n)$ returns an integer from 1 to n and the algorithm chooses one training samples from training examples (Line 3). An inference is performed given input $\mathbf{x}$ and parameter vector $\mathbf{w}$ (Line 4). The algorithm calculates the loss between output $\mathbf{y}$ given $\mathbf{w}$ and correct output $\hat{\mathbf{y}}$ and λ, then, parameter vector $\mathbf{w}$ is updated (Line 6). After the iterations, the algorithm outputs parameter vector $\mathbf{w}$ (Line 7).

2.2.3 Passive-Aggressive Algorithm with Slack Variable

The above update equation, Equation 2.10, is sensitive to noisy examples [18]. To moderate the update, slack variable ξ is introduced to the equation. Although Crammer et al. introduced two types of an update equation, this chapter shows one of these equations, PA-II, which is used in the following sections.

In PA-II, an update is done by solving the following optimization problem:

$$\begin{aligned} \mathbf{w}^{new} &= \underset{\mathbf{w}}{\operatorname{argmin}} \tfrac{1}{2}||\mathbf{w} - \mathbf{w}^{old}||^2 + C\xi^2, \\ \text{s.t. } & \mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y}) - \mathbf{w}^\top \phi(\mathbf{x}, \hat{\mathbf{y}}) + \xi \geq \ell(\hat{\mathbf{y}}; \mathbf{y}), \end{aligned} \tag{2.11}$$

where C is a hyper parameter named aggressiveness parameter, which controls the effect of the hyper parameter on the update. In this case, the update formula is

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \lambda(\phi(\mathbf{x},\mathbf{y}) - \phi(\mathbf{x},\hat{\mathbf{y}})), \tag{2.12}$$

$$\lambda = \frac{\ell(\hat{\mathbf{y}};\mathbf{y}) - \mathbf{w}^{old}\cdot\phi(\mathbf{x},\mathbf{y}) + \mathbf{w}^{old}\cdot\phi(\mathbf{x},\hat{\mathbf{y}})}{||\phi(\mathbf{x},\mathbf{y}) - \phi(\mathbf{x},\hat{\mathbf{y}})||^2 + \frac{1}{2C}}. \tag{2.13}$$

The derivation is basically the same as the case without the slack variable shown above.

Chapter 3

Automatic Summarization

This chapter surveys previous studies in automatic summarization for following chapters.

In Section 3.1, we overview the study of automatic summarization. Then, in Section 3.1, we describe the methods previously studied. This section consists of two parts: sentence extraction and sentence ordering. Each of them is a central concern in this dissertation. Finally, we explain evaluation methods for automatic summarization in Section 3.3.

3.1 Overview

In this section, we overview studies of automatic summarization. First, we confirm the purpose of automatic summarization. Then, we explain some important aspects to understand the focus of this dissertation. Finally, we show kinds of documents previously targeted in the previous work of automatic summarization.

Purpose of automatic summarization

With the growth of the use of the Internet and increase of the computerized documents, we can access more and more information through the Internet. To understand important information in a short time, we clearly need a way to sort out the important information from a whole document. Automatic summarization offers a way to abridge a complex, long document to help readers understand the important information in a short time.

Mani and Maybury defined the summarization as "Text summarization is the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks)" [51]. As they pointed out, the form of summary takes different forms depending on its context. Next, we will get some aspects in automatic summarization sorted out.

Taxonomy of automatic summarization

A study of automatic summarization can be classified according to some aspects.

Single Document or Multi-Document The first grouping is the number of documents to be target of automatic summarization [50, 88, 58]. In the study of automatic summarization can be divided into two fields: single document summarization and multi-document summarization. The former targets a single document as an input of automatic summarization. For example, in single document summarization, a summarizer receives a newswire article and produce a short summary of that article. In contrast to single document summarization, the latter targets multiple documents as an input of automatic summarization. For example, in multi-document summarization, a summarizer receives a bunch of newswire articles and produce a single summary from those. Basically, each document consisting of an input document set has a common topic. For example, each document relates to a certain terrorist attack, even if each document is written by a different author.

In this dissertation, we will treat both single document and multi-document settings. In Chapter 4 and 6, we consider problems in multi-document summarization. In Chapter 5, we consider a problem in single document summarization.

Indicative or Informative A summary can be divided into two types according to its purpose of use: an indicative summary and informative summary [88]. The former is used to judge whether readers should read the body of original document. For example, a headline for a newswire article is an example of indicative summary; readers decide whether they should read the body by taking into account the headline. As another example, snippets in a result of search engine are also an example of indicative summary. Users judge whether they should read the webpage.

In contrast, the latter is used instead of its original document; it is assumed that users only read a summary, do not read its original document.

Generic or Query-Biased As the definition by Mani and Maybury [51], automatic summarization is done for a particular need. If a summarizer receives some specific indications when it generates a summary, it called as query-biased summarization. For example, producing snippets in a result of search engine is an example of query-biased summarization. A user's need is expressed as a query, and hence snippets must be generated by taking into account that query.

In contrast, generic summarization is done without specific queries.

Extractive or Abstractive Finally, automatic summarization can be divided with a method to produce a summary: extractive summarization and abstractive summarization.

The former generates a summary by extracting important sentences and organizing them. In extractive summarization, a summarizer receives a set of sentences as an input. A set of input documents is split into sentences before a summarizer receives them. Then a summarizer selects the important sentences as an output summary. This method reproduces the original sentences, and hence it does not tend to suffer from ungrammatical expressions in the output because those are originally written by humans.

In contrast, abstractive summarization generates expressions which are not included in an original document set. To generate the expressions, basically, an abstractive summarization system needs natural language understanding and generation modules. The system understands the meanings of input sentences based on the natural language understanding module, and then it generates the expressions with the natural language generation module by which the expressions are generated based on the meanings. Therefore, an accurate natural language understanding and generation are essential to produce abstractive summarization, but both of them are still not easy to achieve even with recent natural language processing techniques. Hence, most of recent automatic summarization studies have focused on the former approach, extractive summarization.

In this dissertation, we will deal with three tasks. All the tasks are in the setting of indicative, generic and extractive summarization. The tasks in the Chapter 4 and 6 are multi-document summarization. The task in the Chapter 5 is single document summarization.

Targets of summarization

As a target of automatic summarization, many kinds of documents have been studied.

Newswire Article Newswire articles are the most widely studied targets of automatic summarization [49, 2, 5, 35, 68, 92, 27, 75]. We are surrounded by a lot of news and automatic summarization is particularly useful for selecting the important information among Newswire articles.

Scientific Literature Scientific literature has also been a target of automatic summarization since the research started [49, 21, 66, 42, 55, 67]. In a particular field of science, there are a lot of articles and researchers must find out important information among them.

Book Branavan et al. proposed the method to summarize a book and generated a table-of-content as a summary [11]. To help users find out the important information from a long document like books, the structure of contents in the document is useful.

Speech Transcripts of speech are important targets of automatic summarization. Usually speech contains a verbose and redundant part, those should be removed to help users understand the important part of the speech. Speech summarization targets a transcript of the speech. In some cases, utterances are used as a unit for extraction, instead of sentences [34, 48, 82, 30, 29, 83].

E-mail Some work tried to summarize threads of e-mails [54, 71]. Since we receive a lot of e-mails daily, a summarization method for e-mails is useful.

Review Reviews on the Internet have been a target of automatic summarization [14, 45, 46]. A lot of users have posted reviews to express their opinions about products and services. Although these reviews can help users decide whether they should buy products, it is difficult to read all the reviews due to its sheer number. To address this problem, some previous work tried to summarize a set of reviews [14, 45, 46].

Microblog With the recent surge of microblogs such as twitter, they have become a target of automatic summarization. Although most articles of microblogs are short, the sheer volume of those are posted. Therefore, summarizing microblogs can be formulated as multi-document summarization. Takamura et al. proposed the method to summarize tweets [77]. They formulated the problem as the facility location problem and reported the result.

In this dissertation, we deal with newswire articles, speech and reviews as a target of summarization.

3.2 Methods

3.2.1 Sentence Extraction

Sentence extraction is an operation to extract a set of importance sentences in an input document set. The extracted sentences is used as an element of an output summary. The studies in sentence extraction can be divided into two subfields: model development and feature development.

Models for Sentence Extraction

Early work in automatic summarization heuristically defined importance on sentences in an input document set. Sentence extraction was done by selecting the most important sentence among the pool of sentences iteratively until the sum of length of selected sentences reached the maximum summary length. There was no explicit summarization model, but most of them in the early work can be formulated as follows:

$$\hat{S} = \operatorname*{argmax}_{S \in D} \sum_{s \in S} \text{score}(s), \tag{3.1}$$

$$\text{s.t.} \sum_{s \in S} \text{length}(s) \leq K \tag{3.2}$$

where D is a set of sentences in an input document set, S is a summary as a subset of D, s is a sentence in summary S, function $\text{score}(s)$ is a function that returns the importance of sentence s, K is the maximum summary length, and function $\text{length}(s)$

returns the length of sentence s. To maximize that objective function, most of the previous studies leveraged the greedy method [39].

In contrast to these heuristics, Filatova and Hatzivassiloglou introduced a mathematical model explicitly to formulate a task of automatic summarization as an instance of optimization problems. They formulated multi-document summarization as the maximum coverage problem. By introducing a formal model to automatic summarization, we can discuss the model and other parameters separately. After the work by Filatova and Hatzivassiloglou, some studies carried forward that work and developed novel models. Takamura and Okumura proposed a summarization model based on the facility location problem [93]. This problem is to select the sentences as the selected sentences entail the unselected sentences semantically as much as possible. Similarly, Shen and Li a problem to locate the minimum dominating set [72]. A major drawback of these models is computational complexity; finding the optimal solution of these problems are NP-hard, and hence a long is needed to solve the problems.

To address the problem resulting from the computational complexity, we propose a novel summarization model and decoding method in Chapter 6.

Features for Sentence Extraction

Another subfield concentrates on developing features identifying important sentences. In the 1950s, as the first study of automatic summarization done by Luhn [49], he found out that the frequency of words occurring in a target document could be used to identify important sentences. He divided words occurring in a document into three classes according to their frequency, and pointed out that the middle-frequency words can be a good indicator to identify important sentences. This is because most of the high-frequency words are stop words such as determiners and prepositions, and hence these words can ’t be clues to identify important sentences. The low-frequency words are also could not be used because they occur rarely in important sentences.

In the 1960s, Edmundson [21] carried forward the work by Luhn, he newly proposed several important features. The proposed features are summarized as follows: (1) the number of times a word appears in the document, (2) the number of words in the sentence that also appear in the title of the document or in section headings, (3) position of the sentences in the document and in the section, (4) the number of sentence words matching a pre-compiled list of cue words such as “In sum” [58]. Combined by a machine learning technique, these features were used in the recent studies [84, 92, 75].

Zechner [85] showed the results of the use of tf-idf [74] in automatic summarization. tf-idf is one of the most powerful and widely-used feature in automatic summarization [50, 88, 58]. Zechner used the raw value of tf-idf to identify important sentences; he computed importance of sentences based on tf-idf.

Recent approaches count on machine learning techniques to identify important sentences. [42, 32, 84, 92, 52, 75, 76, 80, 10]. Kupiec et al. [42] leveraged naive Bayes for sentence extraction. They regarded a sentence as a bag of words, and trained a predictor using pairs of an input document and its summary manually written as training data. In contrast to generative approaches such as the naive Bayes, discriminative approaches can combine various features naturally. Hirao et al. combined various features extracted from input documents using support vector machine [32]. Yih et al. used the logistic regression [84]. Takamura leveraged structured support vector machine to identify a set of important sentences [76].

3.2.2 Sentence Ordering

In multi-document summarization, after extracting appropriate sentences, these sentences are must be ordered appropriately. If sentences are extracted from the same document, they can be ordered as they are ordered in the original document. However in multi-document summarization, since there are multiple documents as an input, there is no original order between sentences extracted from different input documents.

To address this problem, some work proposed the method to order extracted sentences [6, 43, 1, 73, 22]. Previous work can be separated into two types according to its contribution.

Features for Sentence Ordering

Some studies concentrated on developing novel features for ordering [6, 43, 22]. Barzilay et al. suggested to order the sentences with publication dates of their original documents [6]. Lapata defined the transition probability between two sentences [43]. That probability is computed based on features extracted from two adjacent sentences and the sentences are ordered so as to maximize the sum of transition probability between adjacent sentences. Elsner et al. integrated features proposed for sentence ordering previously with machine learning [22].

Decoding for Sentence Ordering

Some studies concentrated on finding the optimal order with given feature sets [1, 73]. Since finding the shortest path connecting all nodes in a given set is an NP-hard problem, and hence the search itself can be a research question. Althaus et al. formulated the problem as the Traveling Salesman Problem [1]. They formulated the problem as an instance of ILP, and solved the problem with an ILP solver. Soricut and Marcu proposed a method to decode the problem with a variant of A^* search [73].

3.3 Evaluation

This section describes methods to evaluate a produced summary. There are two major aspects in the evaluation of summaries. One is informativeness, which indicates how a produced summary reflects important information in an input document set. Another is the linguistic quality, which indicates how well an output summary is organized as a document.

In this section, we first explain about informativeness. Then, we explain the linguistic quality.

3.3.1 Informativeness

Informativeness is a measure how well a summary reflects important information contained in original documents. There two main types of evaluation measure for informativeness: manual evaluation and ROUGE.

Manual evaluation is basically done by humans. Human subjects read an input document set, then they evaluate the summary produced from that document set according to some criteria, such as a 5-point scale.

Another measure is ROUGE, which is an automatic evaluation method for automatic summarization. ROUGE is an abbreviation for Recall Oriented Understudy for Gisting Evaluation, and was proposed by Lin [47]. ROUGE evaluates a summary by leveraging a reference summary, which is a summary manually produced by humans as an ideal example. ROUGE compares a summary automatically produced by some sort of summarization method with its references and evaluates the summary according to the similarity between the summary automatically generated and manually written. Among many variants of ROUGE proposed by Lin, $ROUGE-N$ is the most widely

used measure. ROUGE-N is computed based on n-grams. If unigrams are used to compute ROUGE-N, it is mentioned as ROUGE-1. ROUGE-N is computed as follows:

$$\text{ROUGE-N}(S;R) = \frac{\sum_{r \in \text{n-gram}(R)} \min(\text{count}(r;S), \text{count}(r;R))}{\sum_{r \in \text{n-gram}(R)} \text{count}(r;R)}, \tag{3.3}$$

where S and R are a summary and its reference respectively, s and r indicate a n-gram, function $\text{n}-\text{gram}()$ indicates the set of n-grams with a specific n in its argument, function $\text{count}(a,A)$ returns the number of n-gram a in document A, and function $\min()$ returns the minimum number in its arguments. That is, ROUGE-N indicates the similarity between a summary and its references based on n-grams with a specific n. If there are multiple references for one document set, ROUGE is computed for each reference and the maximum value among those is used as the final ROUGE [47]. This is because summaries produced by humans have some variety inherently, and hence a summary automatically produced should be compared with the reference which is the most similar to the summary among them. Lin showed that ROUGE-N had a strong correlation with manual evaluation [47].

Additionally, we explain ROUGE-SU, which is also widely used to evaluate the quality of a summary. ROUGE-SU takes into account skip-bigrams as well as unigrams and bigrams to compute its value. The skip-bigrams are any pairs of words in their sentence order, allowing arbitrary gaps [47]. Suppose that we have a sentence "A man walks on the campus". We can extract skip-bigrams such as "A, walks" and "man, campus" from the sentence. ROUGE-SU is computed as follows:

$$\begin{aligned}
\text{ROUGE-SU}(S;R) &= \frac{(1+\beta^2) \times \text{Recall}(S;R) \times \text{Precision}(S;R)}{\text{Recall}(S;R) + \beta^2 \text{Precision}(S;R)}, && (3.4)\\
\text{Recall}(S;R) &= \frac{\text{SU}(S;R)}{U+V+Y}, && (3.5)\\
\text{Precision}(S;R) &= \frac{\text{SU}(S;R)}{W+X+Z}, && (3.6)
\end{aligned}$$

where $\text{SU}()$ is a function that returns the number of common unigrams, bigrams and skip-bigrams between S and R, U, V and W are the number of unigrams, bigrams and skip-bigrams in R, respectively, X, Y and Z are the number of unigrams, bigrams and skip-bigrams in S, and β is a parameter to adjust weight of precision and recall. The maximum distance between words in a skip-bigram can be set as a parameter of

ROUGE-SU. If the maximum distance is set as 4, i.e. in the case of ROUGE-SU4, it takes into account unigrams, bigrams and skip-bigrams in which word pairs of at most 4 words apart.

There is a trade-off between manual evaluation and ROUGE. Manual evaluation needs human subjects to evaluate each summary, but it is expected that the result of manual evaluation is accurate. Additionally, manual evaluation can be done without references. In contrast, ROUGE can be used without human subjects, but it requires references. Since most of recent studies leveraged ROUGE for evaluation [84, 92, 27, 75], and it showed a strong correlation with manual evaluation as mentioned above, we will use ROUGE-N and ROUGE-SU so as to evaluate the informativeness of summaries in this dissertation.

3.3.2 Linguistic Quality

Linguistic quality is a measure to evaluate how well an output summary is organized as a document. The summary is a document helping readers understand the content of the original document set instead of reading it, and hence the summary must be well organized as a document. Sometimes the linguistic quality is referred as readability [58], and hence hereinafter sometimes the linguistic quality is referred as readability.

Basically, the linguistic quality of automatic summarization is measured manually by human subjects. The National Institute of Science and Technology (NIST) proposed five aspects as elements of linguistic quality so as to evaluate the summaries produced in the Document Understanding Conference (DUC): grammaticality, redundancy, referential clarity, focus, and structure and coherence [56]. Human subjects evaluate a summary according to these five criteria and then evaluate the overall quality of the summary.

Automatic evaluation of linguistic quality for automatic summarization was tried by Pitler et al. [64]. They built a predictor using results of human evaluation in DUC as training data. However, the accuracy of the predictor was not enough to judge the linguistic quality without manual evaluation. Furthermore, the manual evaluation done in DUC was for the summaries written in English, and hence it could not used directly to evaluate the summaries written in Japanese. Therefore, in this dissertation we will leverage manual evaluation for evaluation of linguistic quality.

Chapter 4

Joint Model for Sentence Extraction and Ordering

4.1 Introduction

With the recent expansion of the number of the Web users and commercial trades through the Web, the Web now holds a massive number of reviews describing the opinions of customers about products and services. These reviews can help the customer to reach purchasing decisions and guide the business activities of companies such as product improvement. However, it is almost impossible to read all reviews given their sheer number.

Automatic text summarization, particularly opinion summarization, is expected to allow all possible reviews to be efficiently utilized. Given multiple review documents, our summarizer outputs text consisting of ordered sentences. A typical summary is shown below [1]:

> 小町通りから一本入った路地裏にあるので、ちょっと道がわかりづらいですが、雰囲気は抜群です。料理も素晴らしく、特に鎌倉野菜をふんだんにサラダや相模湾で獲れた魚介類がおすすめです。お酒の種類も豊富で、特に日本酒が充実しています。

[1] Since in this chapter we will use the documents written in Japanese as examples in the evaluation, our example shown here is also written in Japanese. For readers' convenience, we show its English translation as follows: " Since the restaurant is located in a tiny alley, it is hard to locate the shop, but the atmosphere of the interior is outstanding. Foods, especially salads of profuse vegetables from Kamakura and seafoods from Sagami bay, are excellent. Also a lot of alcoholic beverages, especially Japanese sake, are served."

This summary is an example generated from a set of evaluative texts. Summarizing multiple evaluative texts can greatly improve users' information access to evaluative texts. This task is considered as multi-document summarization and some previous work tried to summarize multiple review documents [14, 45].

Existing multi-document summarizers focus on extracting sentences among given documents so as to include important information in the documents in a summary under some size limitation. This method is called "extractive summarization" [50]. A serious problem is that most of extractive summarizers completely ignore coherence of the summary, which improves reader's comprehension as reported by Barzilay et al. [6]. Incoherent summaries not only hamper the convenience of the summaries, which is to help readers understand the content of input documents quickly, but also make the reader misinterpret.

After sentence extraction, to make summaries coherent, the extracted sentences must be appropriately ordered [2]. However, most summarization systems delink sentence extraction from sentence ordering, so a sentence can be extracted that can never be ordered naturally with the other extracted sentences. Moreover, due to recent advances in decoding techniques for text summarization, the summarizers tend to select shorter sentences to optimize summary content. It aggravates this problem.

To overcome this weakness, this chapter introduces a novel joint model for sentence extraction and ordering. The model extracts and orders sentences simultaneously rather than extracts and orders sentences separately. The model is described as a novel Integer Linear Programming (ILP) formulation for searching for the optimal solution efficiently. The multi-document summarization task is formulated as an ILP problem that tries to optimize the content and coherence of the summary by extracting and ordering sentences simultaneously. The sentences are extracted so as to cover important information among the documents and are ordered so as to make the final summary coherent. The model is applied to opinion summarization and show that it outperforms state-of-the-art opinion summarizers in terms of ROUGE [47] evaluations. Additionally, we evaluate the coherence of summary manually and show that the readability of summary is improved by extracting and ordering sentences simultaneously. Although in this chapter we challenge our method with opinion summarization, it can be widely applied to other text generation and summarization tasks.

[2] Of course, there are other factors affecting reader's comprehension. For example, Document Understanding Conference 2007 held by the National Institute of Standards and Technology shows five factors that measure the linguistic quality of the summary: grammaticality, redundancy, referential clarity, focus and structure [56]. Sentence ordering is thought to mainly relate to the structure.

This chapter is organized as follows. Section 4.2 elaborates the background of this chapter. Section 4.3 describes a proposed model. Section 4.4 shows a way to set parameters in the model. Section 4.5 explains a decoding method to locate the optimal solution of the model. Section 4.6 reports our evaluation experiments. We conclude this chapter with Section 4.7.

4.2 Related Work

4.2.1 Multi-Document Summarization

Previous work has been adopted methods of multi-document summarization to summarize a set of evaluative texts about some sort of entity, where each evaluative text contains a writer's evaluation of that entity, such as a restaurant and commodity. [14, 45]. Carenini et al. reproduced the multi-document extractive summarization method designed for newswire articles by Radev et al. [68] to summarize evaluative texts [14]. First, their method extracts opinion information from the evaluative texts. The opinion information consists of an opinion aspect, expression and polarity. For example, an expression "The picture quality is good." consists of the aspect "picture quality", which is a property of something, the expression "good", and its polarity. Next, the score indicating how each pair of the aspect and its polarity is important as an element of the summary is given to the pairs based on the frequency of them in the input texts. Finally, the summary is generated so as to maximize the sum of scores of the pairs, where each pair consists of an aspect and polarity, contained in the summary. Lerman et al. also extracted pairs of an aspect and polarity as a unit from the input texts and they proposed the method that generated the summary so as to minimize the difference between the distributions of aspects and their polarity from the input texts and summary [45]. In this chapter, as with Carenini et al. and Lerman et al., a pair of an aspect and its polarity is used as a unit to summarize the input texts. Although previous work studied by Carenini et al and Lerman et al. [14, 45] focus on extracting appropriate sentences as a summary, the method proposed here takes into account the order of sentences simultaneously.

Recently, as a method which is generic and is not limited to evaluative texts, a multi-document summarization model based on the maximum coverage problem is widely used [23, 84, 27]. The summarization model based on the maximum coverage problem extracts sentences so as to cover concepts such as a unigram and bigram among the in-

put texts. Filatova and Hatzivassiloglou regarded a summarization task as an instance of the maximum coverage problem, solving the problem with the greedy method [23]. Yih et al. proposed a stack decoder to decode the model based on the maximum coverage problem. Takamura and Okumura [92] and Gillick and Favre [27] expressed the summarization model based on the maximum coverage problem as an instance of integer linear programming and showed that the optimal solution (i.e., the optimal summary) is located by branch-and-bound method, which is a generic algorithm that can solve integer linear programming. As with Takamura and Okumura [92] and Gillick and Favre [27], the summarization model based on the maximum coverage problem is used as the summarization model. Although in their model, a unigram or bigram is regarded as a unit to be covered so as to generate the summary, in the model proposed in this chapter, opinion information explained above is to be covered as much as possible so as to generate the summary. This is realized by making the opinion information extracted by opinion extractor the target of covering instead of either a unigram or bigram.

4.2.2 Sentence Ordering

It is known that the readability of a collection of sentences, a summary, can be greatly improved by appropriately ordering them [6]. Features proposed to create the appropriate order include publication date of document [6], content words [43, 1], and syntactic role of words [7, 9, 90]. Some approaches used machine learning to integrate these features [73, 22]. Generally speaking, these methods score the discourse coherence of a fixed set of sentences. Although these methods are separated from the extraction step so they may fail if the set includes sentences that are impossible to order naturally, our proposal jointly selects and orders sentences consisting of the input documents.

4.2.3 Headline Generation

Some studies attempt to generate a single sentence (i.e. headline) from the source document [4, 20]. While they extracted and ordered a *word* from the source document as a unit, our model use a *sentence* as a unit. This problem can be formulated as the Traveling Salesman Problem and its variants. Banko et al. [4] used beam search to

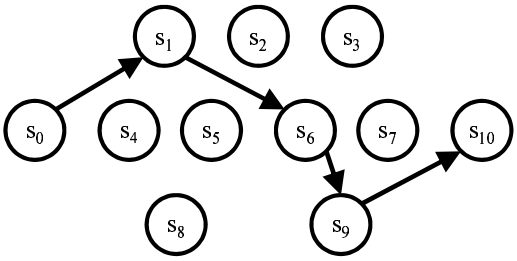

Figure 4.1: The graph expression of multi-document summarization.

identify approximate solutions. Deshpande et al. [20] leveraged ILP and a randomized algorithm to find the optimal solution.

4.3 Model

In this section, we elaborate the summarization model.

As an example, a set of nine sentences $D = s_1, s_2, ..., s_9$, which is composing a set of evaluative texts, are given. A summary consists of three sentences s_1, s_6, s_9 and the sentences are ordered as $s_1 \rightarrow s_6 \rightarrow s_9$. To express the beginning and terminal of the summary, symbol s_0 is added to the beginning of the sequence and symbol s_{10} is added to the terminal of the sequence, the summary is expressed as $S = \langle s_0, s_1, s_6, s_9, s_{10} \rangle$. In this case, summary S is expressed as an instance of directed acyclic graph, where s_0 is a source of the graph and s_{10} is a sink of the graph. As shown in Figure 4.1, an extractive summarization can be expressed as a graph connecting the nodes expressing the sentences with the directed arcs between source node s_0 and sink node s_{10}.

We describe a directed arc between s_i and s_j as $a_{i,j}$. The directed path shown in Figure 4.1 is decomposed into five nodes, $s_0, s_1, s_6, s_9, s_{10}$, and four arcs, $a_{0,1}, a_{1,6}, a_{6,9}, a_{9,10}$.

To represent the discourse coherence of two adjacent sentences, we define weight $c_{i,j}$ as the coherence score on the directed arc $a_{i,j}$. We assume that better summaries have higher coherence scores, i.e. if the sum of the scores of the arcs $\sum_{a_{i,j} \in A(S)} c_{i,j} a_{i,j}$, where $A(S)$ is a set of the arcs in the summary S, is high, the summary is coherent.

We also assume that the input document set D includes set of concepts $e \in E$. Each concept e is covered by one or more of the sentences in the document set. We show this schema in Table 4.1. The inclusion relations between sentences and concepts among the sentences can be expressed as a matrix. For example, if sentence s_i includes concept

Table 4.1: Sentence-Concept Matrix.

	e_1	e_2	e_3	...	e_6	e_7	e_8
s_1	1	0	0		1	0	0
s_2	0	1	0		0	0	0
s_3	0	0	0		0	0	1
$\vdots$				$\ddots$			
s_7	0	0	0		0	0	0
s_8	0	0	1		0	1	0
s_9	0	0	0		0	0	1

e_k, the element at i -th row and k -th column in the matrix is 1. According to Table 4.1, document set D has eight concepts $e_1, e_2, \ldots, e_7, e_8$ and sentence s_1 includes concepts e_1 and e_6 while sentence s_1 includes e_2. We consider each concept e_k has a weight w_k. We refer to that weight as concept score and assume that concept e_k will have a high weight w_k if it is important. Referring the set of concepts in the summary as $E(S)$, we assume that the higher the sum of scores $\sum_{e_k \in E(S)} w_k$ is, the better the summary is. My proposal improves summary quality by maximizing the sum of these weights.

We define, based on the above assumption, the following objective function:

$$\mathscr{L}(S) = \lambda \sum_{e_k \in E(S)} w_k + (1-\lambda) \sum_{a_{i,j} \in A(S)} c_{i,j}, \tag{4.1}$$

$$\text{s.t. length}(S) \leq L$$

where length(S) is a function returning the length of summary S, L is the limit of maximum summary size, and λ is a parameter adjusting the content score and coherent score.

Summarization is, in this chapter, realized by maximizing the sum of weights of concepts included in the summary and the coherence score of all adjacent sentences in the summary under the limit of maximum summary size.

Maximizing Equation 4.1 is NP-hard. If each sentence in the source document set has one concept (i.e. Table 4.1 is a diagonal matrix), Equation 4.1 becomes the Prize Collecting Traveling Salesman Problem [3], which is a variant of well-known NP-hard problem, the Traveling Salesman Problem. Therefore, a highly efficient decoding

method is essential. In this chapter, the problem is formulated as an instance of ILP and solved using a generic ILP solver. The decoding will be elaborated in Section 4.5.

4.4 Parameter Estimation

Our method requires three parameters: weights w of concepts, coherence c of two adjacent sentences, and λ. In this section, we describe the estimation of weights w and coherence c.

4.4.1 Content Score

In this chapter, as mentioned above, since we attempt to summarize reviews, we adopt *opinion* as a concept. We define opinion $e = \langle t, a, p \rangle$ as the tuple of *target* t, *aspect* a and its *polarity* $p \in \{-1, 0, 1\}$. We define target t as the target of an opinion. For example, the target t of the sentence "This digital camera has good image quality." is *digital camera*. We define aspect a as a word that represents a standpoint appropriate for evaluating products and services. With regard to digital cameras, aspects include *image quality*, *design* and *battery life*. In the above example sentence, the aspect is *image quality*. Polarity p represents whether the opinion is positive or negative. In this chapter, we define $p = -1$ as negative, $p = 0$ as neutral and $p = 1$ as positive. Thus the example sentence contains opinion $e = \langle digital\ camera, image\ quality, 1 \rangle$.

As we will elaborate later, in this chapter we will use a set of reviews about some sort of either restaurant or commodity as an input of our method. Since all of input documents correspond to a particular restaurant or commodity, we assume that all of opinions occurring in the documents are the opinions for a particular restaurant or commodity the input documents correspond to. For this reason we henceforth omit *target* t from an opinion tuple $e = \langle t, a, p \rangle$ for simplicity; therefore, we write an opinion as a pair of *aspect* a and its *polarity* p, i.e., $e = \langle a, p \rangle$.

Opinions are extracted using a dictionary which consists of opinion expressions and their polarities and pattern matched from dependency trees of sentences. The dictionary contains [3] consists of pairs of *opinion expressions* and their polarities, for example, *delicious*, *friendly* and *good* as positive opinion expressions, *bad* and *expensive* are negative opinion expressions.

[3] Since the aim of this study is to summarize Japanese reviews, we utilize the dictionary whose entries are Japanese [89]. However, our method is, except for opinion extraction, language independent.

Opinions are extracted from given sentences as following steps:

1. Performing dependency parsing on sentences in the input documents.

2. Identifying opinion expressions among words consisting of parsed sentences. For example, in the case of the sentence "This restaurant offers customers delicious foods and relaxing atmosphere.", *delicious* and *relaxing* are identified as opinion expressions.

3. If the opinion expressions are identified, the expressions and its aspects are extracted as aspect-opinion expression pairs from dependency tree using some rules. In the case of the example sentence, *foods* and *delicious*, *atmosphere* and *relaxing* are extracted as aspect-opinion expression pairs.

4. Finally extracted opinion expressions are converted to polarities; then we acquire the set of opinions from sentences, for example, $\langle$ *foods*, $1\rangle$ and $\langle$ *atmosphere*, $1\rangle$.

Content score w_k is given to the pair consisting the aspect and its polarity. In this chapter, the occurrence of e_k in the input documents is set as score w_k. Although the model will pick up the opinions mentioned by many reviewers in this setting, there can be important opinions rarely mentioned in the input documents. A promising way to identify such kinds of opinions is to use a machine learning technique. For example, important opinions can be identified by including information on who wrote these opinions. However, we leave it future work, because the aim of this chapter is to make a summary more readable.

4.4.2 Connect Score

In this section, we define coherence score c. As explained above, coherence score $c_{i,j}$ is given to the arc between two sentences s_i and s_j. Given two sentences $d = \{s_i, s_j\}$ and its order $o = \langle s_i, s_j \rangle$, its coherence score is defined as follows:

$$c_{i,j} = \mathbf{w} \cdot \phi(d, o), \tag{4.2}$$

where $\mathbf{w}$ is a weight vector, $\phi(d,o)$ is a feature vector of two sentences $d = \{s_i, s_j\}$ and its order $o = \langle s_i, s_j \rangle$, and $\mathbf{w}^\top \phi(d,o)$ is a dot product of $\mathbf{w}$ and $\phi(d,o)$.

As features consisting of a feature vector, features proposed by Lapata [43] are used. The features of two adjacent sentences are the Cartesian product of following elements: surface forms of noun, verb and adjective, their part-of-speech tags, named entity tags such as LOC and ORG, and conjunctions. Taking the summary shown in Section 4.1 as an example, From the sentences "Since the restaurant is located in a tiny alley, it is hard to locate the shop, but the atomosphere of the interior is outstanding." and "Foods, especially salads of profuse vegitables from Kamakura and seafoods from Sagami bay, are excellent." in that summary, ordered pair $(alley, Foods)$ from the words "alley" and "Foods" in that sentences. The ordered pair extracted from adjacent sentences is used as a feature consisting of a feature vector. Symbols indicating the beggining and end of the summary, s_0 and s_{n+1} are represented as tags $< d >$ and $< /d >$. For example, if the first sentence in the summary includes word "alley", ordered pair $(< d >, alley)$ is used as a feature. Similary, if the last sentence in the summary includes word "food", order pair $(food, < /d >)$ is used as a feature.

A coherence score of sentences in whole summary is defined as the sum of the coherence score of each two adjacent sentences, i.e. a feature vector of a set of sentences $\mathbf{d} = \{s_0, s_1, \ldots, s_n, s_{n+1}\}$ and its order $\mathbf{o} = \langle s_0, s_1, \ldots, s_n, s_{n+1} \rangle$ is expressed as follows:

$$\Phi(\mathbf{d}, \mathbf{o}) = \sum_{d,o} \phi(d, o). \tag{4.3}$$

Hence a coherence score of sentences in whole summary is a dot product of a weight vector $\mathbf{w}$ and a feature vector $\Phi(\mathbf{d}, \mathbf{o})$.

Parameter vector $\mathbf{w}$ is estimated using training examples. We assume that training examples $\{\mathbf{d}_t, \mathbf{o}_t\}_{t=1}^{T}$ as a set where each pair consists of a set of sentences $\mathbf{d}_t$ and its order $\mathbf{o}_t$. In this chapter, as with the previous work by Lapata [43], we assume that sentences consisting of a document written by human are ordered correctly. Therefore, by the use of sentence splitting, morphological analysis and name entity recognition, training examples are obtained without human annotations. The details of training examples will be elaborated in Section 4.6. Given training examples, if the parameter vector gives the higher score to correct order $\mathbf{o}_t$ of a set of sentences $\mathbf{d}_t$ than wrong order [4] , it is expected that parameter vector $\mathbf{w}$ may give a higher score to natural order than unnatural order. Based on this assumption, parameter vector $\mathbf{w}$ is learned so as to make a correct order get a higher score than an order.

[4]Given n sentences, there can be ${}_nP_n - 1$, i.e. $n! - 1$ candidates of wrong order $\hat{\mathbf{o}}$. Symbols indicating the beginning and terminal of order, s_0 and s_{n+1} , are not the targets of order estimation.

We use the Passive-Aggressive algorithm [18] explained in Section 2.2.2 to find $\mathbf{w}$. The Passive-Aggressive algorithm is an online learning algorithm that updates the parameter vector by taking up one example from the training examples and outputting the solution that has the highest score under the current parameter vector. If the output differs from the training example, the parameter vector is updated. In this chapter, correct order $\mathbf{o}_t$ in the training examples and order $\hat{\mathbf{o}}$ which is estimated with current parameter vector are compared, and if they differ the parameter vector is updated as follows;

$$\begin{aligned} \mathbf{w}' &= \underset{\mathbf{w}'}{\operatorname{argmin}} \tfrac{1}{2}||\mathbf{w}' - \mathbf{w}||^2 + C\xi^2, \\ \text{s.t. } & \mathbf{w}' \cdot \Phi(\mathbf{d}_t, \mathbf{o}_t) - \mathbf{w}' \cdot \Phi(\mathbf{d}_t, \hat{\mathbf{o}}) + \xi \geq \ell(\hat{\mathbf{o}}; \mathbf{o}_t), \end{aligned} \tag{4.4}$$

Where $\mathbf{w}$ is a parameter vector before the update, $\mathbf{w}'$ is a parameter vector after the update, $\ell(\hat{\mathbf{o}}; \mathbf{o}_t)$ is a loss function, C is a aggressiveness parameter, and ξ is a slack variable. When order $\hat{\mathbf{o}}$, which is estimated with current parameter $\mathbf{w}$, is not the same as correct order $\mathbf{o}_t$, i.e. $\mathbf{o}_t \neq \hat{\mathbf{o}}$, parameter vector before update $\mathbf{w}$ is updated by following equation, which is induced by solving Equation 4.4:

$$\mathbf{w}' = \mathbf{w} + \eta(\Phi(\mathbf{d}_t, \mathbf{o}_t) - \Phi(\mathbf{d}_t, \hat{\mathbf{o}})) \tag{4.5}$$

$$\eta = \frac{\ell(\hat{\mathbf{o}}; \mathbf{o}_t) - \Phi(\mathbf{d}_t, \mathbf{o}_t) + \Phi(\mathbf{d}_t, \hat{\mathbf{o}})}{||\Phi(\mathbf{d}_t, \mathbf{o}_t) - \Phi(\mathbf{d}_t, \hat{\mathbf{o}})||^2 + \frac{1}{2C}} \tag{4.6}$$

When updating the parameter vector, this algorithm requires the solution that has the highest score under the current parameter vector, so we have to run an argmax operation. Since we are attempting to order a set of sentences, the operation is regarded as solving the Traveling Salesman Problem [1]; that is, we locate the path that offers the maximum score through all n sentences where s_0 and s_{n+1} are starting and ending points, respectively. This operation is NP-hard and it is difficult to find the global optimal solution. To overcome this, we find an approximate solution by beam search [5] .

As a loss function, Kendall's tau $\ell(\hat{\mathbf{o}}; \mathbf{o}_t)$ is used;

[5]Obviously, ILP can be used to search for the path that maximizes the score. While beam search tends to fail to find out the optimal solution, it is tractable and the learning algorithm can estimate the parameter from approximate solutions [86] . For these reasons we use beam search.

$$\ell(\hat{\mathbf{o}}; \mathbf{o}_t) = 1 - \tau, \tag{4.7}$$

$$\tau = 1 - 4\frac{\mathrm{Swap}(\hat{\mathbf{o}}, \mathbf{o}_t)}{N(N-1)}, \tag{4.8}$$

where τ is Kendall's tau, $\mathrm{Swap}(\hat{\mathbf{o}}, \mathbf{o}_t)$ is the number of operations that swap two adjacent elements, which are sentences in this case, to transform order $\hat{\mathbf{o}}$ to $\mathbf{o}_t$, and N is the number of elements, i.e. the number of sentences. As Lapata's study [44], Kendall's tau correlates closely with human evaluation in terms of adequateness of sentence order, and hence it is expected that a parameter vector that can reproduce a natural order of sentences can be obtained by minimizing a loss based on Kendall's tau.

4.5 Decoding

This section describes an ILP formulation of the above model and a way to decode the model. The decoding is a way to locate the summary S maximizing 4.1. In this chapter, 4.1 is expressed as an instance of ILP; given esimated parameters , the optimal solution is located by solving the problem with a genetic solver.

In this chapter, we prepare three decision variables, x, y, and z. Variable x_i indicates the inclusion of sentence s_i. If sentence s_i is part of the summary, then x_i is 1. If it is not part of the summary, then x_i is 0. Variable $y_{i,j}$ indicates the adjacency of the s_i and s_j. If these two sentences are included in the summary and they are ordered as $s_i \rightarrow s_j$, then $y_{i,j}$ is 1. If not, $y_{i,j}$ is 0. Variable z_k indicates the inclusion of concept e_k. For example, in the case of Figure 4.1, variables about sentences, $x_0, x_1, x_6, x_9, x_{10}$, are 1, and variables about arcs, $y_{0,1}, y_{1,6}, y_{6,9}, y_{9,10}$ are 1. Variable z_k is 1 if concept e_k is included in the summary, according to sentence-concept matrix shown in Table 4.1.

4.5.1 Objective Function

The above objective function, Equation 4.1, is represented as follows:

$$\max \left\{ \lambda \sum_k w_k z_k + (1-\lambda) \sum_{i,j} c_{i,j} y_{i,j} \right\} \tag{4.9}$$

Equation 4.9 attempts to cover as much of the concepts included in input document set as possible according to their weights w and orders sentences according to discourse coherence c. λ is a scaling factor to balance w and c.

4.5.2 Constraints

Some constraints are imposed on Equation 4.9 to acquire the optimum solution.

First, we range the above three variables x, y, and z as follows:

$$\begin{aligned} x_i &\in \{0,1\} \quad (\forall i), \\ y_{i,j} &\in \{0,1\} \quad (\forall i,j), \\ z_k &\in \{0,1\} \quad (\forall k). \end{aligned}$$

These three variables are all binary variable, where their value is either 0 or 1. In our model, a summary cannot include the same sentence or arc twice. Although the same concept can be included in the summary twice, it is counted once on the objective function, and hence the concept contributes its score to the objective function only once. Taking Table 4.1 for example, if s_3 and s_9 are included in a summary, the summary has two e_8, but e_8 is 1 because variable z_8 is a binary variable. This constraint avoids summary redundancy [92, 27].

The summary must meet the condition of maximum summary size L. The following inequality represents the size constraint:

$$\sum_i l_i x_i \leq L,$$

where l_i indicates the length of sentence s_i. L is the maximum size of the summary.

The following inequality represents the relationship between sentences and concepts in the sentences:

$$\sum_i m_{i,k} x_i \geq z_k \quad \forall k,$$

where $m_{i,j}$ is an element of Table 4.1. The above constraint represents Table 4.1. If s_i is not included in the summary, the concepts in s_i are not included.

To represent the beginning and end of the summary, decision variable x_0 corresponding s_0 and decision variable x_{n+1} corresponding s_{n+1} are introduced:

$$\begin{aligned} x_0 &= 1, \\ x_{n+1} &= 1, \end{aligned}$$

where n is the number of sentences in the input document set. Symbols indicating the beginning and end of the summary must be part of the summary, and hence $x0$ and x_{n+1} are must be 1.

Next, we describe the constraints placed on arcs.

The beginning symbol must be followed by a sentence or a symbol and must not have any preceding sentences/symbols. The end symbol must be preceded by a sentence or a symbol and must not have any following sentences/symbols. The following equations represent these constraints:

$$\begin{aligned} \sum_i y_{0,i} &= 1 \\ \sum_i y_{i,0} &= 0 \\ \sum_i y_{n+1,i} &= 0 \\ \sum_i y_{i,n+1} &= 1 \end{aligned}$$

Each sentence in the summary, excepting the beggining node and end node, must be preceded and followed by a sentence/symbol.

$$\begin{aligned} \sum_i y_{i,j} + \sum_i y_{j,i} &= 2x_j \ \forall j \\ \sum_i y_{i,j} &= \sum_i y_{j,i} \ \forall j \end{aligned}$$

The above constraints fail to prevent cycles. To rectify this, we set the following constraints.

$$
\begin{aligned}
\sum_i f_{0,i} &= n \\
\sum_i f_{i,0} &\geq 1 \\
\sum_i f_{i,j} - \sum_i f_{j,i} &= x_j \ \forall j \\
f_{i,j} &\leq n y_{i,j} \ \forall i, j
\end{aligned}
$$

The above constraints indicate that *flows* f are sent from s_0 as a source to s_{n+1} as a sink. n unit flows are sent from the source and each node expends one unit of flows. More than one flow has to arrive at the sink. By setting these constraints, the nodes consisting of a cycle have no flow. Thus solutions that contain a cycle are prevented. These constraints have also been used to avoid cycles in headline generation [20].

4.6 Experiment

This section shows the results evaluation to verify the effectiveness of our proposal. We tested our method in terms of how adequately the generated summaries reflect the content of input documents and how readable the generated summaries are. The results of evaluation about the content of summary is shown in Section 4.6.1, the results of evaluation about the readability of summary is shown in Section 4.6.2. To generate the summary, a generic solver, IBM ILOG CPLEX [6] was used to decode the problem. We attempted to generate 300 byte summaries, According to the development set, λ in Equation 4.9 was set as 0.5, i.e. equally weigh the content score and coherence score [7]. To segment the words and identify their pos tags, morphological analyzer by Fuchi and Takagi [24] was used. To parse the sentences, dependency analyzer by Imamura et al. [36] was used. As an opinion expression dictionary, the dictionary built by Asano et al. [89] was used.

[6] http://www-06.ibm.com/software/jp/websphere/ilog/optimization/core-products-technologies/cplex/

[7] We prepared the development set separetely from the corpus used in this experiments, and adjusted parameter λ by verifing the summaries generated from the development set. Ideally speaking, parameter λ should be adjusted so as to maximize the measurement that can evaluate the general quality of summary, which include the content and readability. However, at the this moment, the measurement like the above must be done manually, and hence it requires a lot of cost. Therefore, in this chapter, λ was tuned by the author. Optimizing λ is a interesting problem as well as establishing generic automatic evaluation measurement for summarization.

Table 4.2: Statistics of corpus.

	Average number of documents per set	Average size of document set (byte)
Restaurant	29.40	5,343
Commodity	44.75	10,173

To estimate parameter $c_{i,j}$, 4,390 reviews of restaurants and 47,570 reviews of commodities were collected separetely from the corpus for evaluation [8]. Parameter C in Equation 4.6 was set as 100.

4.6.1 Content

To evaluate the summary in terms of its content, 4,475 reviews of 100 commodities and 2,940 reviews of 100 restaurants were collected from websites. The commodities included items such as digital cameras, printers, video games, and wines. The statistics of the corpus is shown in Table 4.2. To evaluate the content of the summary, 4 reference summaries were prepared for each document set. These references were written by hands and the writers had read corresponding reviews. The references corresponding to the same restaurant or commodity were written by different writers. The author was not included among the writers.

To evaluate the content of the summary, automatic evaluation measure for summarization proposed by Lin [47], ROUGE, was used. ROUGE computes the similarity between the summary generated by the program and the reference summary written by human. Among ROUGE variants, ROUGE-2, ROUGE-SU4, and ROUGE-SU9 were used.

To compute the value of ROUGE of the summary, the value between the summary and each reference was computed and then the maximum one among the four values were used as the final value of ROUGE of the summary. Furthurmore, Hirao et al. [94] reported that only noun, verb, adjective and unknown word should be used to compute ROUGE because it shows higher correlation. Hence, in this chapter, only noun, verb, adjective and unknown word were used to compute ROUGE.

[8] The content of reviews of commodities might be more diverse than restaurants, and hence in the domain of commodities more parameter $c_{i,j}$ must be estimated. Therefore, more reviews of commodities were prepared.

Comparative Methods

Following 6 methods were compared to verify the effectiveness of the proposal.

- **Content**: This method covers content words (i.e. noun, verb, adjective and unknown word) as consepts instead of opinion information. The weight of content word was set as its frequency.
- **Carenini**: This method was proposed by Carenini et al. [14]. The detail of the method will be explained below.
- **Lerman**: This method was proposed by Lerman et al. [45]. The detail of the method will be explained below.
- **Method (1)**: This method performs extraction and ordering of sentences separetely. Input documents are summarized by following steps:
  1. First, sentences among the input documents are extracted with the first argument of the objective fucion, i.e. sentences are extracted only with content score. Sentence ordering is not performed.
  2. Then, the extracted sentences are ordered according to the second argument of the objective function, i.e. coherence score.
- **Method (2)**: This is the proposal. This method jointly extract and order sentences in the input document.
- **Human**: This shows the maximum value of ROUGE among the values of pairs of two references in four references. This indicates the upper bound of evaluation.

The sentences extracted with the method of Content, Carenini, and Lerman were ordered according to publication date of document. If two or more sentences were extracted from the same document, they were ordered according to the order of them in the original document.

By comparation of these 6 methods, this section clarifies following 3 points:

- The effect of using opinion information as a concept. It is expected that the use of opinion information as a concept improves the quality of summary because an input is a set of reviews. This effect is verified by comparison of Content and Method (1), and Content and Method (2).

- The advantage over the previous methods in terms of content of summary. As mentioned above, the main contribution of this study is to produce the summary which has better readability. However, even if the readability is improved by the proposal, it is not always productive if the content of the summary deteriorates in compensentation for the improvement of readability. Hence the previous methods and the proposal should be compared in terms of their content. This is verified by comparison of Carenini and Method (2), and Lerman and Method (2).

- The effect of joint inference of extraction and ordering for the content. The quality of the content of summary generated by joint inference of extraction and ordering, Method (2), might be lower than Method (1), because Method (1) extracts sentences according to only content score in contrast to Method (2) which take into account coherence score when it extracts sentences. Hence, the quality of content between Method (1) and Method (2) should be compared.

Baselines

This section elaborates two baselines, the method proposed by Carenini et al. [14] and the method proposed by Lerman et al. [45].

Carenini Carenini et al. proposed two opinion summarizers [14]. One uses a natural language generation module, and other is based on MEAD [68]. Since it is difficult to mimic the natural language generation module, we implemented the latter one. The objective function proposed by Carenini et al. is as follows:

$$\mathscr{L}_1(S) = \sum_{a \in S} \sum_{s \in D} |\text{polarity}_s(a)| \tag{4.10}$$

$\text{polarity}_s(a)$ indicates the polarity of aspect a in sentence s present in source document set D. That is, this function gives a high score to a summary that covers aspects frequently mentioned in the input, and whose polarities tend to be either positive or negative.

The solution is identified using the greedy method. If there is more than one sentence that has the same score, the sentence that has the higher centroid score [68] is extracted.

Lerman Lerman et al. proposed three objective functions for opinion summarization [45], and we implemented one of them. The function is as follows:

$$\mathscr{L}_2(S) = -(\mathrm{KL}(p_S(a), p_D(a)) + \sum_{a \in A} \mathrm{KL}(\mathscr{N}(x|\mu_{a_S}, \sigma^2_{a_S}), \mathscr{N}(x|\mu_{a_D}, \sigma^2_{a_D}))), \tag{4.11}$$

where $\mathrm{KL}(p,q)$ means the Kullback-Leibler divergence between probability distribution p and q, $p_S(a)$ and $p_D(a)$ are probability distributions indicating how often aspect $a \in A$ occurs in summary S and source document set D respectively, $\mathscr{N}(x|\mu, \sigma^2)$ is a Gaussian distribution indicating distribution of polarity of an aspect whose mean is μ and variance is σ^2, and μ_{a_S}, μ_{a_D} and $\sigma^2_{a_S}, \sigma^2_{a_D}$ are the means and the variances of aspect a in summary S and source document set D, respectively. These parameters are determined using maximum-likelihood estimation.

That is, the above objective function gives high score to a summary whose distributions of aspects and polarities mirror those of the source document set.

To identify the optimal solution, Carenini et al. used a randomized algorithm. First, the summarizer randomly extracts sentences from the source document set, then iteratively performs insert/delete/swap operations on the summary to increase Equation 4.11 until summary improvement saturates. While this method is prone to lock onto local solutions, the summarizer can reach the optimal solution by changing the starting sentences and repeating the process. In this experiment, we used 100 randomly selected starting points.

Results and Discussions

The results are shown in Table 4.3 and 4.4. Table 4.3 shows the results of ROUGE in the reviews of restaurants, Table 4.4 shows the results of ROUGE in the reviews of commodities. As shown both Tables, Method (1) and Method (2) surpass methods of Content, Carenin and Lerman. In terms of ROUGE-2, Method (1) and Method (2) surpass other methods, excepting Human, significantly ($p < 0.05$) [9] . In terms of ROUGE-SU4 and ROUGE-SU9, Method (1) and Method (2) surpass methods of Content and Carenini significantly, but not Lerman.

As shown in Table 4.3 and 4.4, Content could not identifies the important information among the input documents. Although the method tries to cover the content words

[9]The statistical significance was verified by Wilcoxon's signed rank test [79].

Table 4.3: Results of ROUGE evaluation on the reviews of restaurants.

	ROUGE-2	ROUGE-SU4	ROUGE-SU9
Content	0.240	0.278	0.255
Carenini	0.249	0.281	0.258
Lerman	0.260	0.296	0.268
Method (1)	0.285	0.312	0.283
Method (2)	0.283	0.313	0.281
Human	0.357	0.367	0.335

Table 4.4: Results of ROUGE evaluation on the reviews of commodities.

	ROUGE-2	ROUGE-SU4	ROUGE-SU9
Content	0.161	0.199	0.185
Carenini	0.158	0.202	0.186
Lerman	0.205	0.247	0.227
Method (1)	0.231	0.261	0.240
Method (2)	0.233	0.258	0.245
Human	0.384	0.405	0.361

in the documents, there are a lot of content words not related to the information that should be captured, i.e. opinion information.

Since the method proposed by Carenini et al. [14] was designed so as to cover the important opinion information in the input documents, the intention of the method proposed by Carenini et al. [14] is similar to the proposal of this study. However, Carenini et al. did not describe the objective fuction explicitly, and its process of generating summary is ad-hoc. In contrast to their method, the advantage of the proposal is to leverage a method developed in the field of combinatorial optimization to solve the problem by formulating a problem of summarization as an instance of ILP explicitly. As a result shown in 4.3 and 4.4, Method (1) and Method (2) achieved the better performance than the method previously proposed in terms of content.

The method proposed by Lerman et al. [45] showed the better performance than Content and Carenini, it fell short of Method (1) and Method (2). A drawback in their method is redundancy. Their method tries to close not only the distributions of polarities of summary and input documents, but also the distribustions of the number of

aspects in summary and input documents. If 40% of aspects in the input documents are the same aspect, the objective function proposed by Lerman et al. values the summary in which 40% of aspects are the same aspect. The summary mostly consisting of the same opinion information is redundant. In contrast to it, the proposal is designed to cover the important opinion information as diverse as possible, and hence it can avoid the redundant summary.

There is no statistical significance between Method (1) and Method (2). Therefore, joint inference of extraction and ordering, Method (2), can generate the summary as well as Method (1), the method extracting and ordering sentences separetely.

As shown in Table 4.3 and 4.4, all methods fell short of Human. The summary generated by human contains not only opinion information but also information related opinions such as basis of the opinions. For example, in the summary shown in Figure 1, it contains the reason why "it is hard to locate the shop" as "the shop is located in the tiny alley". The proposed summarization model does not have a way to identify and take in such information. To produce the summary more similar to ones produced by human, further investigation for information that should be taken into the summary and ways to extract such information from the input documents are needed.

4.6.2 Readability

To evaluate the readability of the summary, manual evaluation were performed [10]. The set of summary used in the evaluation was generated from reviews of restaurants and commodities explained in Section 4.6.1 with two method explained below.

The evaluation was based on pairwise testing; two summaries were shown to the subject and the subject was asked to select the better summary between them. Given summary A and B, the question consists of five choices: A is better than B, A tend to be better than B, A is equal to B, B tend to be better than A, B is better than A.

The subjects were 10 Japanese native speakers. They do not include the author. When the subject evaluates the summary, the method to generate the summary is informed to the subject. For each summary, two subjects give evaluation, and hence each subject evaluate 40 pairs of summary in both domains of restaurant and commodity.

[10] Recently, automatic evaluation of readability of summary has been explorered [64], but there is no standard method.

Table 4.5: Evaluation of readability.

	Restaurant	Commodity
Method (1) is better than Method (2)	18	21
Method (1) tends to be better than Method (2)	20	17
Method (1) is equal to Method (2)	32	41
Method (2) tends to be better than Method (1)	43	23
Method (2) is better than Method (1)	87	98

Comparative Methods

To evaluate the proposal, following two methods were compared:

- **Method (1)**: select and order sentences separately.
- **Method (2)**: select ans order sentences simultaneously.

By the comparison of the methods above, following points are clarified:

- The effect of joint inference of extraction and ordering for readability. It is expected that the joint inference can reduce the number of unreadable summary generated by performing these operations separetely.

Results and Discussions

The results are shown in Table 4.5. As shown in Table 4.5, the proposal improved the readability of the summary in both domains.

Two examples are shown below; these summaries are generated from the same input documents but by different methods, Method (1) and Method (2). The first one is generated by Method (1) [11], and the second one is generated by Method (2) [12].

[11] Since the documents in the aforementioned corpora are written in Japanese, the following summary is also Japanese. We show its English translation here: " Rice is served by Japanese style wooden tub, it helped me feel special, and the foods were mostly Japanese style and quite delicious. I was very pleased because a lot of small dishes are prepared and the amount of rice is quite large. Japanese nouvelle cuisine looks good and the view from the restaurant is excellent. I recommend the lunch with rice served by Japanese style wooden tub. The restaurant is quite good. The atmosphere of the restaurant is also excellent; the Ferris wheel can be seen from the table inside the restaurant. The night view is awesome."

[12] The English translation of the second one is follows: "The night view is awesome. Japanese nou-

おひつに入ったごはんなど、特別な気分も味わえ、お料理も和風で美味しかったです。小鉢も沢山でご飯の量も多くて大満足です。創作和食で食卓も美しく景色も美しい。お昼のおひつご飯もオススメ。いいお店。お店の雰囲気もよく、観覧車が目の前でとても綺麗でした。夜景がとっても綺麗です。

お店から見る夜景がとても綺麗でした。創作和食で食卓も美しく景色も美しい。小鉢も沢山でご飯の量も多くて大満足です。おひつに入ったごはんなど、特別な気分も味わえ、お料理も和風で美味しかったです。お店の雰囲気もよく、観覧車が目の前でとても綺麗でした。デートにお奨めのお店です。

In these examples, the subject assessed that the second one, which is generated by Method (2), our proposal, is better than the first one [13] . By extracting and ordering sentences simultaneously, it can be seen that sentences extracted by Method (2) are different from ones extracted from Method (1), the method extracting and ordering separately, as shown above. While the second example of summary by our proposal, Method (2), include sentence“デートにお奨めのお店です” (I recommend the restaurant for dating) on the last of the summary, the first example does not include that sentence. In the review articles, sometimes some situations where a commodity or restaurant can be recommended to buy or use are written on the ends of the articles. Our proposal could generate the summary similar to its reference by learning such structure through word “お奨め” (recommendation), i.e., sentence orders similar to those organized by humans can be reproduced through feature $(\text{お奨め}, < /d >)$ having a high weight as a result of learning,

One of the factors deteriorating the readability is that the similar topics occur away from each other; the topic about “個室” (private dining room) occurred twice in the

velle cuisine looks good and the view from the restaurant is also excellent. I was very pleased because a lot of small dishes are prepared and the amount of rice is quite large. Rice is served by Japanese style wooden tub, it helped me feel special, and the foods were mostly Japanese style and quite delicious. The atmosphere of the restaurant is excellent; the Ferris wheel can be seen from the table inside the restaurant. I recommend the restaurant for dating.”

[13]These summaries are originally written in Japanese because original documents are written in Japanese. To take them as an example, the author translated them into English.

summary shown in below [14]:

> 個室が多く、周りが騒がしくなくお酒が飲めるのが良かった。店の雰囲気もよかった。値段も手ごろ。ヘルシーな感じで体に良さそうな料理です。照明が暗く、デートにお勧めです。和食が大好きです。大事な接待で利用しました。噂どおりのすごい店。個室が素晴らしい。

Topics related to each other should occur closely in the summary, or it gives an odd impression to readers. However, the proposal assumes the first-order Markov process, and hence similar topics can occur across other topics. One way to handle this problem is to add the constraint that limits the number of occurrences of the same topic in the summary to only once. By adding this constraint, the summary only includes the first sentence or second sentence, it can avoid the occurrence of the same topic. However, this constraint decreases the flexibility of sentence extraction, and hence it can degrade the quality of the content of the summary. Although this negative effect can be reduced by splitting complex sentences like the first sentences of summary shown above, we leave this future work. Assuming the higher order Markov process can treat these problems, but it broadens search space and aggravate the difficulty of decoding, and hence it is impractical. Another way might be to learn the domain-specific order of topics [15], the granularity of domains is not trivial.

At the end of discussion, an interesting point is whether the proposed joint model can improve the readability of documents other than reviews, such as newswire articles. Previous work only focusing on sentence ordering for extracted sentences from newswire articles showed that the readability of a summary could be improved by sentence ordering [6, 63]. As shown above, our joint method improved the readability of a summary significantly; therefore, reviews might be the domain in which our joint model could easily improve the readability of a summary. However, our goal in this chapter was to improve the readability of a summary generated from a set of reviews. On that point our model achieved this goal. To apply the proposed model to other domains can be promising future direction.

[14] The English translation of the following summary is: “There were a lot of private dinning rooms, and hence the restaurant was not noisy and we could have a drink in a quiet manner. The atmosphere was also good. The price is accessible. Foods seem to be healthy and good for the body. Since the restaurant is somewhat dark, it can be recommended for dating. I like Japanese foods. I utilized the restaurant for an important dinner. The restaurant is awesome as well as I heard. The private room is excellent.”

4.7 Conclusion

In this chapter, a novel summarization model to summarize evaluative texts is proposed. The novelty and contributions are summarized below:

- In this chapter, a novel summarization model extracting and ordering sentences simultaneously is proposed. The proposed model is to maximize the information to be covered and coherence between sentences, and is formulated as an instance of ILP. The summary is generated by solving the problem.
- It is proved that the summary that is equal to the summary generated by a baseline model that extracting and ordering separately in terms of ROUGE, and is better than the summary generated by the baseline in terms of readability can be generated by the use of the proposal.

As future work, as described above, the information other than opinion information that should be included in the summary should be identified.

Chapter 5

Transfer Learning for Content Selection

5.1 Introduction

Contact center dialogue summarization is attracting much more attention [13, 30, 29]. If contact center dialogues can be summarized automatically, business enterprises can extract valuable information from the summaries and leverage the data to improve their businesses and make better decisions.

Implementing an automatic summarization system that outputs good summaries requires the manual estimation of parameters from a set of pairs of documents and their references [32]. If there is a sufficient number of training samples, the summarizer can learn what summaries are expected. That is, a summarizer can generate good summaries if it uses the parameters estimated from a sufficient number of training samples.

However, preparing a sufficient number of training examples is expensive. In addition, the properties of the desired summaries depend largely on the domain of the input documents; therefore many training samples of different domains must be made to provide ensure adequate coverage. For the example of contact center dialogue summarization, dialogues in the contact center of a bank and those in the center of an internet service provider should differ quite a bit. Therefore, training samples must be made for both domains which incurs a lot of cost.

To solve this problem, in this chapter, we leverage a domain adaptation technique which uses training samples whose domain is the same as that of the input documents and those whose domains are different from that of the input documents. For example, to summarize the contact center dialogues in the bank domain, the summarizer

uses training samples in that domain and those in other domains, such as internet service provider. We adopt the Augmented Space Method [19], a well-known domain adaptation method, to implement our approach.

We perform experiments to validate its efficacy. Our proposed method surpasses the well-known supervised approach; when training samples from different domains exist, our experiments show that domain adaptation yields the best results.

This chapter is organized as follows: Section 5.2 runs over related work strongly related to this chapter. Section 5.3 elucidates our summarization model. Section 5.4 describes the domain adaptation method that can estimate parameters from multiple domains. Section 5.5 describes a decoding algorithm. Section 5.6 shows our evaluation experiments. We conclude this chapter in Section 5.7.

5.2 Related Work

Some papers have presented methods for summarizing contact center dialogues [13, 30, 29]. Byrd et al. suggested the use of some heuristic rules to summarize contact center dialogues [13]. Higashinaka et al. train HMMs that detect characteristic utterances in the target domains and then uses these HMMs to summarize dialogues by labeling utterances [30, 29]. In contrast, our proposal is the only one to leverage the training samples of different domains.

Although target documents are not contact center dialogues, as the closest work to this chapter, Sandu et al. leverage references of meeting speeches to summarize threads of e-mails [71]. However, their experiment showed that the conventional supervised learning, which uses only training samples whose domain matches that of the target documents, wins against domain adaptation methods. They said that this result is due to the wide difference between the properties of e-mails and meeting speeches. In contrast, we show that our proposed approach works well in the task of contact center dialogue summarization, though we also show that adapting largely different training samples is difficult as they pointed out.

5.3 Summarization Model

In this chapter, we denote a set of sentences to be summarized by $\mathbf{x}$ and its subset that meets the given length of summary by $\mathbf{y}$. Also, we denote an objective function by the

Table 5.1: Features for sentences.

Features	**Value**
Normalized sentence position	[0,1]
# of tokens in sentence	Integer
# of words in sentence	Integer

function that maps summary $\mathbf{y}$ to a real number $\mathbf{f}_{\mathbf{x},\mathbf{w}} : \mathbf{y} \mapsto \mathbb{R}$ under the given sentences to be summarized $\mathbf{x}$ and parameter vector $\mathbf{w}$. In this setting, the summarization problem can be described as follows:

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \subseteq \mathbf{x}}{\operatorname{argmax}} \mathbf{f}_{\mathbf{x},\mathbf{w}}(\mathbf{y}) \tag{5.1}$$

$$\text{s.t.} \quad \text{length}(\mathbf{y}) \leq L$$

where length is a function that returns the length of summary $\mathbf{y}$, L is the maximum summary length.

Some previous work has shown the efficacy of the objective function that scores sentences and words separately for speech summarization [29, 82]. We adopt this kind of objective function and define our objective function as follows:

$$\mathbf{f}_{\mathbf{x},\mathbf{w}}(\mathbf{y}) = \sum_{x_i \in \mathbf{y}} \mathbf{u}^\top \phi(x_i) + \sum_{z_j \in \mathbf{y}} \mathbf{v}^\top \psi(z_j) \tag{5.2}$$

where x_i is the i th sentence present in summary $\mathbf{y}$, z_j is the j th word present in summary $\mathbf{y}$. $\mathbf{u} \in \mathbb{R}^{d_u}$ and $\mathbf{v} \in \mathbb{R}^{d_v}$ are parameter vectors for sentences and words, respectively. $\phi : x \mapsto \mathbb{R}^{d_u}$ and $\psi : z \mapsto \mathbb{R}^{d_v}$ are feature functions for sentences and words, respectively. We show features for sentences in Table 5.1 and features for words in Table 5.2.

Two terms in Eq. 5.2, $\sum_{x_i \in \mathbf{y}} \phi(x_i)$ and $\sum_{z_j \in \mathbf{y}} \psi(z_j)$ can be represented together as $\Phi(\mathbf{x}, \mathbf{y})$, also $\mathbf{u}$ and $\mathbf{v}$ can be merged as $\mathbf{w}^\top = \langle \mathbf{u}^\top, \mathbf{v}^\top \rangle$. Hence our objective function can be represented as $\mathbf{f}_{\mathbf{x},\mathbf{w}}(\mathbf{y}) = \mathbf{w}^\top \Phi(\mathbf{x}, \mathbf{y})$, a linear model.

Table 5.2: Features for words.

Features	Value
Surface of word	$\{0, 1\}$
POS of word	$\{0, 1\}$
Word frequency in input document	Integer
# of sentences containing word	Integer

5.4 Parameter Estimation with Augmented Space Method

In this section we propose a method that applies domain adaptation to training samples of multiple different domains to estimate parameter vector $\mathbf{w}$. First, we describe the domain adaptation method and then the algorithm used to estimate parameter vector $\mathbf{w}$.

5.4.1 Domain Adaptation with Augmented Space Method

We start with N training samples of the documents that belong in the domain that we want to summarize $\{(\mathbf{x}_j^t, \mathbf{y}_j^t)\}_{j=1}^N$ as training samples of the *target domain*. We also refer to M training samples whose domains are different from target domain $\{(\mathbf{x}_i^s, \mathbf{y}_i^s\}_{i=1}^M$ as training samples of the *source domain*.

To leverage training samples in the source domain for learning in the target domain, we adopt the Augmented Space Method (ASM) [19]. The method can be used independently of the learning method and is easy to implement. The method has shown efficacy in the sequential tagging problem [19].

ASM expands feature vector $\Phi(\mathbf{x}, \mathbf{y})$ as follows:

$$\Phi^s(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}, \mathbf{y}), \Phi(\mathbf{x}, \mathbf{y}), \mathbf{0} \rangle$$
$$\Phi^t(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}, \mathbf{y}), \mathbf{0}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

where Φ^s is an expanded feature vector of the source domain examples, Φ^t is an expanded feature vector of the target domain examples. The training samples in the source domain are expanded to Φ^s, and the training samples in the target domain are expanded to Φ^t. If the original feature vector has n dimensions and there are training

samples of k source domains, the method expands the feature vector to $n \times (k+1)$ dimensions. The expanded feature vector consists of n dimensions that are shared between all domains, n dimensions for one domain, and $(n-1) \times k$ dimensions containing all zero elements. Although the above equation is for just two domains, target domain and source domain, the method can be easily expanded to the case that there are two or more source domains. We denote the expanded samples Φ^s and Φ^t by Φ' for simplicity. We also denote the expanded parameter vector by $\mathbf{w}'$.

5.4.2 Structured Learning

In this section we explain our method to estimate parameter vector $\mathbf{w}'$. We adopt structured learning to determine the vector. Instead of learning the probabilities that indicate whether individual sentences and words are included in the summary, we learn the fitness of a summary as a set of sentences and words. We adopt the Online Passive-Aggressive Algorithm [18] to estimate parameter vector $\mathbf{w}'$ from the training samples. Since the algorithm is online, when learning parameter vector $\mathbf{w}'$ it is updated iteratively by solving the following equation:

$$\mathbf{w}'^{new} = \underset{\mathbf{w}'}{\operatorname{argmin}} \frac{1}{2}||\mathbf{w}' - \mathbf{w}'^{old}||^2, \tag{5.3}$$

$$\text{s.t. } \mathbf{w}'^{\top}\Phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}'^{\top}\Phi(\mathbf{x}_i, \hat{\mathbf{y}}) \geq \ell(\hat{\mathbf{y}}; \mathbf{y}_i),$$

where $\mathbf{w}'^{old}$ is the parameter vector before update, $\mathbf{w}'^{new}$ is the parameter vector after update. ℓ is a loss function.

As the loss function, we use ROUGE [47]. From among the ROUGE variants, we use ROUGE-1. The loss function is defined as follows:

$$\ell(\hat{\mathbf{y}}; \mathbf{y}_i) = 1 - \text{ROUGE-1}(\hat{\mathbf{y}}; \mathbf{y}_i),$$

where, as mentioned in Section 3.3.1, ROUGE-1$(\hat{\mathbf{y}}; \mathbf{y}_i)$ is a function that returns a real number from 0 to 1, taking a summary, $\hat{\mathbf{y}}$, and its reference, $\mathbf{y}_i$, as arguments. By incorporating ROUGE in the loss function, the parameter vector is strongly updated when the ROUGE score of a summary is low. Therefore, it is expected that the parameter vector will be sensitive to ROUGE score.

When training, we use the 1-best solution to update the parameter vector. The solution is computed by the algorithm in Figure 3 as with the decoding process.

5.5 Decoding

In this section, we explain the decoding algorithm to generate a summary in Algorithm 3.

Algorithm 3 Decoding Algorithm

$\mathbf{y} \leftarrow \mathbf{0}$
$\mathbf{d} \leftarrow \mathbf{x}_i$
while $\mathbf{d} \neq \emptyset$ **do**
  $\hat{y} = \underset{y \in \mathbf{d}}{\operatorname{argmax}} \frac{\mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, \mathbf{y} \cup y) - \mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, \mathbf{y})}{\text{length}(y)}$
  if $\text{length}(\mathbf{y} \cup \hat{y}) \leq L$ and $\mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, \mathbf{y} \cup \hat{y}) - \mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, \mathbf{y}) \geq 0$ **then**
    $\mathbf{y} = \mathbf{y} \cup \hat{y}$
  end if
  $\mathbf{d} = \mathbf{d} \setminus \hat{y}$
end while
$y^* = \underset{y \in \mathbf{x}_i}{\operatorname{argmax}} \left\{ \mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, y) : \text{length}(y) \leq L \right\}$
$\mathbf{y} = \underset{\mathbf{y} \in \{\mathbf{y}, y^*\}}{\operatorname{argmax}} \mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, \mathbf{y})$
return $\mathbf{y}$

As with notions introduced in previous sections, let $\mathbf{x}_i$ be a set of input sentences. Let $\mathbf{y}$ be a set of sentences, indicating a summary. Let $\mathbf{0}$ be an empty set. Let $\mathbf{d}$ be a set of sentences, preserving senetnces which are not processed yet.

As shown in Algorithm 3, the algorithm is basically a greedy algorithm; it adds sentence $\hat{y}$ to summary $\mathbf{y}$ iteratively. First, the algorithm takes a set of input sentences $\mathbf{x}_i$, feature vector Φ' and learned parameter vector $\mathbf{w}'$ as inputs. $\mathbf{y}$ is initialized and set as an empty set, $\mathbf{0}$. $\mathbf{x}_i$ is copied into $\mathbf{d}$. Then, the sentence $\hat{y}$ that increases the score of the summary $\mathbf{y}$ most at each iteration and satisfy the maximum summary length, L, is added to the summary $\mathbf{y}$. The increase is calculated as $\mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, \mathbf{y} \cup \hat{y}) - \mathbf{w'}^{\top}\Phi'(\mathbf{x}_i, \mathbf{y})$ and the value is normalized by the length of the sentence. Regardless of whether the sentence selected as $\hat{y}$ became a part of the summary, it is removed from $\mathbf{d}$ after it was selected as $\hat{y}$. After the all sentences were removed, the resulted summary, $\mathbf{y}$, is compared to a singleton, y^*, which is a sentence that has the highest score among input sentences that satisfy the maximum summary length. Finally, one which has a higher between the resulted summary, $\mathbf{y}$, and the singleton, y^* is output as a final summary.

Table 5.3: Dialogue Domains.

Domain	Topic
FIN	Inquiries to banks and insurance companies.
ISP	Inquiries to internet service providers.
LGU	Inquiries to local government units.
MO	Inquiries to mail-order companies.
PC	Inquiries to computer manufacturers.
TEL	Inquiries to telecommunications companies.

Table 5.4: Statistics of training corpus.

Domain	# of samples	# of utterances	Avg. sum. rate
FIN	59	10377	13.38%
ISP	64	7062	16.29%
LGU	76	8865	21.20%
MO	70	9694	17.38%
PC	56	10088	13.22%
TEL	66	9774	16.79%

The algorithm shown in Algorithm 3 was originally proposed by Khuller et al. [39]. Although it is not certain that the algorithm will always find the exact solution, Khuller et al. showed that the algorithm gives a good approximate solution[1].

5.6 Experiment

In this section we show the efficacy of our proposed method by experiments. We use a corpus consisting of dialogues in six domains. In this experiment, our aim is to confirm the efficacy of domain adaptation.

[1]Khuller et al. showed that the algorithm is a $(1-1/e)$-approximation algorithm [39].

Table 5.5: Statistics of test corpus.

Domain	# of samples	# of utterances	Avg. sum. rate
FIN	60	8863	16.31%
ISP	59	9563	15.63%
LGU	56	7934	18.43%
MO	47	7305	20.22%
PC	44	11772	10.01%
TEL	41	8069	13.55%

5.6.1 Corpus

We use contact center dialogues as the corpus. To avoid the effects caused by errors in automatic speech recognition systems, these dialogues were manually transcribed, not automatically recognized. Each dialogue consists of utterances of a customer who calls the contact center and the operator who receives the call. Since both customer and operator are Japanese native speakers, transcripts are Japanese. Each dialogue was divided into utterances. We use an utterance as the unit of summarization. Therefore, our summarizer selects a set of utterances that meets the given summary length from an input dialogue consisting of a set of utterances. The references were made by extracting utterances by annotators.

There are six domains in our corpus. We show the main topic of each domain in Table 5.3. As shown in Table 5.3, there are various topics in dialogues. Our aim is to summarize each domain by leveraging the training samples of the target and other domains.

We show the statistics of our corpus in Table 6.2. # of samples is the number of samples contained in training or test set of each domain, # of utterances is the number of utterances contained in training or test set of each domain. Avg. sum. rate is the average summarization rate that is calculated by dividing the reference length by the average length of original input documents. The number of references is 250 characters in both training and test set.

Table 5.6: ROUGE-1 results.

Domain	Method (1)	Method (2)	Method (3)	Method (4)
FIN	0.437	0.558	0.551	**0.584** *
ISP	0.465	0.525	0.508	**0.543**
LGU	0.466	**0.514**	0.506	0.500
MO	0.563	0.632	0.601	**0.635**
PC	0.215	**0.394**	0.330	0.367
TEL	0.421	0.390	**0.457**	0.418

5.6.2 Setting

Following the previous work on domain adaptation [19], we also compare the following four methods.

1. **Source domain only**. When learning, a learner uses only training samples whose domains are different from the domain of documents to be summarized. When test examples in FIN domain are summarized, the parameter vector is trained using ISP, LGU, MO, PC and TEL domains.

2. **Target domain only**. When learning, for each domain, the learner uses only training samples whose domain is the same as the domain of documents to be summarized. When test examples in FIN domain are summarized, the parameter vector is trained using training samples in FIN domain. Hence this situation is the same as usual supervised learning.

3. **All domains**. When learning, a learner uses training samples of all domains without distinction. When test examples in FIN domain are summarized, the parameter vector is trained using training examples in FIN, ISP, LGU, MO, PC and TEL domains.

4. **Domain Adaptation**. Proposed method. We adopt the domain adaptation method mentioned above.

We call these methods Method (1)-(4), respectively. The result we want to clarify is whether our proposed approach, Method (4), is superior to usual supervised approach,

Method (2). If this is true, contact center dialogue summarization systems should use a domain adaptation technique.

We used ROUGE-1 [47] to evaluate our method.

5.6.3 Results and Discussions

We show a result of our experiment in Table 5.6. The values in each row and column are the values of the ROUGE score for the corresponding methods and domains, respectively. Bold-faced values are the highest scores in each domain.

As shown in Table 5.6, our proposed method achieves the best score in three of six domains, FIN, ISP and MO. Method (4) is superior to the usual supervised approach, Method (2), in four of six domains. In FIN domain, the ROUGE scores of Method (4) surpassed those of Method (2) by a statistically significant margin at the 95% level.

In FIN domain, Method (4), achieved the best result, followed by Method (2), Method (3) and Method (1). This trend is also observed in ISP and MO domains. In these domains, supervised learning method, Method (2), achieves good results, though the domain adaptation method leveraged the training samples in different domains to improve usual supervised learning.

In LGU and PC domains, Method (2) surpassed Method (4). This result suggests that their source samples have no training samples that are similar to and that are useful as samples for the target domain. Actually, dialogues in PC domain are particularly troublesome, because customers frequently raise technical problems about their computers and hence the dialogues often become long, as its Avg. Sum. Rate shows. In such long dialogues, important utterances are at the end of the dialogues, while in other domains important utterances are at the front.

In TEL domain, Method (2) failed to match Method (1). That is, training samples in source domains are more useful than training samples in the target, TEL domain. This result implies that in the TEL domain there is some kind of gap between training and test examples. Previous work [19] reported that if a method that leveraged only samples in a source domain beat a method that used only samples in the target domain, domain adaptation was not effective. This result confirms that conclusion.

It is important to find a way to test whether transfer learning can be effective before applying trasfer learning to real data, because transfer learning is not always effective as Table 5.6 showed. As explained above, previous work originally proposed the Augmented Space Method [19] reported that if a method that leveraged only samples in

a source domain beat a method that used only samples in the target domain, domain adaptation was not effective. Therefore, a straightforward way to do testing that is to prepare some test examples and to test the performance in the both settings where the learner uses the training examples only in the source domain and only in the target domain.

5.7 Conclusion

In this chapter we proposed a method to improve the quality of extractive summarization by making use of training examples whose domains are different from that of the target domain. We adopted the Augmented Space Method to adopt samples from source domains to the target domain and validated its efficacy by experiments. By our experiments, our proposed domain adaptation approach achieved the best results.

An immediate research direction is to leverage training samples in other than contact center dialogues, such as news documents. There are a lot of training samples in news domains, and hence using them is promising.

Chapter 6

Fast Decoding with Lagrange Heuristic

6.1 Introduction

Many text summarization studies in recent years formulate text summarization as the maximum coverage problem [23, 84, 92, 75, 27, 60, 59, 29, 62, 61, 91]. The maximum coverage model, based on the maximum coverage problem, generates a summary by selecting sentences to cover as many information units (such as unigrams and bigrams) as possible. Takamura and Okumura [92, 75] and Gillick and Favre [27] demonstrated that the maximum coverage problem offers great performance as a text summarization model. Unfortunately, its potential is hindered by the fact that it is NP-hard (Khuller et al., 1999). There is little hope that a polynomial time algorithm for the problem exists. In the experiment as shown in Section 6.6, it took more than 1 week to summarize 30 input document set. It took over 8 hours on average to summarize one input document set, obviously it is impractical.

Another theoretical framework for text summarization, the knapsack problem, avoids trying to cover unigrams or bigrams, and instead emphasizes the selection of important sentences under the constraint of summary length. The knapsack problem can be solved by a dynamic programming algorithm in pseudo-polynomial time (Korte and Vygen, 2008). However, the knapsack model, a text summarization model based on the knapsack problem, scores each sentence independently. While it can easily maximize the sum of their scores, it threatens to generate redundant summaries unlike the maximum coverage model.

Compared to the knapsack model, the maximum coverage model has higher perfor-

Table 6.1: Advantage of Redundancy-constraint knapsak model.

	Summary quality	Decoding speed
Redundancy-Constrained Knapsack Model	◯	◯
Knapsack Model	×	◯
Maximum Coverage Model	◯	×

mance as a summarization model. However, the knapsack model can be decoded far faster than the maximum coverage model.

To tackle this trade-off between summary quality and decoding speed, we propose a novel text summarization model, the redundancy-constrained knapsack model. Starting with the advantage of the knapsack model, it uses dynamic programming to achieve optimization in pseudo-polynomial time. We add to it a constraint that curbs summary redundancy.

Although this constraint can suppress summary redundancy, finding the optimal solution again becomes a challenge. To ensure that our proposed model can find good approximate solutions, we turn to the Lagrange heuristic [28, 78]. This is an algorithm that finds a feasible solution from the relaxed, infeasible solution induced by Lagrange relaxation. It is known to be effective in finding good approximate solutions for the set covering problem [28, 78]. To be more specifically, the constraint suppressing redundancy is relaxed and is integrated into the objective function. Then, the optimal solution, i.e. relaxed solution, of the objective function can be decoded by the dynamic programming. Finally, the feasible solution is obtained from the relaxed solution.

The advantage of the redundancy-constrained knapsack model against the knapsack model and maximum coverage model is shown in Table 6.1. The proposed summarization model has both the summary quality of the maximum coverage model and the decoding speed of the knapsack model.

This chapter is organized as follows. In Section 6.2, we describe related work. In Section 6.3, we compare the maximum coverage model and knapsack model. In Section 6.4, we elaborate our proposed model. In Section 6.5, we introduce the algorithm that finds a good approximate solution for our proposed model. In Section 6.6, we show the results of experiments conducted to evaluate our proposal. In Section 6.7, we conclude this chapter.

6.2 Related Work

In multi-document summarization, handling redundancy in summary is an important question [50, 88, 58] . When generating a summry from multiple documents, if two or more documents include similar information, both information could be included in the summary. In cosideration of multi-document summarization, it is undesireble that the similar information is included in the summary twice, and hence the problem must be handled.

To tackle this problem, the text summarization model based on the maximum coverage problem was proposed by Filatova and Hatzivassiloglou [23]. The maximum coverage model takes a set of sentences and the maximum summary length as inputs. Each sentences in the set include some concepts, and each concept has importance according to its property. A concept is an unit that can be extracted from sentences, such as an unigram or bigram. Sentences have the length according to its number of words or characters. The optimal solution of the maximum coverage problem is a set of sentences that has the highest sum of scores of concepts in the set and does not exceed the maximum summary lengthamong the possible set of sentences from an input. When summing up scores of the concepts in a set of sentences, the same concept is summed up only once. For example, even if a set of sentences includes the same concept thrice, in the maximum coverage model, the sum of the scores of these concepts are equal to the sum of scores of the set that includes that concept once. This property plays an important roll in multi-document summarization, i.e. the better summary includes diverse information.

Such property of the maximum coverage problem, where the better solution is to include more diverse information, is natural to be a model of multi-document summarization. There are plenty of previous work leveraging the maximum coverage problem to model multi-document summarization. Filatova and Hatzivassiloglou and used tf-idf as importance and solved their model by a greedy algorithm [39], set a word as a concept. Yih et al. solved the model by a stack decoder [84]. Takamura and Okumura [92, 75] and Gillick and Favre [27] formulated the model as Integer Linear Programming (ILP) and solved the model using a branch-and-bound method. While the above work targets newswire articles as inputs, there are some work that targets other fields. Higashinaka et al. summarized logs of contact centers [29].

The maximum coverage problem is NP-hard [39], and hence it is important to locate an approximate solution quickly. The solution located by simple search algorithms, such as the greedy method, is not always good, but it can find a solution quickly. The

solution of the greedy method achieves $(1-1/e)/2$ of the value of the objective function [39] . Takamura and Okumura reproduced this to multi-document summarization and reported its results [92]. More complex algorithms, such as the stack decoder [38, 84] can find the better solution than the greedy algorithm, it take more time to locate a solution. The branch-and-bound method can find the optimul solution, the time to locate the solution rapidly increases according to the size of the problem.

In contrast to multi-document summarization, redundancy matters relatively little in single document summarization, and hence it can use the method not to take into account redundancy. Single-document summarization can be formulated as the knapsack problem [53, 96] . The knapsack problem takes a set of sentences and the maximum summary length as an input. Unlike the maximum coverage problem, in the case of the knapsack problem, each sentence among the input sentences has importance directly. The optimal solution of the knapsack problem is the set where the sum of the importance of the sentences among the set is maximum among possible sets which do not exceed the maximum summary length. The optimal solution of the knapsack problem can be located with dynamic programming knapsack algorithm in pseudo-polynomial time [41], and hence the optimal solution can be found out quickly. One major drawback of the knapsack problem as a model of multi-document summarization is redundancy; the knapsack model defines importance directly on the sentences and maximizes the sum of importance, and hence it lacks a way to reduce the redundancy in the summary. This drawback is particularly severe in the context of multi-document summarization.

As mentioned above, the maximum coverage problem is suitable for multi-document summarization, it takes a long time to locate to locate a good solution. In contrast to the maximum coverage problem, the knapsack problem is not suitable for multi-document summarization, but the optimal solution is easily located. The proposal described in this chapter solves this trade-off by adding the constraint reducing the redundancy to the knapsack problem.

6.3 Maximum Coverage Model and Knapsack Model

In this section, we compare the model based on the maximum coverage problem, the maximum coverage model, with the model based on the knapsack model, the knapsack model.

6.3.1 Maximum Coverage Model

We consider there are n input sentences containing m unique information units, such as unigrams and bigrams, or some information can be extracted from the sentences. Let $\mathbf{x}$ be a binary vector whose element x_i is a decision variable indicating whether sentence i is contained in the summary. If sentence i is contained in the summary, $x_i = 1$. Let $\mathbf{z}$ be a binary vector whose element z_j is a decision variable indicating whether information unit j is contained in the summary. If information unit j is contained in the summary, $z_j = 1$. Let w be a vector whose element w_j indicates the importance of information unit j . Let $\mathbf{A}$ be a matrix whose element $a_{j,i}$ indicates the number of information units j , contained in sentence i . If sentence i contains two information units j, $a_{j,i} = 2$. Let l be a vector whose element l_i indicates the length of sentence i . Let K be the maximum summary length desired. In this setting, the maximum coverage model is formulated as follows:

$$\max_{\mathbf{z}} \quad \mathbf{w}^\top \mathbf{z} \tag{6.1}$$

$$s.t. \quad \mathbf{A}\mathbf{x} \geq \mathbf{z} \tag{6.2}$$

$$\mathbf{x} \in \{0,1\}^n \tag{6.3}$$

$$\mathbf{z} \in \{0,1\}^m \tag{6.4}$$

$$\mathbf{l}^\top \mathbf{x} \leq K \tag{6.5}$$

The objective function is Equation 6.1 , and Equation 6.2 [1] to 6.5 are constraints. As indicated in Equation 6.4, an element of $\mathbf{z}$ is either 0 or 1. If concept j is in the summary, z_j is 1 and its importance w_j is added to the objective function. Although all concepts should be included in the summary to maximize the objective function, the constraint of maximum summary length disallow it. As indicated in Equation 6.2, at least one sentence containing j is must be contained in the summary to contain concept j in the summary. If sentence i is in the summary, l_i is added to the left part of Equation 6.5. As indicated in Equation 6.5, the sum of the length of sentences in the summary must be within K. Equation 6.3 indicates that an element of $\mathbf{x}$ is either 0 or 1, and hence each sentence is contained in the summary only once.

From the objective function and constraints explained above, the optimal solution of the maximum coverage model is located by afinding out the combination of sentences

[1] In this chapter the magnitute relationship between two n-dimentional vector $\mathbf{a}$ and $\mathbf{b}$ holds if and only if $a_i \geq b_i$ $(\forall i)$, where a_i is an element of $\mathbf{a}$ and b_i is an element of $\mathbf{b}$, holds.

maximizing the sum of importance of them among the possible sets. However, even though some combinations do not satisfy the length constraint, the number of possible sentence sets is 2^n . Hence it is difficult to enumerate all combinations.

6.3.2 Knapsack Model

Next, we describe the knapsack model. Let $\mathbf{x}$ be a binary vector whose element x_i is a decision variable indicating whether sentence i is contained in the summary. If sentence i is contained in the summary, $x_i = 1$. Let $\mathbf{z}$ be a binary vector whose element z_j is a decision variable indicating whether information unit j is contained in the summary. If information unit j is contained in the summary, $z_j = 1$. The knapsack model is formulated as follows:

$$\max_{\mathbf{z}} \quad \mathbf{w}^\top \mathbf{z} \tag{6.6}$$

$$s.t. \quad \mathbf{A}\mathbf{x} = \mathbf{z} \tag{6.7}$$

$$\mathbf{x} \in \{0,1\}^n \tag{6.8}$$

$$\mathbf{z} \in (\mathbb{N}^0)^m \tag{6.9}$$

$$\mathbf{l}^\top \mathbf{x} \leq K \tag{6.10}$$

where $\mathbb{N}^0$ is the natural numbers above and including 0. Equation 6,6, 6.8 and 6.10 are the same as the maximum coverage model, Equation 6.7 and 6.9 are different model.

In Equation 6.4, an element of vector $\mathbf{z}$ is either 0 or 1, in Equation 6.9, an element of the vector is the natural numbers above and including 0. In the maximum coverage model, the importance of each concept is counted only once on the objective function no matter how many times the concept is included in the summary. In contract to the maximum coverage model, the knapsack model counts each occurrence of the same concept on the objective function. From this property, the knapsack model is likely to generate a redundant summary, and hence the performance of the knapsack model in multi-document summarization is not good [53] .

Unlike Equation 6.2, Equation 6.7 reflects the number of concepts to vector $\mathbf{z}$ directly. For example, if sentence 1 contains concept 5 twice, $a_{5,1}$ is 2. If sentence 2 contains concept 5 once, $a_{5,2}$ is 1. If both sentence 1 and 2 are selected as the summary (i.e. $x_1 = 1$ and $x_2 = 1$), Equation 6.7 is calculated as $z_5 = a_{5,1} \times x_1 + a_{5,2} \times x_2 = 2 \times 1 + 1 \times 1 = 3$, and hence there are three concepts 5 in the summary.

6.4 Redundancy-Constrained Knapsack Model

As explained in the preceding section, the reason why the maximum coverage model is robust over redundancy lies in Equation 6.4, and the reason why the knapsack model is sensitive to redundancy lies in Equation 6.9. Therefore, the redundancy can be curbed by modifying Equation 6.9, i.e. reducing the redundancy by controlling the number of a concept in the summary directly.

6.4.1 Introduce Redundancy Constrait to Knapsack Model

The knapsack model with the constraint that controls the number of occurrence of the concepts in the summary is shown below:

$$\max_{\mathbf{z}} \quad \mathbf{w}^{\top}\mathbf{z} \tag{6.11}$$

$$s.t. \quad \mathbf{A}\mathbf{x} = \mathbf{z} \tag{6.12}$$

$$\mathbf{x} \in \{0,1\}^{n} \tag{6.13}$$

$$\mathbf{z} \in \{z_j | \mathbb{N}^0 \cap [0, r_j]\}^{m} \tag{6.14}$$

$$\mathbf{l}^{\top}\mathbf{x} \leq K \tag{6.15}$$

Equation 6.14 indicates that an element of vector $\mathbf{z}$ is the natural numbers that are equal to or higher than 0, and are equal to or lower than r_j. The redundancy in the summary is reduced by vector $\mathbf{r} = (r_1, r_2, ..., r_m)$, which limits the number of occurrence of concepts in the summary. In this chapter, the summarization model described with Equation 6.11 to Equation 6.15 is named the redundancy-constrained knapsack model.

This model is not equal to the maximum coverage model. The maximum coverage model allows the summary to contain the same concept more than once. However, the model counts the importance of concepts only once on the objective function. In contrast to the maximum coverage model, the knapsack model limits the number of occurrence of the concepts in the summary directly.

For example, if sentence 1 contains concept 5 twice (i.e. $a_{5,1} = 2$) , sentence 2 contains concept 5 once （ $a_{5,2} = 1$) and $r_5 = 2$, sentence 1 and 2 cannot be selected simultaneously in the summary. Concept 5 can be contained in the summary only up to twice, it is contained thrice if both sentence 1 and 2 are included in the summary. The

maximum coverage model allows that combination unless it violates the length constraint, but the redundancy-constrained knapsack model disallow the combinations violating Equation 6.14. As just described, the redundancy-constrained knapsack model limits the number of occurrence of the concepts directly.

By adding the constraint controlling the occurrence defined in Equation 6.14 the redudancy in the summary can be reduced, the optimal solution of the model cannot be located in pseudo-polynomial time by the dynamic programming knapsack algorithm [2] . The redundancy-constrained knapsack model is solved by the dynamic programming knapsack algorithm [3] , to solve the model by the algorithm, the number of occurrence of concepts in the summary must be saved in the search process. There are many combinations of occurrence of the conpepts. Hence, it expands the search space and rules out quick inference.

6.4.2 Lagrange Relaxation for Redundancy Constraint

By removing the redundancy constraint described in Equation 6.14, the original knapsack model is obtained. Making the redundancy constrained knapsack model tractable, we count on the Lagrange relaxation [41]; Equation 6.14 is relaxed with the Lagrange relaxation and the redundancy constrained with vector $\mathbf{r}$ is incorporated in the objective function as follows:

$$\max_{\mathbf{z}} \quad \mathbf{w}^\top \mathbf{z} + \lambda(\mathbf{r} - \mathbf{z}) \tag{6.16}$$

$$s.t. \quad \mathbf{A}\mathbf{x} = \mathbf{z} \tag{6.17}$$

$$\mathbf{x} \in \{0,1\}^n \tag{6.18}$$

$$\mathbf{z} \in (\mathbb{N}^0)^m \tag{6.19}$$

$$\lambda \in (\mathbb{R}^+)^m \tag{6.20}$$

$$\mathbf{l}^\top \mathbf{x} \leq K \tag{6.21}$$

where $\lambda = (\lambda_1, \lambda_2, ..., \lambda_m)$ is a vector of non-negative Lagrange multipliers. The model described by Equation 6.16 to 6.21 is the same as the knapsack model excepting

[2] More preciously, the order of solving the model by that algorithm is pseudo-polynomial in n , but exponential in m . Usually the number of concepts, m is far larger than the number of sentences, n, i.e $m \gg n$, and hence it is virtually exponential.

[3] Of course, it can also be solved by an ILP solver. In this chapter, the optimal solution is located by using the ILP solver to examine the approximation performance of the algorithm.

the second argument of Equation 6.16, $\lambda(\mathbf{r}-\mathbf{z})$, and Equation 20. This Lagrange relaxation problem penalize the objective function through λ_j when the number of concept j in the summary, z_j, exceeds the redundancy constraint, r_j. For example, if concept 5 is included in the summary thrice, but vector $\mathbf{r}$ limits the occurrence of concept 5 to be up to twice, and Lagrange multipler λ_5 is 1, the value of the objective function drops as $\lambda_j(r_5 - z_5) = 1(2-3) = -1$. Lagrange multiplers λ is adjusted by solving the Lagrange dual problem of this relaxation problem, $L(\lambda) = \min_\lambda \{\max_{\mathbf{z}} \mathbf{w}^\top \mathbf{z} + \lambda(\mathbf{r}-\mathbf{z})\}$.

As explained above, the proposal in this chapter is to acquire a good solution as a summary of multi-document summarization even with the dynamic programming knapsack algorithm, by decreasing the importance of concepts causing redundancy through adjust Lagrange multipliers λ. A specific way to decode the proposed model will be elaborated in next section.

6.5 Decoding

To decode the redundancy-constrained knapsack model, vector of the Lagrange multipliers λ must appropriately be adjusted. If the value of λ is set appropriately, a good approximate solution can be obtained by solving the model with the dynamic programming knapsack problem. λ is adjusted by solving the Langrange dual problem explained in the preceding section. Since the maximization is nested inside the minimization in the Lagrange dual problem, it is difficult to optimize it. However, Umetani et al. showed that with the subgradient method [41] a good approximate solution can quickly be obtained by the subgradient method [78]. The subgradient method updates the value of λ iteratively.

The problem is to decide to what extent the value of λ_j should be changed at one time; It is expected that changing the value substantially at one time shorten the decoding time, but small changes of the value are needed to find the optimal solution. To adjust the Lagrange multipliers, the Langrange heurisic is used [28]. The Lagrange heuristic updates the Langrange multipliers according to the gap between the upper bound and lower bound.

The upper bound is the solution of the Langrange relaxation problem at an iteration. Since the redundancy constraint is relaxed, the value of the objective function of the relaxed problem is obviously higher than the value of the objective function of the original problem without relaxation. In the process of updating the Langrange multipliers, the solution of the relaxation problem is approaching to the feasible solution from the

infeasible solution, which violates the constraint, decreasing the value of the objective function.

The lower bound is the feasible solution obtained by some sort of heurisics. In this chapter, the feasible solution is recoverd by the way proposed by Haddadi [28] and the greedy method [39]. The specific way to recover the feasible solution will be elaborated in next section.

The specific decoding algorithm with the Lagrange heuristic is shown in Algorithm 1. The basic steps of the algorithm are shown below:

1. Initialize vector of the Lagrange multipliers λ .
2. Iterate following steps at times set.
   (a) Locate the optimal solution of Equation 6.16 by the dynamic programming knapsack algorithm.
   (b) If the solution located at (a) satisfy the constraits, set it as the lower bound and go to 3. If not, find out the feasible solution by the heuristic.
   (c) If the feasible solution found out at (b) exceeds the current lower bound, update the lower bound.
3. Output the lower bound.

α is a parameter setting the stepsize of λ . An element of vector $\mathbf{s} = (s_1, s_2, ..., s_n)$, s_i indicate the importance of sentence i . The importance of each sentence is calculated by the function *sentence* . Function $dpkp$ is the dynamic programming knapsack algorithm. The detail of the algorithm is shown in Algorithm 2. b_l and b_u are the lower bound and upper bound of the objective function, respectively. They are used to adjust the step size of λ . Function *score* calculates the sum of importance of summary $\mathbf{x}$. Function *count* returns the number of concepts in summary $\mathbf{x}$, $\mathbf{z}$. $\mathbf{x}_l$ is a solution corresponding the lower bound, b_l .

Subgradient vector of the Lagrange relaxation problem, $\mathbf{d} = (d_1, d_2, ..., d_m)$, is as follows:

$$d_j = r_j - z_j \tag{6.22}$$

Based on the updating formula for the Lagrange-relaxed set covering problem suggested by Umetani [78], the Lagrange multipliers are updated along with the following formula:

Algorithm 4 Decoding Algorithm with Lagrange Heuristic

$\lambda \leftarrow \mathbf{0},\ \mathbf{s} \leftarrow \mathbf{0},\ \mathbf{x} \leftarrow \mathbf{0},\ \mathbf{z} \leftarrow \mathbf{0}$
for $t = 1$ to T **do**
  $\mathbf{s} \leftarrow sentence(\mathbf{A},\ \lambda,\ m,\ n,\ \mathbf{w})$
  $\mathbf{x} \leftarrow dpkp(K,\ \mathbf{l},\ n,\ \mathbf{s})$
  if $score(\mathbf{A}, m, n, \mathbf{x}, \mathbf{w}) \leq b_u$ **then**
    $b_u \leftarrow score(\mathbf{A}, m, n, \mathbf{x}, \mathbf{w})$
  end if
  $\mathbf{z} \leftarrow count(\mathbf{A},\ m,\ n,\ \mathbf{x})$
  if $\mathbf{z}$ violates $\mathbf{r}$ **then**
    $\mathbf{x} \leftarrow heuristic(\mathbf{A}, K, \mathbf{l}, m, n, \mathbf{w})$
    if $score(\mathbf{A}, m, n, \mathbf{x}, \mathbf{w}) \geq b_l$ **then**
      $b_l \leftarrow score(\mathbf{A}, m, n, \mathbf{x}, \mathbf{w})$
      $\mathbf{x}_l \leftarrow \mathbf{x}$
    end if
    $\lambda \leftarrow update(\alpha,\ b_l,\ b_u,\ \lambda,\ m,\ \mathbf{r},\ \mathbf{z})$
  else
    return $\mathbf{x}$
  end if
end for
return $\mathbf{x}_l$

$$\lambda_j^{new} \leftarrow \max\left(\lambda_j^{old} + \alpha \frac{b_u - b_l}{||\mathbf{d}||^2}(z_j - r_j), 0\right) \tag{6.23}$$

α is a parameter adjusting the range of update. Basically, the formula updates the Lagrange multipliers significantly along with sub-gradient vector when the difference between upper bound b_u and lower bound b_l .

6.5.1 Recovering Feasible Solution with Greedy Algorithm

The Lagrange heuristic recovers a feasible solution from an infeasible solution using some sort of heuristic. We recover the feasible solution with following steps:

1. Remove the sentence from the summary, which has the lowest score among sentences containing concepts that violates redundancy constraints.

2. It the summary satisfies the constraints, generate a sub-problem consisting of sentences not contained in the summary and the difference in length between K and the length of summary satisfying constraints, and then solve the sub-problem using the greedy algorithm. If not, go to step (1).

For example, if the maximum length of summary is 300 characters and the length of summary is 200 characters after removing the sentences violating the constraints, there is 100 characters room for sentences not selected. A feasible solution is found out by packing the sentences not selected in the room of 100 characters using the greedy algorithm [39].

6.5.2 Dynamic Programming Knapsack Algorithm

The dynamic programming knapsack algorithm is used to decode the knapsack mode. The algorithm is shown in Algorithm 2.

While the dynamic programming knapsack algorithm is running, the states in process of dynamnic programming are preserved on the table with $n+1$ rows and $K+1$ columns.

Element $T[i][k]$ on Table T preserves the optimal solution at the case where sentences 1 to sentence i and the maximum length of summary, k, are given. Element $U[i][k]$ on

Algorithm 5 Dynamic Programming Knapsack Algorithm

```
1:  x ← 0
2:  for k = 0 to K do
3:      T[0][k] ← 0
4:  end for
5:  for i = 1 to n do
6:      for k = 0 to K do
7:          T[i][k] ← T[i−1][k]
8:          U[i][k] ← 0
9:      end for
10:     for k = l_i to K do
11:         if T[i−1][k−l_i] + s_i ≥ T[i][k] then
12:             T[i][k] ← T[i−1][k−l_i] + s_i
13:             U[i][k] ← 1
14:         end if
15:     end for
16: end for
17: k ← K
18: for i = n to 1 do
19:     if U[i][k] = 1 then
20:         x_i ← 1
21:         k ← k − l_i
22:     end if
23: end for
24: return x
```

Table U preserves binary variables, either 1 or 0. If sentence i is contained in the summary at the time when the value of $T[i][k]$ is computed, $U[i][k]$ is 1. If not, $U[i][k]$ is 0. That is, at the end of computing $T[n][k]$, U preserves which sentences are used as the resulted summary in the case where the maximum summary length is k and there are sentences from sentence 1 to sentence n. Therefore, after computing $U[n][k]$, the optimal solution of the knapsack model is can be located by backtracking the path to $U[n][K]$ from $U[0][0]$.

6.6 Experiments

In this section, results of evaluation are reported. Methods explained in Section 6.6.1 are used to generate summaries using corpora explained in Setion 6.6.2 and evaluate these summaries by the evaluation method explained in Section 6.6.3. Parameter estimation for these methods are explained in Section 6.6.4. The results and discussions are shown in Section 6.6.5.

6.6.1 Methods

Following methods were compared.

1. **RCKM** This is a proposed method, and decode the redundancy-constrained knapsack model with an ILP solver. It shows a performance of the optimal solution of the redundancy-constrained knapsack model. As a solver, `lp_solve` [4] was used [5] .

2. **RCLM-LH** This is also a proposed method, and decode the redundancy-constrained knapsack model with the Lagrange heuristic proposed above. This shows a performance of an approximate solution located by the proposed decoding algorithm, the Lagrange heuristic.

3. **MCM** This is a baseline. The maximum coverage model is decoded by the ILP solver.

[4] http://lpsolve.sourceforge.net/5.5/

[5] Of course, other commercial solvers can be used to decode and it is expected that commercial solvers can solve the model more quickly, but they are paid and `lp_solve` is widely used in a research community, and hence it was used to decode.

4. **MCM-GR** This is a baseline. The maximum coverage model is decoded by the greedy method.

5. **KM** This is a baseline. The knapsack model is decoded by the dynamic programming knapsack algorithm.

6. **HUMAN** To clarify the upper bound of performance which can be achieved, calculate the upper bound with multiple references. Among the corpora explained in Section 6.6.2, the review corpus contains four references for each document set, and hence the upper bound is calculated by comparing these references. Among ${}_4C_2 = 6$ combinations from four references, the combination which has the highest ROUGE score, which is the evaluation measure of summarization and will be explained in Section 6.6.3, is used as the upper bound. The TSC-3 corpus only contains one reference for each summary, and hence it can be calculated only in review corpus.

The decoder of **RCKM-LH** , **MCM-GR** and **KM** are implemented with Perl. All programs were run on the computer with two Intel Xeon X5560 (Quad Core) 2.8GHz CPU and 64G byte memory.

6.6.2 Data

The methods described above were evaluated using following two corpora.

1. **TSC-3** The TSC-3 corpus [31] is a corpus used in the shared task of automatic summarization, Text Summarization Challenge 3 [6] and includes the evaluation set for multi-document summarization. Targets of summarization are newswire articles writte in Japanese; these articles were collected from Mainichi shinbun and Yomiuri shinbun. Each evaluation set consists of newswire articles with a specific topic, such as a corporate acquisition and terrorism. The TSC-3 corpus consists of 30 document sets; each document set consists of about 10 documents and the sum of characters in one document sets is about 6,564 characters. The whole corpus contains 352 articles and 3,587 sentences. For each document set, two references, shorter one and longer one, written by humans are given. Hirao et al. reported that that humans read all articles to be summarized and

[6] http://lr-www.pi.titech.ac.jp/tsc/tsc3.html

Table 6.2: Statistics of two corpora.

	# of docs sets	Avg. # of docs per set	Avg. # of sents. per set	Avg. # of charas. per set	Avg. summ. rate
TSC-3	30	11.7	119.6	6,564.1	6.3%
Review	30	15.6	75.8	2,472.4	8.1%

then make a summary of them [31]. The average length of shorter summaries is almost 413 characters, and the summarization rate [7] is almost 6%. The average length of longer summaries is almost 801 characters and the summarization rate is almost 12%. In this experiment, shorter ones are used. Since the length of reference summary is different among the document sets, the same length as the coresponding reference has is given to the model as the maximum summary length when generating a summary. The average input consists of 12 articles with 6,564 characters and the average length of output is 413 characters. The summarization rate is almost 6%.

2. **Review** To evaluate the proposal in a domain other than the domain of newswire articles, review articles were used. Review documents corresponding to 30 restaurants were collected from the Internet website as 30 document sets for multi-document summarization. One set consists of around 15 review articles and the sum of the characters of one document set is 2,472 characters. The whole corpus contains 468 articles and 2,275 sentences. For each document set, four humans made references. Each human read all articles to be summarized and then made a summary for the articles. The maximum summary length is 200 characters for all document sets. The summarization rate is almost 8%. The average input consists of 16 articles and 2,472 characters and the maximum summary length is 200 characters.

The statistics of the corpora is shown in Table 6.2.

[7] The summarization rate is the number calculated by diving the maximum summary length by the sum of the length of input documents [88] . For exapmle, the maximum summary length is 400 characters and the sum of the length of input documents is 8000 characters, the summarization rate is $5\% = \frac{400}{8000}$.

6.6.3 Evaluation Measure

To evaluate our proposal, following two measurement are used.

1. **The quality of summary** To evaluate the quality of summary, ROUGE [47], was used. ROUGE is an automatic evaluation method for document summarization, proposed by Lin [47]. Among variants of ROUGE, ROUGE-1 and ROUGE-2 were used to evaluate. Since Hirao et al. reported that higher correlation could be acquired when only content words [8] were used to calculate ROUGE, only content words were used. To segment a summary into words and identify their pos tags, a morphological analyzer proposed by Fuchi and Takagi [24] was used. A program calculating ROUGE is implemented by the author according to the paper [47]. There are four references for each document set, and hence the highest ROUGE score among the four ROUGE score is used as the ROUGE score.

2. **The decoding speed** The time taken to generate summaries was measured. In both two corpora, the time taken to summarize all 30 document sets was measured.

For statistical significance test, Wilcoxon's signed rank test [79] was used. Since the test was multiple comparison, Holm's method [33] was used to adjust the significance level. The whole significance level was set as 0.05.

6.6.4 Parameter Setting

This section describes the settings of pamameter. Among the following parameters, the concept importance was used by all methods, the concept redundancy was used by **RCKM** and **RCKM-LH**, and the step size and the number of iterations was used by only **RCKM-LH**.

Concept Importance

According to the corpus, concept j and its importance w_j was set as follows:

[8] Noun, verb, adjective and unknown word.

1. **TSC-3** Concept j was set as a content word , its importance w_j was set as $w_j = tf_j log(\frac{N}{df_j})$ based on tf-idf [23, 16], where tf_j is the number of occurrence of content word j in the input document set, df_j is the number of documents containing word j in a newswire article corpus, N is the number of documents in the newswire article corpus. As the newswire article corpus, the 2003 and 2004 Mainichi shinbun corpus [9] was used. To segment the sentences and identify their pos tags, a morphological analyzer proposed by Fuchi and Takagi [24] was used.

2. **Review** Since the target of summarization is a set of reviews, concept j was set as opinion information. The definition of opinion information and the way to extract the information are the same as explained in Section 4.4.1. Its importance w_j was set as the frequency of concept j in the input document set.

Concept Redundancy

Following four redundancy parameter, r_j , were tested.

1. **ON** This setting sets the concept redundancy as 1. It forbids a concept from occurring more than once in the same summary, i.e. $r_j = 1$. In the case where this constraint is used, the same concept can occur only once in the summary, and hence it is expected that the redundancy of the summary is reduced as well as the maximum coverage model. The difference between the maximum coverage model and the redundancy-constrained knapsack model with this constraint is that the former redunce the redundancy with the objective function, the latter do so with this constraint.

2. **KL** This setting sets the concept redundancy as the product of the frequency of occurrence of concept j in the input document set and the ratio of maximum summary length K and sum of the length of the input documents $L = \sum_{i=1}^{n} l_i$. If the value is decimal, the value is rouded out, i.e. $r_j = \lceil tf_j \frac{K}{L} \rceil$. By this constraint, it is expected that the distribution of the frequency of concepts in the input documents is reproduced in the summary.

3. **SR** This setting sets the concept redundancy as the square root of the frequency of the words in the input documents. If the value is decimal, the value is rounded

[9] http://www.nichigai.co.jp/sales/corpus.html

off, i.e. $r_j = \lfloor \sqrt{tf_j} \rfloor$. Unlike **KL**, this constraint is unaffected by the maximum summary length and more tolerant of redundancy.

4. **RF** This setting set the concept redundancy as the number of occurrence of concepts in the reference. This setting shows the perfomance where the ideal concept redundancy is set. Since there are multiple references for each input document set in the review corpus, the average number of occurrence was used as the constraint. If the value is decimal, the value is rounded out.

These parameteres were common in the both corpora.

Step Size

As an initial value, α starts with 1, then the value is inverse of the times of update of the Lagrange multipliers, i.e. for the first update α is 1, then for the second update α is $\frac{1}{2}$, for the third update α is $\frac{1}{3}$.

Iteration

The number of iterations T must be set when the decoding method is the Lagrange heuristic. In this experiments, the case of $T = 10$ and $T = 100$ were tested. Their results are indicated as **RCKM-LH(10)** and **RCKM-LH(100)**, respectively.

6.6.5 Results and Discussion

The results of the quality is shown in Table 6.3 and the results of the speed is shown in Table 6.4.

First, we describe the results of evaluation in the TSC-3 corpus. Among the variants of **RCKM**, **RF** achieved the best ROUGE score, then **SR** followed. **RF** and **SR** surpassed **MCM** significantly at their optimal solution. When the decoding with the Lagrange heuristic **RCKM-LH** was used, **RF** surpassed **MCM** in the both case of **RCKM-LH(100)** and **RCKM-LH(10)**. However, **SR** has no significant difference against **MCM**. As shown in Table 6.4, the proposed decoding method **RCKM-LH** successfully generated summaries far faster than **MCM** did; it could generate the summaries quickly, which were either equal to or better than the summaries generated by **MCM**. Given n sentences and maximum summary length K the runnning time of

Table 6.3: Results of the evaluation of the summary quality.

		TSC-3		**Review**	
		ROUGE-1	ROUGE-2	ROUGE-1	ROUGE-2
ON	RCKM	0.319	0.168	0.401	0.244
	RCKM-LH(100)	0.249	0.111	0.400	0.229
	RCKM-LH(10)	0.229	0.106	0.400	0.229
KL	RCKM	0.388	0.189	0.410	0.257
	RCKM-LH(100)	0.298	0.155	0.407	0.240
	RCKM-LH(10)	0.296	0.152	0.406	0.240
SR	RCKM	0.493	0.238	0.404	0.245
	RCKM-LH(100)	0.466	0.223	0.403	0.232
	RCKM-LH(10)	0.454	0.217	0.403	0.232
RF	RCKM	0.501	0.244	0.410	0.242
	RCKM-LH(100)	0.470	0.229	0.406	0.240
	RCKM-LH(10)	0.451	0.213	0.403	0.240
	MCM	0.454	0.218	0.403	0.242
	MCM-GR	0.392	0.143	0.396	0.226
	KM	0.443	0.204	0.351	0.203
	HUMAN	-	-	0.510	0.333

RCKM-LH is linearithmic in n, because the running time of the dynamic programming knapsack algorithm and the greedy algorithm, which are invoked in the algorithm of **RCKM-LH**, are $O(nK)$ [41] and $O(n\log n)$ [39], respectively. The running time of **RCKM-LH** also depends on the number of iteration, T. As shown in Table 6.4, the time **MCM** took to decode the all 30 instances was almost 11 days. There was no instance solved within an hour by **MCM**. In contrast, **RCKM-LH(100)** found out good approximate solutions equal to **MCM** within an hour. **MCM-GR** could generate summaries far faster than **MCM** did, but the ROUGE score of **MCM-GR** was siginicantly lower than that of **MCM**. It presented that search error occurred in the process of search. There is no significant difference between **KM** and **MCM**, **MCM** showed a higher ROUGE score as a whole.

The proposed summarization model, **RCKM**, surpassed **MCM** when its concept redundancy is set as either **RF** or **SR**. The reason is that the words in the references have a certain degree of redundancy. Figure 6.1 is a reference in the TSC-3 corpus. The

Table 6.4: The results of evaluation of the decoding speed.

		TSC-3 (sec.)	**Review** (sec.)
ON	RCKM	2,271.5	115.4
	RCKM-LH(100)	304.9	13.9
	RCKM-LH(10)	38.8	2.2
KL	RCKM	2,398.1	136.6
	RCKM-LH(100)	372.4	14.2
	RCKM-LH(10)	48.4	2.3
SR	RCKM	2,642.4	148.2
	RCKM-LH(100)	649.8	14.7
	RCKM-LH(10)	72.4	2.3
RF	RCKM	2,466.3	142.4
	RCKM-LH(100)	427.3	14.6
	RCKM-LH(10)	51.9	2.7
	MCM	924,349.3	2,458.3
	MCM-GR	25.6	0.3
	KM	8.1	0.5

same content words are indicated as block letter. As shown in Figure 6.1, reference summaries have redundancy.

Figure 6.2 is a distribution of the redundancy in summaries from the TSC-3 corpus. The horizontal axis indicates the frequency of content word occurrence in one reference; the vertical axis indicates the number of words. Dots in the figure indicate a distribution of redundancy of methods and parameters explained in Section 6.6.4. For example, there are 2093 words that occur once in one reference; there are 10 words that occur more than 9 times in one reference. This graph shows that some words occur more than once in one reference. As shown in Figure 6.1 and 6.2, it is not unusual that the same words occur multiple times in one reference.

The redundancy-constrained knapsack model proposed in this chapter can allow summaries to have some redundancy through the parameter of the concept redundancy. In contrast that model, **MCM** avoids the redundancy. As shown in Figure 6.2, the summaries generated by **MCM** have lower redundancy than the references have. Hence, it is unable to fully capture the phenomenon that the same words occur multiple times in the summary shown in Figure 6.1. In costrast, the knapsack model, **KM**, generated

エチオピア 北部の約２５０万年前の地層で米国・ **エチオピア** ・日本の研究チームが発見した人骨 **化石** が、 **猿人** から **原人** へ **進化** する過程にある **新種** の **猿人** であることが分かった。現代人の直接の祖先とされる約２００万年前の **原人** 、 **ホモ** ・ハビリスとそれ以前の **猿人** の間をつなぐ **化石** はなく、 **人類** の **進化** のミッシングリンクとされてきた。この **化石** は **新種** の **猿人 化石** として、現地語で「驚き」を意味する言葉から「ガルヒ **猿人** 」と名づけられた。今回発掘された手足の **骨** は、長さのバランスが約３２０万年前のアファール **猿人** 、約１７０万年前の **原人** 、 **ホモ** ・エレクトスの中間にあたる。二足歩行に適するよう脚が長くなっていたとみられる。また、 **石器** で切り取られた跡のある大型動物レイヨウの **骨** なども見つかった。 **人類** が肉食をしたことを示す **最古** の証拠で、 **石器** を実際に使用した跡としても **最古** となる。

Figure 6.1: A reference in the TSC-3 corpus (in Japanese)

summaries that have high redundancy. As Figure 6.2 shows, in summaries generated with **KM**, there are more words that occur more than three times in the same summary; there are 42 words that occur more than nine times in the same summary. This property that tends to generate redundant summaries should be avoided in the context of multi-document summarization.

Figure 6.2 shows that **SR** generates the summary whose redundancy is close to that of references. In general, in the knapsack model, the types of words that occur certain times in the summary is decreasing slowly from the left to the right. This shows that in the knapsack model there are little difference between the number of concepts that occur only once in the summary and the number of concepts that occur multiple times in the summary, i.e. this shows that the amount of the latter is relatively large. In contrast, in the maximum coverage model, the types of words are decreasing rapidly from the left to the right. This shows that the type of words that occur only once in the summary are relatively large. The redundancy of references shifts as get between the knapsack model and maximum coverage model and the redundancy of summaries by **SR** shifts close to that of references. Since **SR** reproduced the redundancy close to that of references, it could show a good performance. Excess redundancy like the summary by the knapsack model is problematic in multi-document summarization. In contrast, the property that the same words or related words occur in adjacent sentences is called as Lexical chain, and it is used as a cue to identify important sentences [5, 16]. **RCKM**

could capture this property.

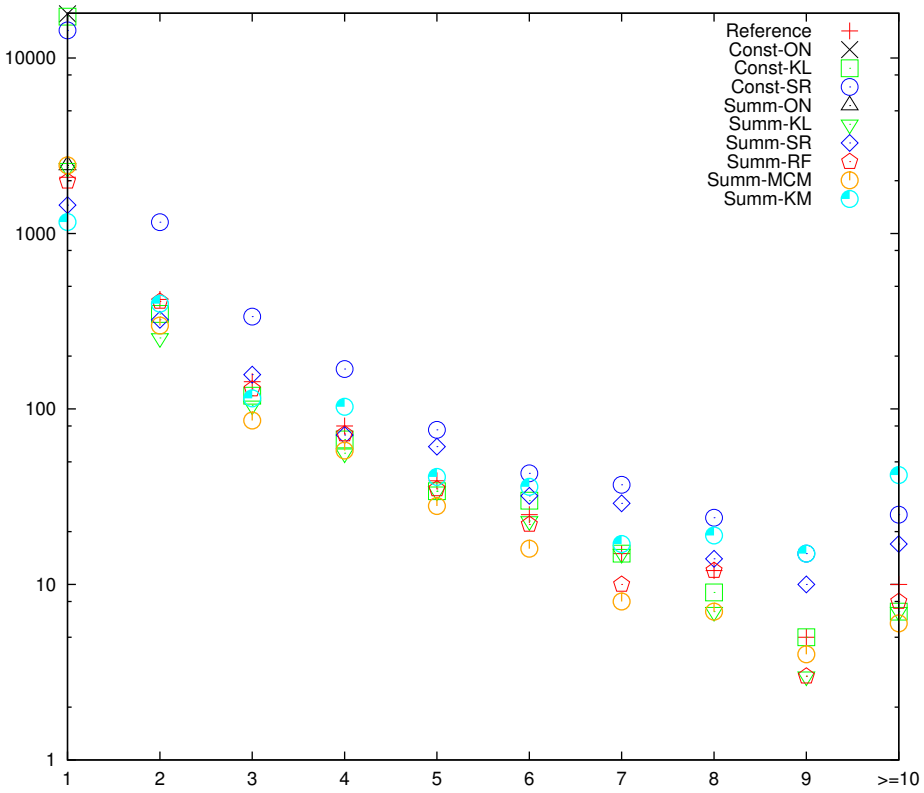

Figure 6.2: Distributions of redundancy in results of the TSC-3 corpus

The study of text coherence evaluation leverages this repetition to capture the coherence. A method to evalute the text coherence, Entity grid [7, 9, 90], leverages the transition of syntax role of nouns as features, assumed that the same words occur multiple times in the same document. References were written manually, and hence they have this property from the aspect of text coherene. In view of this, redundancy parameter **r** can be estimated from the aspect of text coherence.

Among the constraints of concept redundancy in **RCKM**, **RF** showed the best performance, then **SR** and **KL**, **ON** followed. Since the setting that mimics the redundancy of references achieved the highst pefromance, the setting of concept redundancy is important to **RCKM**. To set redundancy parameter **r** appropriately, the regression model predicting the appropriate concept redundancy from the input documents and corresponding references can be used. The quality of summaries generated by **ON** was notably worse than **SR** and **KL**. This can be explained from the aspect of the text

coherence. The constraint of **ON** forcing the same words to occur only once in the summary disallows summaries to have the property explained above, and hence the summries generated by **ON** lacks the text coherence. This is problematic from the aspect of mimicking the human references.

In comparison to **KL**, **SR** achieved a significantly higher ROUGE score. As Const-KL in Figure 6.2 shows, the redundancy of **KL** mimics that of references well, but the types of words occuring more than once are lower than **SR**. Since the summarization rate of document sets in the TSC-3 corpus is almost 6%, **KL** imposes the strict constraint to summaries. As Summ-KL in Figure 6.2 shows, this disallow the summaries to have enough redundancy, **KL** is inferior as compared with **SR** in terms of ROUGE.

In cotrast, Const-SR allows summaries to have high redundancy, the redundancy of summaries generated, Summ-SR, is close to that of references.

Next, the results of evaluation in the review domain are described. Among the methods, **RCKM** and **MCM**, **KM**, the ROUGE score of **KM** is significantly lower than that of **MCM**. Unlike the results of the TSC-3, there is no significant difference between **MCM** and **MCM-GR**. There is also no significant difference between **MCM** and **RCKM** and among the constraints of concept redundancy in **RCKM**. In contrast, as with the evaluation in the TSC-3 corpus, Table 6.4 shows that the proposed decoding method generates summaries faster than **MCM**.

All methods other than **HUMAN** fell short of **HUMAN**. There are two possible reasons. One reason is the setting of the concept importance. In this experiment, the frequency of opinion information in the input documents was used as the concept importance, but by learning the importance with the references a better ROUGE socre can be obtained. Another reason lies in the way to produce a summary: sentence extraction. As previously mentioned, the references of the review corpus were written by humans without any constraints, it might be difficult to reach the quality of them by mere extraction. To address this problem, the operation of rewriting sentences, such as sentence compression [95], should be considered.

As the results of the TSC-3 and review corpus shows, in the former **RCKM** surpasses **MCM**, but in the latter they have no difference. It clearly shows the difference of summarization of newswire articles and reviews. The former considers content words as a target to be covered, and hence it can achieve a higher ROUGE score by allowing a summary to contain the same words multiple times as shown above. In contrast, the latter considers opinion information as a target to be covered, and hence it needs the same concepts only once in a summary, as **ON** showing an inferior perfor-

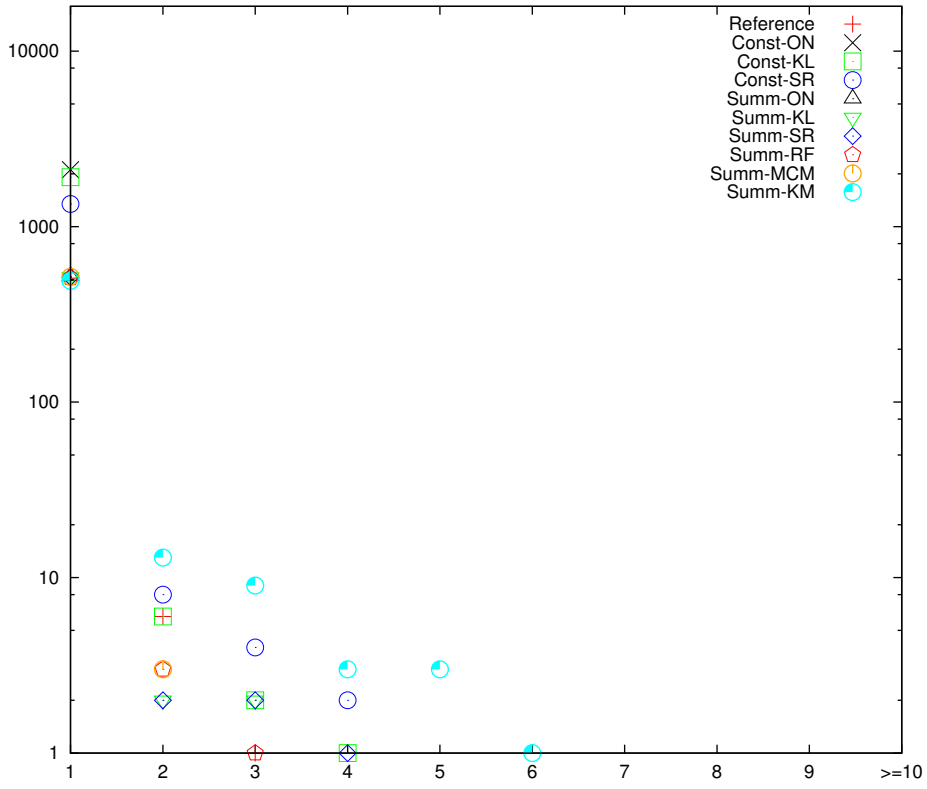

Figure 6.3: Distributions of redundancy in results of the review corpus

mance in the TSC-3 corpus achieved the performance as well as the other methods in the review corpus.

Figure 6.3 shows a distribution of the redundancy in summaries from the review corpus. The horizontal axis indicates the frequency of opinion information occurrence in one reference; the vertical axis indicates the number of opinion information. As compared to Figure 6.2, the difference is obvious; in the review domain, one opinion information occurs only once in the reference.

Finally, we show the approximation accuracy of the Lagrange heuristic proposed in this chapter. The accuracy is shown in Table 6.5. The value indicates the ratio of the value of the objective function of summary generated by the above approximation method to that of the optimal solution in percentage as the value of the optimal solution is 100. The value was the average of each accuracy of 30 document sets. The concept redundancy was set as **SR**. As Table 6.5 shows, the proposed decoding methods have good approximation accuracy.

Table 6.5: Approximation accuracy

	TSC-3	**Review**
RCKM	100.0	100.0
RCKM-LH(100)	97.0	97.7
RCKM-LH(10)	95.4	97.7

6.7 Conclusion

This chapter proposed a novel multi-document summarization model, the redundancy-constrained knapsack model, aimed to speed up the decoding of the maximum coverage model, which plays an important role in the context of multi-document summarization. The summary of this chapter is shown below:

- A performance of the summarization model based on the redundancy-constrained knapsack problem is equal to or higher than that based on the maximum coverage problem in terms of ROUGE [47].
- Approximate solutions acquired by a proposed decoding method based on the Lagrange heuristic stack up with the optimal solutions acquired by the maximum coverage model in terms of ROUGE [47].
- The decoding speed of the proposed method for the redundancy-constrained knapsack model is far faster than that of an ILP solver for the maximum coverage problem.

As future work, the parameter of concept redundancy should be set from the aspect of the text coherence.

Chapter 7

Conclusion

7.1 Summary

In this dissertation, we addressed three problems: readability, diversity of content, and speed.

For the first problem, readability, we proposed a joint model for sentence extraction and ordering in Chapter 4. The proposed method performs sentence extraction and ordering simultaneously with a novel formulation of multi-document summarization and its integer linear programming formulation. The proposed formulation is a mixture of the maximum coverage model and traveling salesperson problem and can be decoded by an integer linear programming solver. By the proposal, a resulted summary is more readable than a summary produced by a method which extracts and orders sentences separately.

For the second problem, we proposed a method to leverage transfer learning for addressing the diversity of topics in the input documents in Chapter 5. The proposed method leverages feature augmentation [19] to transfer the knowledge from other domains to the target domain for summarizing the documents. The experiments showed that our proposed method improved the quality of a summary in terms of ROUGE evaluation.

For the third problem, we proposed a novel model for multi-document summarization, the redundancy-constrained knapsack model, and a fast decoding method for the model in Chapter 6. The proposed model is based on the knapsack model, which suffers from the redundancy in a summary, but the redundancy constraint curbs the redundancy effectively. Since the model is a variant of the knapsack model, we can draw on the effective dynamic programming algorithm to decode the model. Furthermore, the

model improves coherence of a resulted summary, it produces a better summary than a summary of the maximum coverage model, which lacks the coherence of a summary due to its avoidance of redundancy.

As shown above, we provided the methods to address the problems resulted from the recent growth of the use of the Internet.

These three methods interrelate closely with each other. The method introduced in Chapter 5 can be used to estimate the weights for content selection in the method introduced in other chapters. In Chapter 4, the content score of an opinion is calculated based on its frequency of occurrence in the input document set. However, it can be calculated by machine learning and the method introduced in Chapter 5. In particular, since the proposed opinion summarization method was tested by two domains, reviews of restaurants and commodities, to bridge the gap between these two domains by transfer learning can be promising in future research.

The decoding method introduced in Chapter 6 can be used to select the utterances in Chapter 5. The model in Chapter 5 is based on the maximum coverage problem; therefore the model can naturally be replaced with the redundancy-constrained knapsack problem which can be decoded by the proposed method based on the Lagrange heuristic. In contrast, it cannot directly be used to the model introduced in Chapter 4 because it selects and orders the sentences jointly. To develop a method to decode quickly the model that selects and orders sentences jointly is also important in future research.

7.2 Future Directions

There are two paths for future work: sentence rewriting and readability prediction.

Sentence Rewriting In this dissertation, we did not touch a rewriting operation of sentences, like sentence compression [40, 16, 52, 97, 10], sentence simplification [87, 81], and paraphrasing [37, 25]. Sentence rewriting can be incorporated in the proposed methods and it is expected to improve the content and readability of a resulted summary.

Readability Prediction To make an output summary more readable, readability prediction is a promising technique. Readability prediction is a task to predict the readability of a document [65, 64, 57]. A produced summary must be readable, because it

is generated for helping users understand the important information in the documents, and must avoid making users misinformed. In this dissertation, we incorporated a function evaluating the coherence of sentences in the objective function. However, the proposed function can be improved by incorporating other features.

Bibliography

[1] Ernst Althaus, Nikiforos Karamanis, and Alexander Koller. Computing locally coherent discourses. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pp. 399–406, 2004.

[2] Chinatsu Aone, Mary E. Okurowski, James Gorlinsky, and Bjornar Larsen. A trainable summarizer with knowledge acquired from robust nlp techniques. In Inderjeet Mani and Mark T. Maybury, editors, *Advances in Automatic Text Summarization*, pp. 71–80. MIT Press, 1999.

[3] Egon Balas. The prize collecting traveling salesman problem. *Networks*, Vol. 19, No. 6, pp. 621–636, 1989.

[4] Michele Banko, Vibhu O. Mittal, and Michael J. Witbrock. Headline generation based on statistical translation. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pp. 318–325, 2000.

[5] Regina Barzilay and Michael Elhadad. Using lexical chains for text summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS)*, pp. 10–17, 1997.

[6] Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, Vol. 17, pp. 35–55, 2002.

[7] Regina Barzilay and Mirella Lapata. Modeling local coherence: an entity-based approach. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL)*, pp. 141–148, 2005.

[8] Regina Barzilay and Mirella Lapata. Aggregation via set partitioning for natural language generation. In *Proceedings of the Human Language Technology Conference of the NAACL*, pp. 359–366, June 2006.

[9] Regina Barzilay and Mirella Lapata. Modeling local coherence: An entity-based approach. *Computational Linguistics*, Vol. 34, No. 1, pp. 1–34, 2008.

[10] Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pp. 481–490, 2011.

[11] S. R. K. Branavan, Pawan Deshpande, and Regina Barzilay. Generating a table-of-contents. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)*, pp. 544–551, 2007.

[12] Peter E. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, Vol. 19, No. 2, pp. 263–312, 1993.

[13] Roy J. Byrd, Mary S. Neff, Wilfried Teiken, Youngja Park, Keh-Shin F. Cheng, Stephen C. Gates, and Karthik Visweswariah. Semi-automated logging of contact center telephone calls. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM)*, 2008.

[14] Giuseppe Carenini, Raymond Ng, and Adam Pauls. Multidocument summarization of evaluative text. In *Proceedings of the 11th European Chapter of the Association for Computational Linguistics (EACL)*, pp. 305–312, 2006.

[15] Harr Chen, S.R.K. Branavan, Regina Barzilay, and David R. Karger. Global models of document structure using latent permutations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 371–379, 2009.

[16] James Clarke and Mirella Lapata. Modelling compression with discourse constraints. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 1–11, 2007.

[17] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1–8, 2002.

[18] Koby Crammer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, Vol. 7, No. Mar, pp. 551–585, 2006.

[19] Hal Daume, III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2006.

[20] Pawan Deshpande, Regina Barzilay, and David Karger. Randomized decoding for selection-and-ordering problems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pp. 444–451, Rochester, New York, April 2007. Association for Computational Linguistics.

[21] Harold P. Edmundson. New methods in automatic extracting. *Journal of ACM*, Vol. 16, No. 2, pp. 264–285, 1969.

[22] Micha Elsner, Joseph Austerweil, and Eugene Charniak. A unified local and global model for discourse coherence. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pp. 436–443, Rochester, New York, April 2007. Association for Computational Linguistics.

[23] Elena Filatova and Vasileios Hatzivassiloglou. A formal model for information selection in multi-sentence text extraction. In *Proceedings of Coling 2004*, pp. 397–403, 2004.

[24] Takeshi Fuchi and Shinichiro Takagi. Japanese morphological analyzer using word co-occurrence: Jtag. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (ACL-COLING)*, pp. 409–413, 1998.

[25] Atsushi Fujita, Pierre Isabelle, and Roland Kuhn. Enlarging paraphrase collections through generalization and instantiation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 631–642, 2012.

[26] Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. Fast decoding and optimal decoding for machine translation. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 228–235, 2001.

[27] Dan Gillick and Benoit Favre. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pp. 10–18, 2009.

[28] Salim Haddadi. Simple lagrangian heuristic for the set covering problem. *European Journal of Operational Research*, Vol. 97, pp. 200–204, 1997.

[29] Ryuichiro Higashinaka, Yasuhiro Minami, Hitoshi Nishikawa, Kohji Dohsaka, Toyomi Meguro, Satoshi Kobashikawa, Hirokazu Masataki, Osamu Yoshioka, Satoshi Takahashi, and Genichiro Kikui. Improving hmm-based extractive summarization for multi-domain contact center dialogues. In *Proceedings of the IEEE Workshop on Spoken Language Technology (SLT)*, pp. 61–66, 2010.

[30] Ryuichiro Higashinaka, Yasuhiro Minami, Hitoshi Nishikawa, Kohji Dohsaka, Toyomi Meguro, Satoshi Takahashi, and Genichiro Kikui. Learning to model domain-specific utterance sequences for extractive summarization of contact center dialogues. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling)*, 2010.

[31] Tsutomu Hirao, Takahiro Fukushima, Manabu Okumura, Chikashi Nobata, and Hidetsugu Nanba. Corpus and evaluation measures for multiple document summarization with multiple sources. In *Proceedings of Coling 2004*, pp. 446–452, 2004.

[32] Tsutomu Hirao, Hideki Isozaki, Eisaku Maeda, and Yuji Matsumoto. Extracting important sentences with support vector machines. In *Proceedings of the 19th international conference on Computational linguistics (Coling)*, 2002.

[33] Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, Vol. 6, No. 2, pp. 65–70, 1979.

[34] Chiori Hori and Sadaoki Furui. Speech summarization: An approach through word extraction and a method for evaluation. *IEICE TRANSACTIONS on Information and Systems*, Vol. 87, No. 1, pp. 15–25, 2004.

[35] Eduard Hovy and Chin-Yew Lin. Automated text summarization in summarist. In Inderjeet Mani and Mark T. Maybury, editors, *Advances in Automatic Text Summarization*, pp. 81–94. MIT Press, 1999.

[36] Kenji Imamura, Genichiro Kikui, and Norihito Yasuda. Japanese dependency parsing using sequential labeling for semi-spoken language. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pp. 225–228, 2007.

[37] Tomoko Izumi, Kenji Imamura, Genichiro Kikui, Atsushi Fujita, and Satoshi Sato. Paraphrasing japanese light verb constructions: Towards the normalization of complex predicates. *International Journal of Computer Processing of Languages*, Vol. 23, No. 2, pp. 147–167, 2011.

[38] Frederick Jelinek. Fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development*, Vol. 13, No. 6, pp. 675–685, 1969.

[39] Samir Khuller, Anna Moss, and Joseph Naor. The budgeted maximum coverage problem. *Information Processing Letters*, Vol. 70, No. 1, pp. 39–45, 1999.

[40] Kevin Knight and Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, Vol. 1, No. 139, pp. 91–107, 2002.

[41] Bernhard Korte and Jens Vygen. *Combinatorial Optimization*. Springer-Verlag, third edition, 2008.

[42] Julian Kupiec, Jan Pedersen, and Francine Chen. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 68–73, 1995.

[43] Mirella Lapata. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pp. 545–552, Sapporo, Japan, July 2003. Association for Computational Linguistics.

[44] Mirella Lapata. Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, Vol. 32, No. 4, pp. 471–484, 2006.

[45] Kevin Lerman, Sasha Blair-Goldensohn, and Ryan McDonald. Sentiment summarization: Evaluating and learning user preferences. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL)*, pp. 514–522, 2009.

[46] Kevin Lerman and Ryan McDonald. Contrastive summarization: An experiment with consumer reviews. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pp. 113–116, 2009.

[47] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of ACL Workshop Text Summarization Branches Out*, pp. 74–81, 2004.

[48] Hui Lin, Jeff Bilmes, and Shasha Xie. Graph-based submodular selection for extractive summarization. In *Proceedings of the 11th Biannual IEEE Workshop on Automatic Speech Recognition and Understanding (ARSU)*, 2009.

[49] Hans P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, Vol. 22, No. 2, pp. 159–165, 1958.

[50] Inderjeet Mani. *Automatic Summarization*. John Benjamins Pub Co, 2001.

[51] Inderjeet Mani and Mark T. Maybury. *Advances in Automatic Text Summarization*. MIT Press, 1999.

[52] Andre Martins and Noah A. Smith. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pp. 1–9, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[53] Ryan McDonald. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European Conference on Information Retrieval (ECIR)*, pp. 557–564, 2007.

[54] Smaranda Muresan, Evelyne Tzoukermann, and Judith Klavans. Combining linguistic and machine learning techniques for email summarization. In *Proceedings of CoNLL 2001 Workshop at ACL/EACL 2001 Conference*, 2001.

[55] Sung Hyon Myaeng and Dong-Hyun Jang. Development and evaluation of a statistically-based document summarization system. In Inderjeet Mani and Mark T. Maybury, editors, *Advances in Automatic Text Summarization*, pp. 61–70. MIT Press, 1999.

[56] National Institute of Standards and Technology. The linguistic quality questions, 2007. http://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt.

[57] Ani Nenkova, Jieun Chae, Annie Louis, and Emily Pitler. Structural features for predicting the linguistic quality of text: Applications to machine translation, automatic summarization and human-authored text. In Emiel Krahmer and Theunem Mariet, editors, *Empirical Methods in Natural Language Generation: Data-oriented Methods and Empirical Evaluation*, pp. 222–241. Springer, 2010.

[58] Ani Nenkova and Kathleen McKeown. *Automatic Summarization*. Now Publishers, 2011.

[59] Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Genichiro Kikui. Opinion summarization with integer linear programming formulation for sentence extraction and ordering. In *Coling 2010: Posters*, pp. 910–918, 2010.

[60] Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Genichiro Kikui. Optimizing informativeness and readability for sentiment summarization. In *Proceedings of the ACL 2010 Conference Short Papers*, pp. 325–330, 2010.

[61] Hitoshi Nishikawa, Tsutomu Hirao, Toshiro Makino, and Yoshihiro Matsuo. Text summarization model based on redundancy-constrained knapsack problem. In *Proceedings of COLING 2012: Posters*, pp. 893–902, 2012.

[62] Hitoshi Nishikawa, Toshiro Makino, and Yoshihiro Matsuo. Domain adaptation with augmented space method for multi-domain contact center dialogue summarization. In *Proceedings of the Symposium on Machine Learning in Speech and Language Processing (MLSLP)*, 2012.

[63] Naoaki Okazaki, Yutaka Matsuo, and Mitsuru Ishizuka. Improving chronological sentence ordering by precedence relation. In *Proceedings of Coling 2004*, pp. 750–756, Geneva, Switzerland, Aug 23–Aug 27 2004. COLING.

[64] Emily Pitler, Annie Louis, and Ani Nenkova. Automatic evaluation of linguistic quality in multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 544–554, 2010.

[65] Emily Pitler and Ani Nenkova. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 186–195, 2008.

[66] Joseph J. Pollock and Antonio zamora. Automatic abstracting research at chemical abstracts service. *Journal of Chemical Information and Computer Sciences*, Vol. 15, No. 4, pp. 226–232, 1975.

[67] Vahed Qazvinian, Dragomir R. Radev, and Arzucan Ozgur. Citation summarization through keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pp. 895–903, 2010.

[68] Dragomir R. Radev, Hongyan Jing, Malgorzata Stys, and Daniel Tam. Centroid-based summarization of multiple documents. *Information Processing and Management*, Vol. 40, No. 6, pp. 919–938, 2004.

[69] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, Vol. 65, No. 6, pp. 386–408, 1958.

[70] Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, pp. 1–8, 2004.

[71] Oana Sandu, Giuseppe Carenini, Gabriel Murray, and Raymond Ng. Domain adaptation to summarize human conversations. In *Proceedings of ACL Workshop on Domain Adaptation in NLP*, 2010.

[72] Chao Shen and Tao Li. Multi-document summarization via the minimum dominating set. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling)*, pp. 984–992, 2010.

[73] Radu Soricut and Daniel Marcu. Stochastic language generation using widl-expressions and its application in machine translation and summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 1105–1112, Sydney, Australia, July 2006. Association for Computational Linguistics.

[74] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, Vol. 28, pp. 11–21, 1972.

[75] Hiroya Takamura and Manabu Okumura. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL)*, pp. 781–789, 2009.

[76] Hiroya Takamura and Manabu Okumura. Learning to generate summary as structured output. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM)*, 2010.

[77] Hiroya Takamura, Hikaru Yokono, and Manabu Okumura. Summarizing a document stream. In *Proceedings of the 33rd European Conference on Information Retrieval (ECIR)*, 2011.

[78] Shunji Umetani and Mutsunori Yagiura. Relaxation heuristics for the set covering problem. *Journal of the Operations Research Society of Japan*, Vol. 50, pp. 350–375, 2007.

[79] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, Vol. 1, No. 6, pp. 80–83, 1945.

[80] Kristian Woodsend and Mirella Lapata. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 565–574, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[81] Kristian Woodsend and Mirella Lapata. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 409–420, 2011.

[82] Shasha Xie, Benoit Favre, Dilek Hakkani-Tur, and Yang Liu. Leveraging sentence weights in a concept-based optimization framework for extractive meeting summarization. In *Proceedings of Interspeech*, 2009.

[83] Shasha Xie, Hui Lin, and Yang Liu. Semi-supervised extractive speech summarization via co-training algorithm. In *Proceedings of Interspeech*, 2010.

[84] Wen-tau Yih, Joshua Goodman, Lucy Vanderwende, and Hisami Suzuki. Multi-document summarization by maximizing informative content-words. In *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pp. 1776–1782, 2007.

[85] Klaus Zechner. Fast generation of abstracts from general domain text corpora by extracting relevant sentences. In *Proceedings of the 16th International Conference on Computational Linguistics (Coling)*, pp. 986–989, 1996.

[86] Yue Zhang and Stephen Clark. Joint word segmentation and pos tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, pp. 888–896, 2008.

[87] Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pp. 1353–1361, 2010.

[88] 奥村学, 難波英嗣. テキスト自動要約. オーム社, 2005.

[89] 浅野久子, 平野徹, 小林のぞみ, 松尾義博. Web 上の口コミを分析する評判情報インデクシング技術. NTT 技術ジャーナル, Vol. 20, No. 6, pp. 12–15, 2008.

[90] 横野光, 奥村学. テキスト結束性を考慮した entity grid に基づく局所的一貫性モデル. 自然言語処理, Vol. 17, No. 1, pp. 161–182, 2010.

[91] 西川仁, 長谷川隆明, 松尾義博, 菊井玄一郎. 文の選択と順序付けを同時に行う評価文書要約モデル. 人工知能学会論文誌, Vol. 28, No. 1, pp. 88–99, 2013.

[92] 高村大也, 奥村学. 最大被覆問題とその変種による文書要約モデル. 人工知能学会論文誌, Vol. 23, No. 6, pp. 505–513, 2008.

[93] 高村大也, 奥村学. 施設配置問題による文書要約のモデル化. 人工知能学会論文誌, Vol. 25, No. 1, pp. 174–182, 2010.

[94] 平尾努, 奥村学, 磯崎秀樹. 拡張ストリングカーネルを用いた要約システム自動評価法. 情報処理学会論文誌, Vol. 47, No. 6, pp. 1753–1766, 2006.

[95] 平尾努, 鈴木潤, 磯崎秀樹. 構文情報に依存しない文短縮手法. 情報処理学会論文誌：データベース, Vol. 2, No. 1, pp. 1–9, 2009.

[96] 平尾努, 鈴木潤, 磯崎秀樹. 最適化問題としての文書要約. 人工知能学会論文誌, Vol. 24, No. 2, pp. 223–231, 2009.

[97] 長谷川隆明, 西川仁, 今村賢治, 菊井玄一郎, 奥村学. 携帯端末のための web ページからの概要文生成. 人工知能学会論文誌, Vol. 25, No. 1, pp. 133–143, 2010.

List of Publications

Journal Papers

- 西川仁, 平尾努, 牧野俊朗, 松尾義博, 松本裕治. 冗長性制約付きナップサック問題に基づく複数文書要約モデル. 自然言語処理. Vol. 20, No. 4, pp. xx–xx, 2013. (in press)
- 西川仁, 長谷川隆明, 松尾義博, 菊井玄一郎. 文の選択と順序付けを同時に行う評価文書要約モデル. 人工知能学会論文誌, Vol. 28, No. 1, pp. 88–99, 2013.

International Conference and Workshop Papers (refereed)

- Hitoshi Nishikawa, Tsutomu Hirao, Toshiro Makino and Yoshihiro Matsuo. Text Summarization Model based on Redundancy-Constrained Knapsack Problem. In Proc. of the 24th International Conference on Computational Linguistics (Posters), pp. 893–902, 2012.
- Hitoshi Nishikawa, Toshiro Makino and Yoshihiro Matsuo. Domain Adaptation with Augmented Space Method for Multi-Domain Contact Center Dialogue Summarization. In Proc. of the 2nd Symposium on Machine Learning in Speech and Language Processing, 2012.
- Hitoshi Nishikawa, Takaaki Hasegawa, Matsuo Yoshihiro and Genichiro Kikui. Opinion Summarization with Integer Linear Programming Formulation for Sentence Extraction and Ordering. In Proc. of the 23rd International Conference on Computational Linguistics (Posters), pp. 910–918, 2010.
- Hitoshi Nishikawa, Takaaki Hasegawa, Matsuo Yoshihiro and Genichiro Kikui. Optimizing Informativeness and Readability for Sentiment Summarization. In

Proc. of the 48th Annual Meeting of the Association for Computational Linguistics (Short Papers), pp. 325–330, 2010.

Patents (allowed)

- 西川仁, 長谷川隆明. テキスト要約装置、テキスト要約方法及びテキスト要約プログラム. 特開 2011-150515.
- 西川仁, 長谷川隆明. テキスト要約方法、その装置およびプログラム. 特開 2011-085986.

Other Publications

- Hitoshi Nishikawa, Toshiro Makino and Yoshihiro Matsuo. A Pilot Study of Readability Prediction with Reading Time. In Proc. of the 2nd Workshop on Predicting and Improving Text Readability for Target Reader Populations, 2013.
- 今村賢治, 齋藤邦子, 貞光九月, 西川仁. 小規模誤りデータからの日本語学習者作文の助詞誤り訂正. 自然言語処理, Vol. 19, No. 5, pp. 381–400, 2012.
- Kenji Imamura, Kuniko Saito, Kugatsu Sadamitsu and Hitoshi Nishikawa. Grammar Error Correction Using Pseudo-Error Sentences and Domain Adaptation. In Proc. of the 50th Annual Meeting of the Association for Computational Linguistics (Short Papers), pp. 388–392, 2012.
- 今村賢治, 齋藤邦子, 貞光九月, 西川仁. 識別的系列変換を用いた日本語助詞誤りの訂正. 言語処理学会第 18 回年次大会予稿集, pp. 18–21, 2012.
- 西川仁, 平尾努, 牧野俊朗, 松尾義博. ラグランジュ緩和による複数文書要約の高速求解. 言語処理学会第 18 回年次大会予稿集, pp. 1071–1074, 2012.
- 西川仁, 牧野俊朗, 松尾義博. 転移学習による抽出型要約の精度向上. 情報処理学会研究報告, Vol.2011-NL-204, No.8, pp. 1–7, 2011.
- 西川仁, 長谷川隆明, 松尾義博, 菊井玄一郎. 文外照応を含む文の検出による抽出型要約の品質向上. 言語処理学会第 17 回年次大会予稿集, pp. 216–219, 2011.

- 東中竜一郎, 南泰浩, 西川仁, 堂坂浩二, 目黒豊美, 小橋川哲, 政瀧浩和, 吉岡理, 高橋敏, 菊井玄一郎. 共通状態と連結学習を用いたHMMによるコールセンタ対話の要約. 言語処理学会第17回年次大会予稿集, pp. 920–923, 2011.

- Ryuichiro Higashinaka, Yasuhiro Minami, Hitoshi Nishikawa, Kohji Dohsaka, Toyomi Meguro, Satoshi Kobashikawa, Hirokazu Masataki, Osamu Yoshioka, Satoshi Takahashi and Genichiro Kikui. Improving HMM-Based Extractive Summarization for Multi-Domain Contact Center Dialogues. In Proc. of the IEEE Workshop on Spoken Language Technology, 2010.

- Ryuichiro Higashinaka, Yasuhiro Minami, Hitoshi Nishikawa, Kohji Dohsaka, Toyomi Meguro, Satoshi Takahashi and Genichiro Kikui. Learning to Model Domain-Specific Utterance Sequences for Extractive Summarization of Contact Center Dialogues. In Proc. of the 23rd International Conference on Computational Linguistics (Posters), pp. 400–408, 2010.

- 西川仁, 長谷川隆明, 松尾義博, 菊井玄一郎. 文の内容性と連接性を目的関数とする複数の評価文書の要約. 言語処理学会第16回年次大会予稿集, pp. 39–42, 2010.

- 長谷川隆明, 西川仁, 今村賢治, 菊井玄一郎, 奥村学. 携帯端末のためのWebページからの概要文生成. 人工知能学会論文誌, Vol. 25, No. 1, pp. 133–143, 2010.