A Study on Hierarchical Table of Indexes for Multi-documents

By LE Thi Ngoc Tho

A thesis submitted to
School of Information Science,
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Master of Information Science
Graduate Program in Information Science

Written under the direction of
Professor Akira Shimazu

September, 2012

A Study on Hierarchical Table of Indexes for Multi-documents

By LE Thi Ngoc Tho (1010226)

A thesis submitted to
School of Information Science,
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Master of Information Science
Graduate Program in Information Science

Written under the direction of
Professor Akira Shimazu

and approved by
Professor Akira Shimazu
Associate Professor Kiyoaki Shirai
Professor Satoshi Tojo

August, 2012 (Submitted)

Acknowledgements

I would like to express my gratitude to all those who gave me the possibility to complete this thesis.

First and foremost, I am greatly indebted to Assistant Professor Nguyen Le Minh and Professor Akira Shimazu for their valuable advices and consistent encouragement since the early stage of my study. My deep appreciation is sent to Associate Professor Kiyoaki Shirai, Professor Satoshi Tojo and Professor Hiroyuki Iida for reviewing and comments this thesis. This thesis is dedicated to my colleagues in Shimazu-Shirai laboratory and teachers at language center who has always helped me and believed that I could do it.

Second, it gives me great pleasure in acknowledging the support of the Vietnamese Ministry of Education and Training, VNU-HCM University of Science and Japan Advanced Institute of Science and Technology to be studied in Japan.

Last but not least, I would like to show my great gratitude to my beloved family and friends for giving me strength and motivation during the time at Master course.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Research Context

As we all known, to keep all activities in stable and smooth, our society is governed by several kinds of rules and laws, such as the laws of Administration, Construction, Environment, Finance, Tax, Education, Industrial, etc. With so many kinds of laws, there is a need to support the laws systems. *Legal Engineering* is a new research field studies the methodologies of how to apply information science to laws. Legal Engineering is not just seen as a way of capturing and distribute knowledge, but also an analytical approach that help to improve legal quality. There are many problems that Legal Engineering concerns to, among of them, two needs which information science can be applied to :

1. A developing methodology to aid the design of big information processing system; and

2. Supporting the law officers check related documents when look up information for editing laws.

In this thesis, we concern the second need, and we study to generate a navigation tool to help the law officers access to their interested law sections quickly as well as provide them the general ideas of the laws when navigating. This research is about apply Natural Language Processing (NLP) approaches to reply to the requirements. Specifically, this research is a sub-field of NLP, called *automatic summarization* for multi-documents.

A summary of a document or a collection of documents is a "condense" representation of main ideas of the content. It is obvious that the summary of documents will help the readers gain the general ideas of documents. However, in case of the output of the summary for multiple documents, even if the summary is much shorter and more concise than the original documents, it is still difficult for reader to understand all main ideas in structural organization. So, while trying to summarize documents, we take into account not only the main ideas for multiple documents but also the organization of main ideas as well. In order to do that, we generate a tree-based structure, called *hierarchical table of indexes*. A table of indexes in hierarchical structure helps the readers understanding the content and the structure in semantics aspects. This hierarchical structure

of representation also provides a navigation for the readers to quickly refer to interested information.

Hierarchical summarization has been noticed more than 10 years ago by work of Lawrie et al. [23] [24]. In [23], the topic hierarchies is constructed based on probabilistic model, but the output of this work are single words only, that may not express the ideas naturally. In [24], the author introduced a framework for automatically building hierarchies for small collections of texts, based on statistical models of language. However, this work does not consider the semantic aspect of words. To improve these drawbacks, we represent the output of hierarchical summary in form of a combination of words, called *keyphrases*, which load the important information of the document. We also consider the semantic aspect of the words when extracting them from multiple documents.

So far, Branavan et al.[4] and Nguyen et al.[45] have used supervised technique to generate a tree wherein a node represents a segment of text and a title that summarizes its content with assumption that the hierarchical tree of summary is available. Moreover, those work applied for single documents without awareness of *overlapping*, *supplementing* and *contrasting* in the content.

In this thesis, we also focus on the constructing of hierarchical tree while trying to generate the summary of multiple documents. In details, we create a structural summary in representation of a hierarchical table of indexes with an unsupervised approach. A hierarchical table of indexes is similar to a table of indexes appearing at the back of books. In addition the role as a representation the summary, it helps readers quickly refer to their interested sections containing their interested key words. *"Hierarchical"* in HTI means the set of indexes in lower tier contain more specific information than the higher one. To Legal Engineering, this work contributes a navigation which helps law officers refer to the interested sections quickly while review the main ideas of the law documents. To the sub-field automatic summarization of NLP, this works introduce a new representation, in which the output of summary is a collection of keyphrases in hierarchical structure.

1.2 Goal of Thesis

The specific problem statement of this thesis is: given a collection of related documents in a specific theme (such as law documents), our main goals are:

(i) Generating a hierarchical table of indexes for representing the main ideas of multi-documents; and

(ii) Providing a navigation to get details information quickly.

Generally, because of involving multiple sources of information, the challenges we have to face is not only the *overlapping* of the ideas between documents, but some parts of a document can *supplement* to some parts of another one, or the content between documents becomes *contradiction* to each other. To treat such problems may occur, the redundancy of information should be identified as well as recognized the novelty in the content; also,

the output summary should be ensured to be coherent and complete. So, in the scope of this research, we assume:

(i) One segment should be belong to (only) one topic;

(ii) Two segments are in the same topic if they are semantic related, or they are similar in semantic aspect.

Specifically, we take into account the organization of main ideas as well when trying to get the summary of multiple documents. In order to do that, we generate a tree-based structure, called *hierarchical table of indexes (HTI)*. A table of indexes in hierarchical structure will help the readers understanding the main ideas and their organizations in semantics aspects. It also provides a navigation for the readers to quickly refer to interested information. The process to generate HTI involves to three steps:

(i) Segment all documents into separate segments based on topics;

(ii) Construct hierarchical tree of segments (HTS);

(iii) Extract key-phrases from each segments, and generate HTI from HTS.

This thesis mainly focuses on making all the process automatic. The first step is assumed to be available by apply existing methods of segmentation such as TextSeg[44], MinCutSeg [30], BayesSeg [12] in which a document is separated into segments based on topic by finding the maximum-probability segmentation. We will mainly focus on the second and third step of the process.

In detail, this study concentrates on automatic process to construct a hierarchical tree of segments (HTS) from segmented texts, and then to generate a HTI for multi-documents based on the HTS by extracting keyphrases of each segment. To construct HTS, an unsupervised clustering algorithm is applied recursively to construct the hierarchical structure; after that HTI is drawn from HTS using an unsupervised keyphrase extraction algorithm. It results in the summary of multi-documents in hierarchical structure (in bag-of-words format) while generating the indexes. Experiments were applied for both English and Japanese in the law field, which will serve as a task in contribution to Legal Engineering.

The thesis is organized as follows. Chapter 2 presents some background knowledge about Text Segmentation, Clustering algorithms, Text Similarity, and Word Extraction. Chapter 3 describes our proposed approach to solve the problem. Chapter 4 demonstrates the proposed framework with some the experiments and preliminary results. Finally, Chapter 5 presents a conclusion and future works.

Chapter 2

Background

In this chapter, we introduce some background knowledge relevant to our framework to generate Hierarchical Table of Indexes. Corresponding to the first step, we provide an overview of Text Segmentation in Section 2.1. For constructing Hierarchical Tree of Segment in step 2, Clustering approaches and Text Similarity metrics will be presented in Section 2.2 and Section 2.3, respectively. Also Key Word Extraction in Section 2.4 for step 3 to extract keyphrases when building hierarchical table of indexes.

2.1 Text Segmentation

Text segmentation is one of the fundamental tasks in natural language processing problems such as information retrieval or automatic summarization. In real world, the natural language patterns usually appear without the explicit boundaries. The goal of this process is to divide the text document into smaller unit based on the topic boundaries; in another word, segmentation process is a task of finding the boundaries of segments in a stream of text (see Figure 2.1).

On view point of methodology, we can divide segmentation into two main approaches: supervised [34] [16] and unsupervised [18] [44] [30], where supervised approach requires annotated data for training but unsupervised does not. On the other hand, segmentation process can be divided into two types of representation output: linear [19] [44] [12] and hierarchy [11] [6]; in which, linear segmentation separate text document into continuous parts. In this research, we consider the unsupervised approach in text segmentation, and the view of representation output.

In unsupervised textual segmentation, previous work on discourse segmentation indicated that *lexical cohesion* [17] of text is a strong feature. Many algorithms such as TextTiling [19], C99 [7], TextSeg [44], LCSeg [16], MinCutSeg [30], BayesSeg [12] assume that variations in lexical distribution indicate topic changes, though each work explore lexical cohesion in different aspects, such as similarity between words [21], the cosine similarity between blocks of text [18], adaptive language model [1], word frequency model [38], inter-sentence similarity matrix [7], lexical distribution [44], variation of cosine similarity between sentence [30].

==========
Shares specified by Ordinance of the Ministry of Finance as being similar to shares or capital contributions listed on a financial instruments exchange prescribed in Article 291, paragraph (9), item (i) (Taxable Income of Nonresidents Who Have No Permanent Establishments) of the Order shall be as follows:Shares registered as issues registered for over-the-counter trading (meaning shares (including capital contributions and units of investment prescribed in Article 2, paragraph (14) (Definitions) of the Act on Investment Trusts and Investment Corporations; hereinafter the same shall apply in this Article) that an authorized financial instruments firms association prescribed in Article 2, paragraph (13) (Definitions) of the Financial Instruments and Exchange Act (referred to as an "authorized financial instruments firms association" in the following item) has registered, in accordance with the rules it has defined, as shares whose sales prices for over-the-counter trading shall be made public and for which materials concerning the issuing corporation thereof shall be open to the public)Shares of over-the-counter managed issues (meaning shares which have been delisted from a financial instruments exchange prescribed in Article 2, paragraph (16) of the Financial Instruments and Exchange Act or whose registration as issues registered for over-the-counter trading prescribed in the preceding item has been canceled, and which have been designated by an authorized financial instruments firms association in accordance with the rules it has defined)Shares traded on a foreign financial instruments market prescribed in Article 2, paragraph (8), item (iii), (b) of the Financial Instruments and Exchange Act
==========
With regard to matters concerning the application of the provisions of Part II, Chapter V (Filing of Returns, Payment, and Refunds for Residents) of the Act that are applied mutatis mutandis pursuant to Article 166 (Mutatis Mutandis Application to Nonresidents) of the Act, and the provisions of Part II, Chapter V (Filing of Returns, Payment, and Refunds for Residents) of the Order that are applied mutatis mutandis pursuant to Article 293 (Mutatis Mutandis Application to Nonresidents) of the Order, the provisions of Chapter III of the preceding Part (Filing of Returns, Payment, and Refunds for Residents) shall apply mutatis mutandis.
==========
....
....

Figure 2.1: An example on text segmentation

Unsupervised cohesion-based approaches can be distinguished based on the metric used in quantify cohesion and search technique. *Lexical chains* [16], which is defined as the repetition of a given lexical item over a fixed-length window of sentences, are characterized to be used in inference by selecting segmentation points at the local maxima of cohesion function. Lexical chain. MinCutSeg [30] optimized a normalized minimum-cut criteria based on a variation of the cosine similarity between sentences. TextSeg [44] search for

segmentations with compact language models using dynamic programming to search the space of segmentations. Bayesian approach to segmentation [36] assume a set of documents is characterized by some number of hidden topics that are shared across multiple documents, then linear segmentation is built by adding a switching variable to indicate whether to topic distribution for each sentence is identical to that of its predecessor.

Recently, *topic model* or topic-based representation of text documents is a research topic tendency. Topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. The first work on topic model is *latent semantic indexing (LSI)* [10], using singular value decomposition (SVD) to capture the most of variance in the collection and even some aspects of basic linguistic notions, though the resulting concepts may be difficult to interpret. Then *probabilistic LSI (pLSI)* [20] models each word in a document as a sample from a mixture model to capture the semantic relationship within a text, but pLSI is not a generative model of new documents. To solve the above drawbacks of previous work on topic model, Blei et al. presented a generative model *Latent Dirichlet Allocation (LDA)* [3] by incorporating the Dirichlet priors for topic mixtures shared by seen and unseen documents through the variational inference procedure.

Topic model can be applied to linear text segmentation [12], and on hierarchical text segmentation [11] [6] too. The first applying of Bayesian topic model are works of Blei and Moreno [2] and Purver et al. [36], using HMM-like graphical models for linear segmentation. Then, Eisenstein et al. in BayeseSeg [12] extent this model by marginalizing the language model using Dirichlet compound multinominal distribution. Later, in [11] multi-scale phenomenon of lexical cohesion is leveraged in Bayesian generative model for hierarchical topic segmentation.

2.2 Clustering

With recent information explosion, people may encounter a large amount of information which is stored as data for many purposes. One important task is that human have to separate a set of data objects to separate groups for more relationships between them and discover its structure to learn new knowledge in data mining process. To do that, machine learning approach can be applied and let computer learn some pre-specified structure of data, then distribute new data into appropriate structure then. However, annotating or assign label for data to specific structure is a costly task, especially with a large amount of data. Another difficulty is new given data might be outlier data samples that it is hard to decide which group it should be belong to. That problems leads to a need to distribute unlabelled data to groups, or to discover the structure of data.

2.2.1 Clustering Algorithms in General

Clustering is a automatic process of division unlabelled data into groups of similar objects. Each *cluster* (so-called *subset*, *group*, or *category*) consists of objects that are similar between themselves and dissimilar to objects of other groups. Specifically, clustering is

an unsupervised process of exploratory data analysis, the goal of clustering is deliver unlabelled data set into a finite and discrete set of hidden data structure. It means data is represented by a few clusters, though information may loss the simplification will be achieved instead.

Depending on the properties of cluster, the understanding of cluster may vary in different ways. The clusters can be *exclusive*, that means a data sample should belong to only one cluster; or, there may be *overlapping* between two or more clusters, that means a data sample may appear in more than one cluster. We will review some algorithms which separate data in exclusive clusters. Because there is no best criteria to divide a data set into clusters, it depends on the purposes or the problems that we design an appropriate clusters criteria. On aspect of clustering model, there are various models with different characteristics such as *connectivity* model, *centroid* model, *distribution* model, *density* model, *subspace* model, *graph-based* model, and so on.

The idea of *connectivity-based* clustering algorithm is objects being more related to nearby objects than the objects. A representative algorithm for this clustering model is *hierarchical clustering* which can be either top-down (divisive) or bottom-up (agglomerative). In detail, hierarchical agglomerative clustering (HAC), the bottom-up strategy, considers each object as a cluster at first, then combine two nearest clusters into one until all objects belongs to an exclusive cluster like demonstrated demonstrated in Figure 2.2. In contrast, the top-down strategy considers all objects belong to a cluster at first, and continuously divide the cluster until each object belongs to a separate cluster.

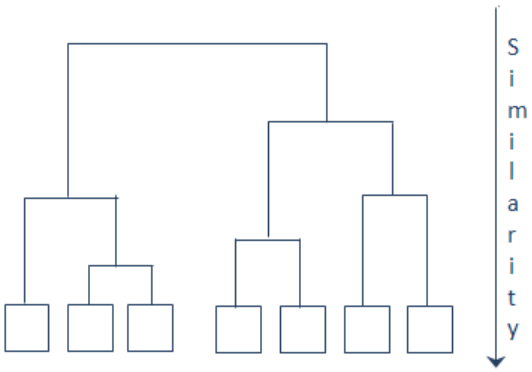


Figure 2.2: The Hierarchical Agglomerative Clustering (HAC) algorithm

In *centroid-based* clustering algorithms, clusters are represented as a single object though the popular object may not really a member of data set. The objective of centroid-based clustering algorithm is minimized the average distance between objects of the cluster. A representative algorithm of this clustering model is *K-Means*. When the number of cluster is fixed to K, from the data set K objects are selected randomly as initial centres (so-called *exemplars*), then objects are assigned to the nearest cluster based on distance. After that, new optimized centres are found by computing the average distance of clusters. This process is repeated until the centres of cluster is unchanged or slightly changed, and result the clusters with their members as in Figure 2.3. However, most

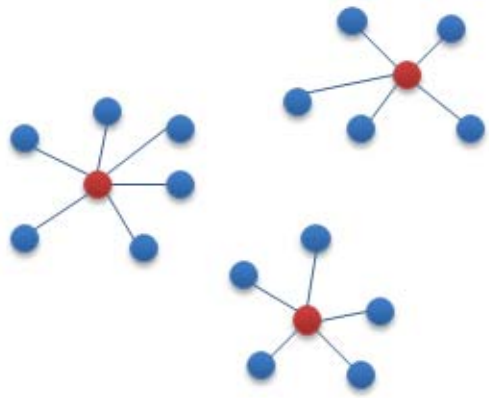


Figure 2.3: The K-Means algorithm

of centroid-based clustering algorithms is they require a pre-specified number of clusters that lead to another optimal problem that is how to find the number of clusters K. Another disadvantage is that wrong assigning objects to cluster may cause the incorrect of cut-borders between clusters.

The *distribution-based* clustering algorithms exploit distribution models to deliver objects to clusters. The key idea of this clustering model is objects belonging to the same cluster are likely to have the same distribution, and each object is assumed to be generated by a probability distribution. Then, clustering becomes the process of estimating the parameters of many underlying models. This method also provides the correlation and dependence of attributes between objects. A famous algorithm representing for distribute-

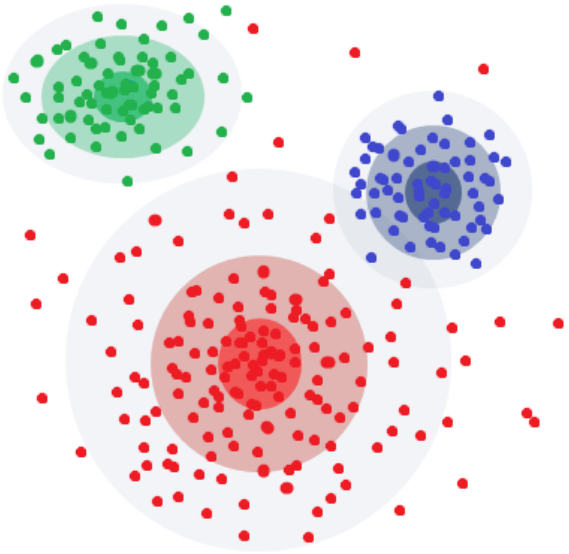


Figure 2.4: The Expectation Maximization clustering (EM-clustering) algorithm

based model is *Expectation Maximization clustering (EM-clustering)* illustrated in Figure 2.4. But, this algorithm is applied on the assumption that the data is followed by Gaussian

distribution which is a strong assumption on data.

In *density-based* clustering algorithms, a cluster is defined as an area which has higher density than other areas in the data set. This algorithm is able to capture the clusters in "natural" arbitrary shapes. Other objects in sparse areas are considered as noise objects. A famous algorithm based on density is *density-based spatial clustering of applications with noise (DBSCAN)* illustrated in Figure 2.5. This kind of algorithm bases on the number of objects in neighbourhood, each cluster continuously grows by recruiting neighbour objects given that the objects satisfy a specific objective function.

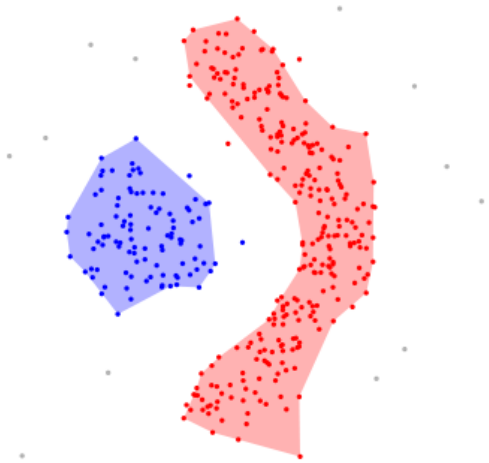


Figure 2.5: DBSCAN clustering algorithm

Another clustering model is *graph-based* clustering algorithms. This method models a complete graph from data set, where nodes in graph represent objects in data set, and the edges between nodes are relations between objects. These relations can be the distances or similarities between objects in data set. The key idea to find clusters of this clustering model is separating the graph into many sub-graphs whose nodes in the same cluster are closer than those in different clusters (Figure 2.6). This kind of clustering model will be discussed more details in Section 2.2.2.

2.2.2 Clustering in Computational Linguistics

In the research context of natural language processing (NLP), we are going to review the clustering algorithms applicable to computational linguistics. In the field of NLP, graph-based clustering has gained the intention in recent years [5] though some formal academic meeting such as annual workshop TextGraph [1]. The applications of graph-based clustering in NLP also varies in a wide range from document clustering, word clustering, co-reference resolution, word sense disambiguation.

To divide objects to cluster, the data set is model as a graph $G = (V, E, W)$, where $V = \{v_1, ..., v_N\}$ is the vertices of graph, with v_i represents for an object in data set

[1] `http://www.textgraphs.org/`

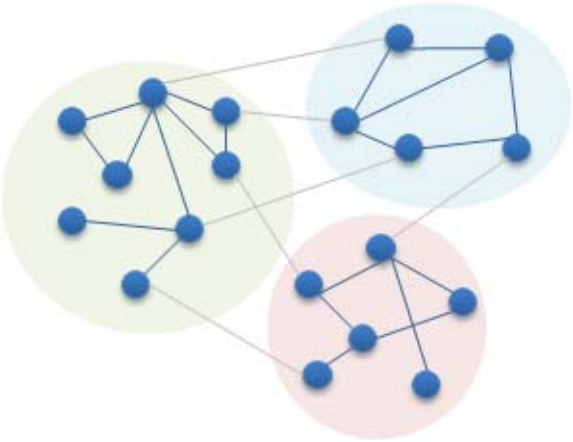


Figure 2.6: Graph-based clustering algorithm

(usually the object is word, phrase, sentence, segment, or document); the relationships between objects is expressed in the set of $E = \{v_{ij} \mid i, j \in [1, N]\}$ means the edges between nodes of graph; the important of relations between objects are reflected by the edge weights W which means is the distance between vertices in graph. In NLP the distance of text is usually the text similarity that will be introduced in detail in Section 2.3.

Clustering a data set which can be modelled as a graph is equivalent to find subgraphs of the complete graph. The key idea behind graph-based clustering algorithms is the graph consists of dense subgraphs provide that the distances between vertices in the same subgraph are closer than the external ones. There are two types of clustering algorithm using graph-based approach, which are divisive and agglomerative. Divisive clustering is top-down approach, previous work showed that divisive approaches are more efficient than the other one. In its turn, divisive approach can be divided to several subclasses, namely, *cut-based clustering, spectral clustering, multilevel clustering, random walks, shortest path.* All algorithms follow divisive approach are based on the hypothesis that split a graph to subgraphs recursively, and agglomerative clustering is bottom-up approach. Divisive approach also results the hierarchy in output because of multi-level process of clustering.

There is a variety of problems in NLP can be naturally represented as graphs, such as co-occurrence graphs, co-reference graphs, word/sentence/document graphs which are use in NLP problems such as *co-reference resolution*, *word clustering*, and *word sense disambiguation.*

Co-reference resolution is the problem of partitioning a set of *mentions* into *entities*, with entity is an object or a set of objects in real world (e.g people, organization), and mention is a textual reference to an entity. One of common approach for co-reference resolution in early stage based on clustering involving to two steps:

(i) Classification step to compute how likely one mention co-refers with the others;

(ii) Clustering step to group the mentions into clusters such that all mentions in a cluster refer to the same entity.

However, the early clustering approach to this problem suffers from a disadvantage that they do not search all the possible clustering to find the co-reference. On the other hand, this problem can be modelled as a graph such that the vertex represents a mention, and edge weight carries the co-reference likelihood between two mentions. So that the clusters can be identified globally, and graph-clustering approach applying to this problem yields competitive results.

The problem of *word clustering* is defined as clustering a set of words in to group such that similar words are in the same cluster. This technique takes benefit to many NLP problems, e.g., text classification or word sense disambiguation. Word clustering can be solved by follow these steps:

(i) Represent each word as a feature vector and computing the similarity of all pair of words;

(ii) Cluster the words, such the the similar words are grouped together.

In word clustering, the co-reference graph is also constructed as previous problem, but the relation between two vertices is usually computed by applying similarity measure (e.g., χ^2, cosine) on a co-occurence matrix.

There is no general graph-based clustering algorithm [5] effective for all NLP problems. Depend on the purposes or the problems, a clustering algorithm is proposed to optimize some quality measure, and there is no perfect measure that can capture all characteristics of cluster structure as well as there is no criteria to define the best characteristics of a cluster.

2.3 Text Similarity

Text similarity (or text relatedness) is a concept measuring the degree of overlapping in meaning between words, sentences, paragraphs, or documents in general. Measures of text semantic similarity have been used in many applications of NLP and related areas. One of earliest applications of text similarity is the vectorial model [39]. Text similarity has been used in many problems in NLP such as text classification, word sense disambiguation, extractive information, and text summarization. Specifically, given two texts (words, sentences, documents), the purpose of measuring text similarity is to figure out a score indicate their relations in meaning.

The simplest approach to find the similarity between two text segments is to use *lexical matching* method, and compute the similarity score based on the number of lexical units that occur in both input texts. To improve this simple method, weighting and factorizations [40] are considered, such as removing functional words (stop words), part of speech tagging, longest subsequence matching. However, the semantic of text is still hard to capture. For example, with two input *I have a dog* and *I own an animal*, lexical matching approach fails to discover the link between *dog* and *animal*, and unaware the identical meaning of *have* and *own* in this context. So, the purpose of finding text similarity score

is not only take into account the similarity on the word surface but also on the semantic meaning.

To overcome the limit of semantic in lexical approaches, corpus-based and knowledge-based approaches use a large corpus and thesaurus to capture the semantic aspect of word [42] [25] [47] based on the probability and statistics of words in input text. These semantic metrics have been successfully applied to NLP tasks such as word sense disambiguation [35], and synonym identication [42]. The vector-based approach is also a common choice to compare two strings of text in Information Retrieval systems [31]. This approach represent a document as a vector, then comparing a pair of documents is equivalent to compute the distance or similarity of a pair of vector (e.g Euclidean, cosine...).

Another well-known methods to compute similarity with corpus-based is the Latent Semantic Analysis (LSA) [22]. LSA is a high-dimensional linear association model, it analyses a large corpus of natural language text and generates a representation that get the similarity of words and text messages.

We distinguish text similarity to two main levels. The basic level is the similarity between words is mentions in Section 2.3.1. More advance in similarity is the semantic similarity between sentences or document is given in Section 2.3.2.

2.3.1 Similarity of Words

There is a large number of word-to-word similarity metrics that were proposed using distance-oriented measures computed from semantic networks, or using metrics based on models of distribute similarity learned from a large thesaurus.

The approach using distance-oriented measures computed the similarity of words from semantic networks such as WordNet [25] [2]. This kind of metric considers the words as concepts and calculates the similarity of concepts based on the distance of them on the semantic networks. We recall some common metrics proposed in previous work based on WordNet, such as:

- Leacock & Chodorow similarity [25], the length of the shortest path between two concepts in WordNet is exploited using node counting and the maximum depth of taxonomy D.
$$Sim = -\log \frac{length}{2D}$$
- Lesk similarity [26], the similarity of two concepts is defined as a function overlap between the corresponding definitions in dictionary.
- Wu & Palmer similarity [47], the similarity of two concept is measured by the depth of two concepts in the taxonomy and the depth of the least common subsumer (LCS).
$$Sim = \frac{2 \times depth(LCS)}{depth(concept_1) + depth(concept_2)}$$

[2] http://wordnet.princeton.edu/

- Resnik similarity [37] combines the probability of encountering an instance of LCS to the information content (IC)
- Lin similarity [27] add a normalization factor consisting of the information content of the input concepts.

$$Sim = \frac{2 \times IC(LCS)}{IC(concept_1) + IC(concept_2)}$$

Recently, when the free encyclopaedia Wikipedia [3] become popular dictionary, most concepts have been defined well by the the world community. Wikipedia become a promise thesaurus to look up the definitions of concepts. So some works are based on Wikipedia to find the similarity between concepts, such as uses snippets from Wikipedia to calculate the semantic similarity between words by using cosine similarity and TF-IDF [48]. Another use machine learning techniques to explicitly represent the meaning of any text as a weighted vector of Wikipedia-based concepts, and calculate the similarity between words as the cosine between the corresponding vectors [15].

Beside knowledge-based methods as introduced above, corpus-based methods are also explored for usage in measure the similarity. Such as PMI-IR uses Pointwise Mutual Information (PMI) and Information Retrieval (IR) to measure the similarity of pairs of words [42] using data collected by information retrieval. PMI-IR is an unsupervised measure for the evaluation of the semantic similarity of words. It is based on word co-occurrence using counts collected over very large corpora. With LSA, term cooccurrences in a corpus are captured by means of a dimensionality reduction operated by a singular value decomposition (SVD) on the term-by-document matrix T representing the corpus.

2.3.2 Similarity of Sentences and Documents

The combination of word similarity might not reveal how similar of two sentences or two documents, because word is just small unit in sentences or documents. Even word stores significant meaning in sentence and document, its meaning may vary depending on the context and usages. Then, the similarity (relatedness) between two sentences or two documents is still a challenge in NLP because of the meaning of text may vary in different context, or the complex pragmatic of sentences or document depends on the their usages.

From the first stage, the text similarity between sentences or documents can be easy figured out by vectorial representation, then various improvements proposed recently for such techniques towards inventing more sophisticated weighting schemes for the text words, such as TF-IDF and its variations (Aizawa, 2003). Though those techniques achieve certain results, the semantic aspect is still remained to be researched more. Usually, word-to-word similarity can be extended to more general text similarity [9]. Co-occurrence method in word-to-word similarity is extended to pattern matching method [9] which is often used in text mining, this technique relies on the assumption that documents are more similar if

[3] http://www.wikipedia.org/

they contain more words in common. The word, in its turn, is also considers in concepts aspect, or the semantic similarity of words rather than the lexical similarity.

A measure of relatedness between text segments must take into account both the lexical and the semantic relatedness between words. *Omiotis* [41], a thesaurus-based similarity method exploits only a word thesaurus in order to devise implicit semantic links between words, which measure of semantic relatedness between texts which capitalizes on the word-to-word semantic relatedness measure (SR) and extends it to measure the relatedness between texts. Other approach employs the sentence syntax as *SyMSS* [33] to measure the similarity for short texts. SyMSS captures and combines syntactic and semantic information to compute the semantic similarity of two sentences. Semantic information is obtained from a lexical database and through a deep parsing process that finds the phrases in each sentence.

Despite the research has been doing for a long time, text semantic similarity also need to be considered to other languages beside English, such as Asian languages. Among of Asian languages, Chinese is perhaps the language under the focus of research while its other sibling languages such as Japanese and Korean have not been considered on aspect of similarity.

2.4 Keyphrase Extraction

Keyword and keyphrase are word or a multi-words (or term) that describe the content of documents. From now on, we mention to both of them as keyphrase generally. When the readers read a whole document, the remaining ideas in their mind are usually some keyphrases representing for the general and main information of the document. In other words, keyphrases give the reader a brief summary or main ideas of the document. Keyphrase extraction provided as essential step in many task of NLP such as document classification, clustering and summarization. There are two main approaches to extract keyphrases: supervised and unsupervised.

2.4.1 Supervised Keyphrase Extraction

Supervised approach for extracting keyphrases require training data. A typical work in supervised approach is *GenEx* algorithm [43] regards extraction as a classification task, in which a document is treated as a set of phrases, and apply a train model to determine whether a candidate phrase is significant to be keyphrase or not based on statistical and linguistic feature. Other proposed a procedure for keyphrase extraction based on the naive Bayes learning scheme [13]. In addition, some work on generating *table-of-contents* [4] [45] capture both the global dependencies across different tittles and local constrains with a section of document

A disadvantage of supervised approach is it need a considerable amount of training data. In real world, the annotated data does not always available for training phase, especially when the information is continuously increased and vary in many domains, the task of labelling data becomes a time-consuming and costly task. On the other hand, the

increasing of information lead to the need of extracting important information. So, it is necessary to extract information from document even in case the annotated data is not available.

2.4.2 Unsupervised Keyphrase Extraction

The first approach appear in our mind when thinking about unsupervised approach to extract keyphrases is to count and find the most frequent phrases and consider them as the keyphrases representing for document. The approach using *term frequency - inverse document frequency (TF-IDF)* [8] to extract keyphrases achieved some certain results. The term frequency (TF) takes the role as the measurement of how importance of a term in document, and the inverse document frequency (IDF) measures the important of a term in a collection of documents. The combination of TF-IDF measure the significant of a term in a document provide that it is included in a specific corpus, that it can get the context of the corpus.

Recently, in unsupervised approaches, the state of the art method is graph-based ranking with the first work is TextRank [32]. The variation of TextRank is SingleRank and ExpandRank [46]. This kind of algorithm first build a graph from candidate words in document, then measure the important of these words by calculating the scores. After that, the high ranked candidate words are considered as the keyphrase of the document. To extract keyphrases form a document, a typical system consists of three steps:

(i) Candidate words selection. This step choose potential key words from documents using heuristics such as removing stop words [29], choosing words with specific part-of-speech tags (e.g., nouns, verbs, adjectives) [28] [32];

(ii) Cadidate words ranking. When the list of candidate words is obtained, the relations between words should be design in order to rank these words and get the important candidate words. The relations can be the co-occurrence of candidate words in a fixed-size window, in a document, or in a corpus;

(iii) Keyphrase formation. In this step, top ranking words are collapsed to form the phrases.

Others consider extracting keyphrases as clustering task [28], and the exemplars of clusters are the keyphrases representing for the document. The clustering-based keyphrase extraction algorithm first filters out the stop words from a given document and treats the remaining unigrams as candidate words. Then, for each candidate word, compute its relatedness by co-occurrence with a window in size W or by statistics metrics. After that, candidate words are clustered based on their mutual relatedness.Finally, the exemplars or the centres for clusters resulted by clustering step are treated as the keyphrases of given document. The unsupervised method provide us a promising option to extract keyphrases from document without spending a lot of time on annotating the data for training. It has another advantage that the number of keyphrases is not fixed, but it depends on the length of document.

Chapter 3

Approach

In this chapter, we introduce our proposed approach to generate hierarchical table of indexes. A hierarchical table of indexes (HTI) is similar to a table of indexes appearing at the back of books, which help the readers tracing to sections relating to given key words. Beside the role as a representation the summary (or the overview, general ideas) of documents in structural organization, HTI also helps readers quickly refer to the sections containing their interested key words.

Hierarchical in HTI means the set of indexes in lower tier contain more specific information than the higher one. Given a collection of related documents in a specific field, our target is generating a hierarchical table of indexes. Because of involving multiple sources of information, the challenges we have to face is not only the *overlapping* of the ideas between documents, but some parts of a document can *supplement* to some parts of another one, or the content between documents becomes *contradiction* to each other. To treat such problems may occur, the redundancy of information should be identified as well as recognized the novelty in the content; also, the output summary should be ensure to be coherent and complete. The HTI is generated under assumptions:

(i) One segment should be belong to (only) one topic;

(ii) Two segments are in the same topic if they are semantic related.

To achieve the target, we proposed a framework involving to three steps as illustrated in Fingure 3.1 for more intuitively:

1. Segment all documents into separate segments based on topics;

2. Construct hierarchical tree of segments (HTS);

3. Extract key-phrases from each segments, and generate HTI from HTS.

To make the process automatic, all three steps in the framework are unsupervised algorithms. We assumed the text segmentation in first step is available with various of existing algorithms such as TextTiling [18], C99 [7], TextSeg [44], LCSeg [16], MinCutSeg [30], BayesSeg [12], which are introduced in Section 2.1. These text segmentation

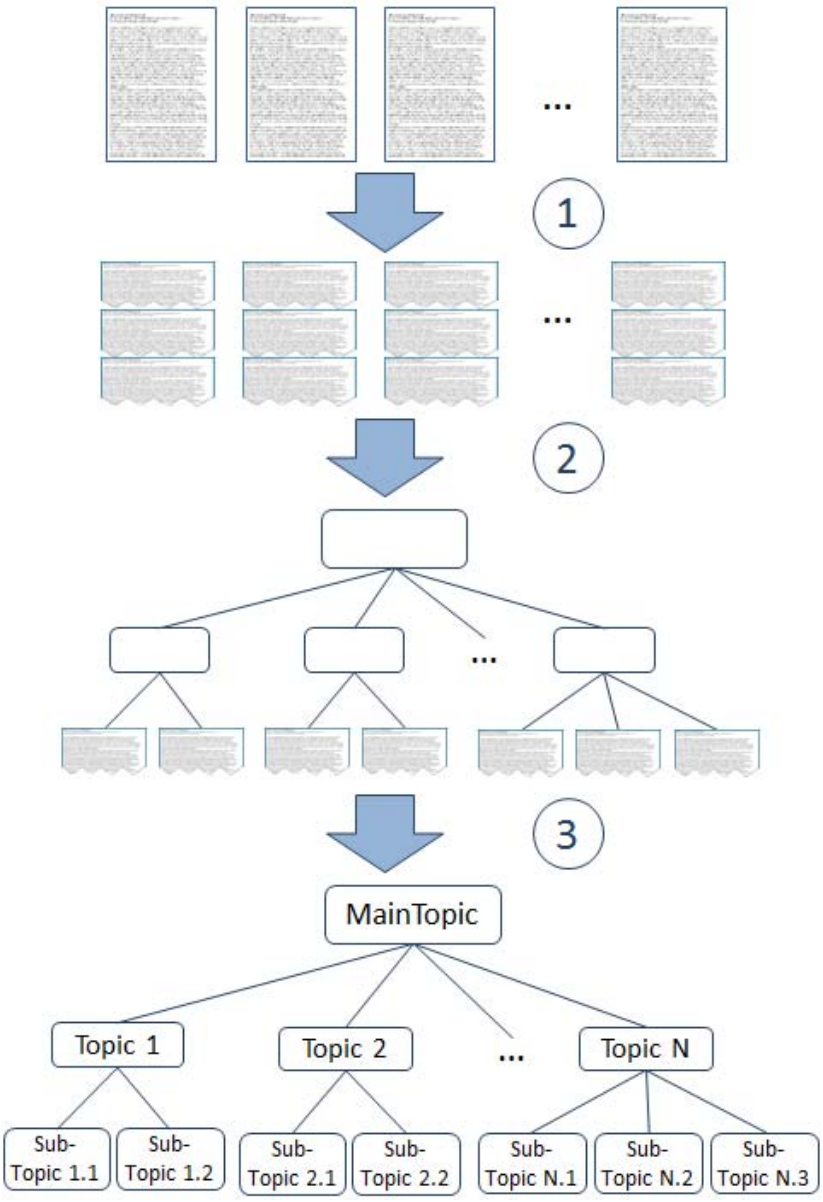


Figure 3.1: The illustration of three steps in proposed framework

algorithms aim to partition text document into coherent segments, where each segment refers to distinguished topics comparing to its adjacent segments. So that, when applying segmentation algorithms on a (collection of) documents, the output will be a set of segments.

We mainly focus on two other steps of the framework. The constructing of a hierarchical tree of segments is described in Section 3.1. Then, the description of how to generate hierarchical table of indexes will be given in Section 3.2.

3.1 Construct Hierarchical Tree of Segments (HTS)

At first, the constructing of Hierarchical Tree of Segments (HTS) may be done by combining segments into composite tree in order to reflect the internal hierarchical structure, using Hierarchical Agglomerative Clustering (HAC) algorithm. HAC algorithm merges two clusters until all clusters have been merged into a single cluster [8]. However, the structure of HAC is not "natural" when apply to construct HTS. Even if "flatten" the HAC dendrogram (Figure 3.2) to get the hierarchical structure more natural, because HAC algorithm does not require that all objects within a cluster be similar to a single center, two segments that should not belong to the same cluster may be grouped together by an unfortunate sequence of pairwise grouping [14].

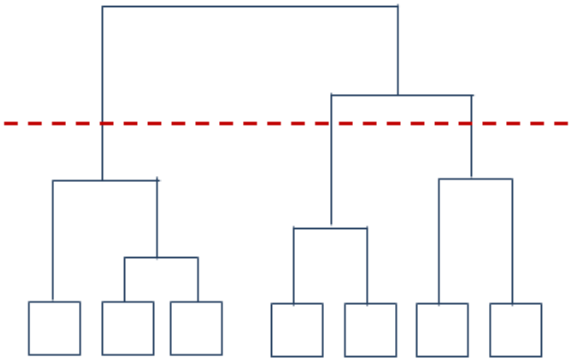


Figure 3.2: The HAC algorithm with flatten technique

Because the above reasons, we employ another clustering technique which is *Affinity Propagation* [14] to clustering the segments. Affinity Propagation takes input measure of similarity between pairs of data points. Real-valued messages are exchanged between data points until a high-quality set of *"exemplars"* (centres of clusters) and corresponding clusters gradually emerges. The advantage of Affinity Propagation clustering algorithm is not only self-determine the number of clusters, but also the data points are grouped in more natural clusters by considering the distance between all data points to a single centre.

The key idea to construct Hierarchical Tree of Segments is: divide the set of segments into groups based on its semantic similarity until all groups can not be divided. We visualize this idea in the Figure 3.3.

Let consider the set of segments obtained from first step to be a set of data points. The HTS is constructed by modelling the set of data points as a weighted graph, and dividing the set of data points of graph into clusters based on their semantic distances; then, each cluster will be re-divided into smaller pieces until it cannot be splitted; the sub-clusters obtained will form lower tier of hierarchical tree structure as described in details in Algorithm 1.

We demonstrate this idea in Figure 3.4 for more visually. The original set of data points originally be viewed as a cluster, it will be first divided into three clusters, then

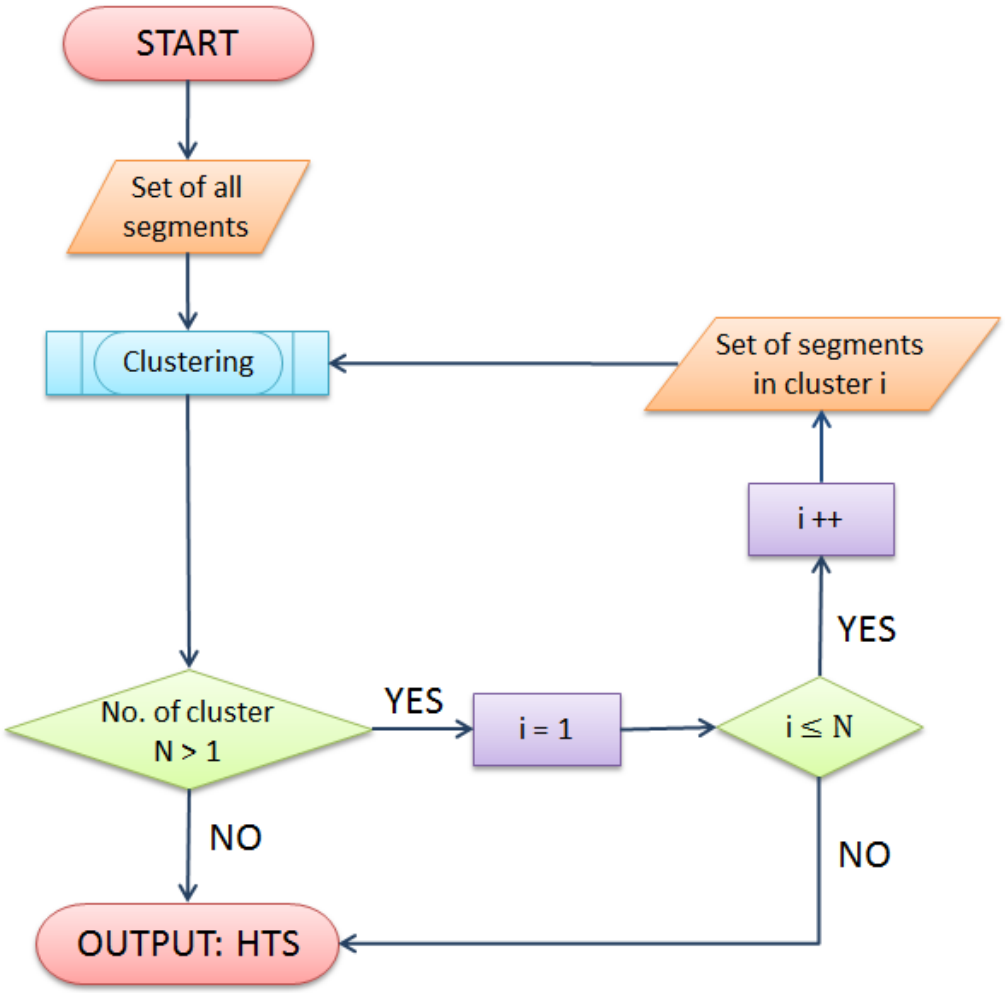


Figure 3.3: The flowchart of constructing Hierachcial Tree of Segments (HTS)

cluster 1 and cluster 2 are continuously divided until we cannot divide any cluster into smaller clusters. As a result, the division process form the hierarchical structure of the data points based on the relation between data points as in Figure 3.5. In other word, this process construct the hierarchical structure of the set of segments based on semantic relations between them.

Technically, the graph is modelled as a triple of $G = (V, E, W)$, where:

(i) $V = \{v_1, ..., v_N\}$ is set of vertices, each vertex represent a segment;

(ii) $E \subseteq V \times V$ is set of edges;

(iii) $W = (w_{ij})_{i,j=1,...,N}$ is adjacency matrix, with each element w_{ij} is the weight of edge between two vertices v_i and v_j and $w_{ij} \in [0, 1]$.

The weight of edge means how related between two data points, or the semantic similarity [9] of two segments, the more related of two segments the more higher similarity it will

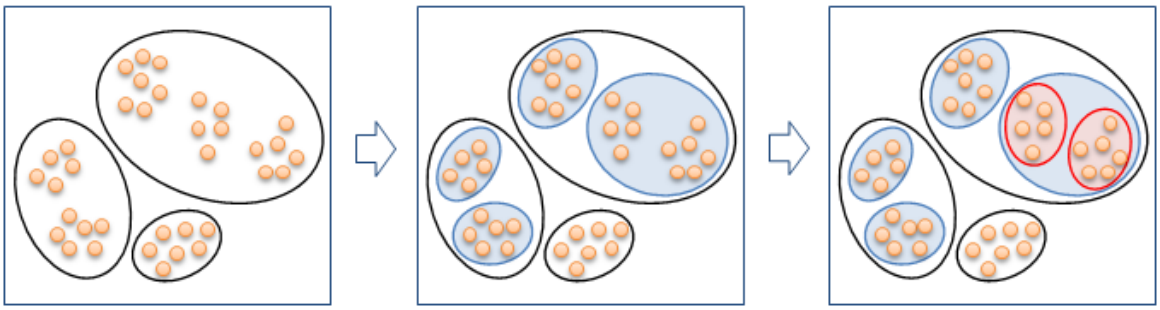


Figure 3.4: Constructing hierarchical tree of segments based on clustering approach

be. To cluster the set of segments, the Affinity Propagation algorithm [14] is applied, so that the number of clusters is drawn by algorithm automatically, and the segments within a cluster are all similar to a center segments.

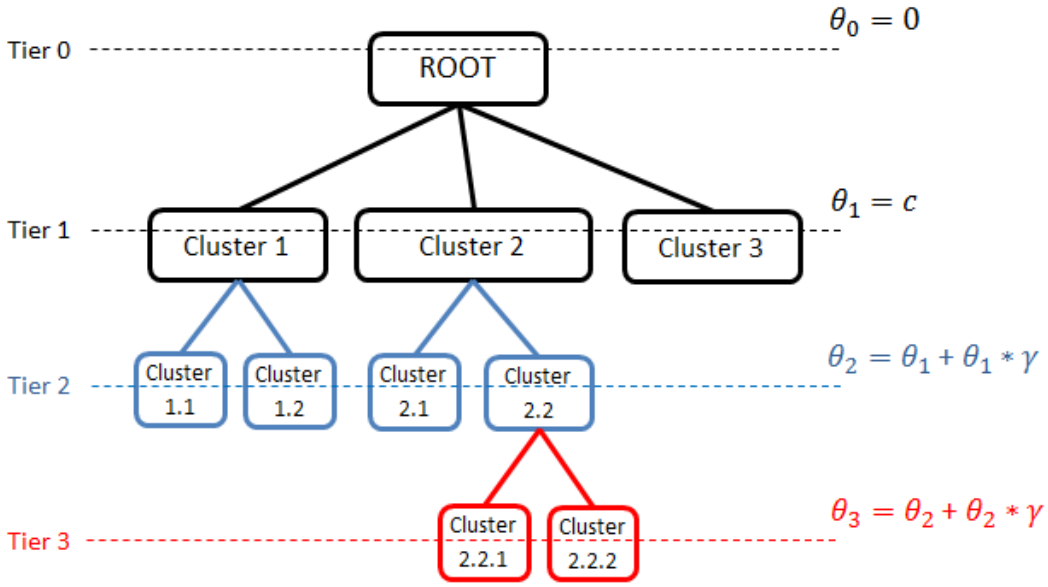


Figure 3.5: The structure of hierarchical tree of segments

At each tier, a threshold θ is applied to cut off the edge weights smaller than it, so that it will make the ideas in lower tier become more distinguishable than the higher one. In details, the completed graph at initial tier 0 is divided into some clusters; then, graphs corresponding to clusters are formed in which edges' weight lower than θ are removed. At tier 0, threshold $\theta = 0$ means all relations between segments are remained. At tier 1, threshold θ is initialized to $\theta = c$. From now on, threshold θ at child node is larger than its ancestor by adding a percentage $\gamma \in [0, 1]$ to its ancestor's threshold. The recursive process will be stopped when the graph cannot be divided into clusters. As the result, the depth and the width of HTS are drawn automatically. In output, the root node of HTS includes branch node(s), the branch node contains other branch nodes or leaf nodes, and the leaf node includes segments.

Algorithm 1: Construct Hierarchical Tree of Segments

input : Graph $G = (V, E, W)$, Threshold θ
output: Hierarchical Tree of Segments

Clustering graph G to get a set of clusters C;
if *number of clusters in* $C > 1$ **then**
    $\theta_k = \theta + \theta \times \gamma$; `// constant` γ `is increasing coefficient`
    foreach *cluster* $C_k \in C$ **do**
        Construct sub-graph $G_k = (V_k, E_k, W_k)$, where $V_k \in C_k$ and $e_{ij} \in E_k \mid i, j \in V_k, w_{ij} \geq \theta_k$;
        $ConstructHTS(G_k, \theta_k)$;
    end
end

As the result, the Affinity Propagation algorithm gives a proper solution to the constructing of the hierarchical structure in our problem, where the segments "overlapping" in content (meaning) will be grouped together by similarity measures, and sub-clusters are served as the "supplement" for the content of clusters.

3.2 Build Hierarchical Table of Indexes (HTI)

In previous work, the table of contents (TOC) [4] [45], which appear at the beginning of every book, is explored and considered as a hierarchical summary for single document by generating sentences from text segments. Those work capture both the global dependencies across different tittles and local constrains with a section of document. Although those work result quite good TOC in fixed length, they need a lot of training data which is not always available when change application to other domains. Especially, when apply to multiple documents, the *overlapping* and *supplement* of content are not considered, caused by using hierarchical segmentation [4] and assumption of available hierarchical structure of content. Another work captures the hierarchical structure of word [23] but, it outputs single word only, though the combination of words (*keyphrases*) is used more natural in real world. So, this thesis proposed an approach to extract keyphrases from multiple documents using unsupervised technique.

The hierarchical table of indexes (HTI) is build agglomeratively from the HTS in previous step using ranking approach [32]. This tasks includes two phases:

(i) Extract keyphrases from each segment in leaf nodes, and consider these keyphrases as indexes for corresponding segments.

(ii) Select representing keyphrases of a cluster among keyphrases of its segments or sub-clusters.

For more specifics, each segment in the leaf nodes of HTS is modelled as a graph to extract keyphrases. All graphs of a leaf node are then merged into a graph, then extract

keyphrases representative for the merged graph. The merged graph of a leaf node, in its turn, will be merge to other graphs to form bigger graph until reach the root node. To make clear, we visualize this idea in flowchart in Figure 3.6 and in Algorithm 2.

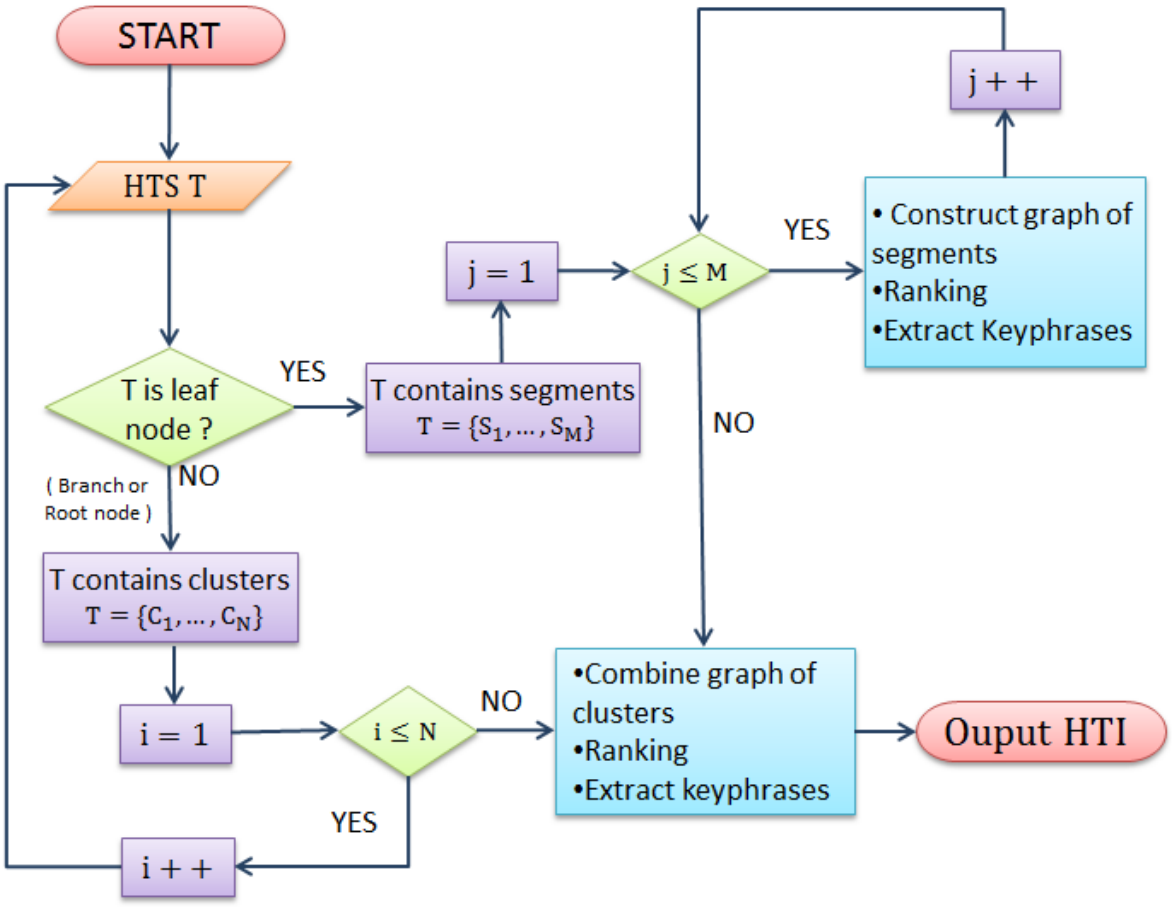


Figure 3.6: The flowchart of building Hierarchical Table of Indexes

In the research context, keyphrase means a word or a group of words standing together and containing significant information of the segment. Usually, in English text, keyphrases of news or technical materials are the combination of adjective and noun [citation] which contain meaningful concepts. But, in the context of this research, the keyphrase may include verb, article (a, an, the), preposition (of, on, in ...) or conjunctions (and, or, ...). For example an English keyphrase "a balance sheet and profit and loss statement" corresponding to Japanese keyphrase "貸借照表及び損益計算書",where the conjunction *"and"* in English and "及び" in Japanese does not load any meaning but the link of concepts in the keyphrase. In addition, even there is no conjunction, keyphrase "損益計算書" means "profit and loss statement" still contains conjunction *"and"* in its meaning.

3.2.1 Extract Keyphrases from Single Segments

To extract keyphrases from segment, the segment is modelled as a graph $G = (V, E)$, where:

(i) V=$\{words \in S\}$ is set of vertices which are candidate words from segment text; and

(ii) E is the relation between vertices determining by sliding a co-occurrence window in size of N on the text, there will be a relation between two vertices if they occur in the window.

The candidate words are those contain meaning such as nouns, verbs, adjectives, adverbs, numbers. When get the graph of candidate words and their relations, compute the vertex weights in graph with following formula until convergence [1] [9]:

$$WS(V_i) = (1-d) + d \times \sum_{V_j \in In(V_i)} \frac{WS(V_j)}{\sum_{V_k \in Out(V_j)} w_{jk}}$$

Where, d is damping factor and usually set $d = 0.85$; $In(V_i)$ and $Out(V_i)$ are in-degree and out-degree of vertex V_i or the number of edges go from and point to vertex V_i respectively. In this case, $In(V_i) = Out(V_i)$ because graph in use is undirected. The vertex weights will served as the rank of vertices, so then candidate keyphrases are extracted by combining high ranked key words with the dependencies of words in sentence. These candidate keyphrases also be added to graph as vertices, with relations to candidate words they contain (illustrasted in Figure 3.7). This graph is ranked again, and keyphrases for each segment are the top ranked vertices from the graph.

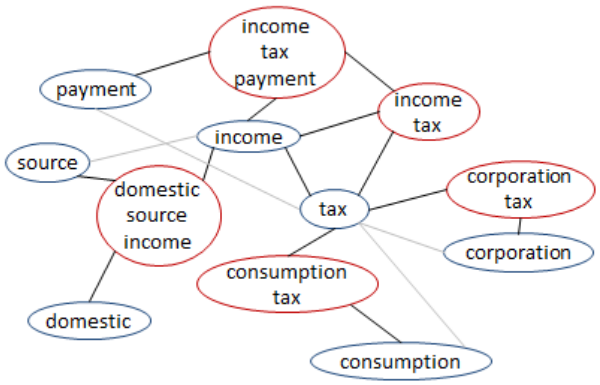


Figure 3.7: Constructing graph to extract keyphrases from a segment.

[1] The convergence is achieved when the change of vertices' weight (or the value of $S^{k+1}(V_i) - S^k(V_i)$) in graph falls below a given threshold, falls bellow the given threshold.

3.2.2 Extract keyphrases from Clusters

From HTS, a cluster may include other sub-clusters or segments. The approach to extract keyphrases from a cluster including many sub-clusters is similar to those including many segments. The idea to extract is choosing high ranked keyphrases from the graph combined from children's graph recursively. In the hierarchical structure, clusters including segments are leaf nodes, those including other sub-clusters but be included in another cluster are branch nodes, and the biggest cluster is the root node. The nearer a node to root the more general of keyphrases will be, and the more nearer to leaf node the more specific of keyphrases of the node is.

To extract keyphrases of a cluster which includes many segments, all graphs modelled from segments are merged together, then compute the vertices' weight of combined graph, after that extract top ranked vertices as keyphrases for the cluster. When combine two graphs, the two sets of vertices from two graphs are aggregated into one set. To ensure the value of a vertex is unique in the graph, in the union set of vertices, if two vertices have the same value (the words it contains are the same), these vertices will be collapsed into one, and their corresponding edges from two identical vertices are connected to the unique one. The complete process to extract keyphrases from a single segment and from a cluster to build up HTI is described in Algorithm 2 specifically.

Algorithm 2: Construct Hierarchical Table of Indexes

input : Hierarchical Tree of Segment HTS T
output: Hierarchical Table of Indexes HTI

if *T does not contain child node* **then** `// tree node` T `is leaf node`
  foreach *segment* $S \in T$ **do**
    Construct graph $G = (V, E)$, where V=$\{words \in S\}$, E=$\{e_{ij} \mid i, j \in V$ and i, j appear in co-occurrence window$\}$;
    Compute all vertices' weight until convergence;
    Rank graph vertices by descending order of vertices' weight;
    From S extract keyphrases K=$\{$combination of words appear at p% top ranked in graph with sentence dependencies$\}$;
    Construct new graph $G' = (V', E')$, where $V' = V + K$ and E'=$\{e_{ij} \mid i \in V, j \in K$ and $\exists w \in j, w = i\}$;
  end
else `// tree node` T `is root or branch node`
  From graphs $(G_i = (V_i, E_i))_{i=1,\ldots,N}$ constructed from the children of T , construct graph $H = (V_h, E_h)$, where $V_h = \{V_1 \cup \cdots \cup V_N\}$, $E_h = \{E_1 \cup \cdots \cup E_N \cup E\}$ with $E = \{e_{ij} \mid \exists w \in j \in E_p, \exists v \in j \in E_q, w = v\}$;
end
Compute all vertices' weight until convergence;
Rank all graph vertices again;
Extract top ranked vertices to be keyphrases for the current node;

We argue that, by combining the graph together, the content overlapping between segments (clusters) which is important will appear in the output by ranking process. Furthermore, the keyphrases in lower tiers content more details information than those in higher tiers, so that, the content of a cluster at lower tier are considered as supplement for the content of its ancestor.

Compare to Original TextRank

To extract keyphrases in original TextRank [32], documents should be scanned twice, the first time is to find the candidate words and their relationships, the second time is to take context when collapsing candidate words to form keyphrases. Assume that in a cluster, all segments content is concatenate into one to apply TextRank for extracting keyphrases of cluster. When using the original technique to build HTI, it will take $2 \times$ *the height of HTS* times scanning over documents to construct graph and extract keyphrases.

In contrast, the proposed method need to scan over all segments twice only. Moreover, merging vertices whose value are the same in different graph still get the relation of words, or in another words, the the context of words still remain when ranking to get the keyphrases for cluster. For those reasons, we believe that proposed merging graph technique is more efficient than the original TextRank when apply to extract keyphrase for long text or multiple documents.

Chapter 4

Experiments

Data Preparation

Experiment data is Pension Law in both Japanese and English version collected from Japanese Law Translation [1]. The data includes 315 documents filed in 12 categories. The documents collected for experiment are in XML format.

4.1 Experiment Setup

The experiment is performed by following steps:

1. Preprocess data for experiment: parse data from XML format to text format
2. In first step, all documents are then be segmented based on topics.
3. In the second step, construct HTS with approach described in Section 3.1
4. In the last step, keyphrases are extracted from the segments of leaf nodes of HTS to generate HTI using approach described in Section 3.2.

Step 0: Preprocessing

The text of documents are read from XML documents (Figure 4.1) and parsed to text file (Figure 4.2). Note that the title and heading of documents are omitted from text document, and each sentence is placed in a line of text file.

Step 1: Text Segmentation

In first step, all documents are then be segmented by TextSeg [44], which is an unsupervised text segmentation method. The boundaries betweem segment in output is marked by Choi notation (==========) as Figure 4.3.

[1] `http://www.japaneselawtranslation.go.jp/`

Figure 4.1: A part of XML file used in experiment

The purpose of this Act is to take necessary measures in connection with the import, export, and recovery of stolen cultural property in order to ensure proper implementation of the Convention on the Means of Prohibiting and Preventing the Illicit Import, Export and Transfer of Ownership of Cultural Property (hereinafter referred to as the "Convention").

In this Act, the term "cultural property" means domestic cultural property and property which a foreign government that is a State Party to the Convention (hereinafter referred to as a "foreign government") has designated pursuant to Article 1 of the Convention.

.....

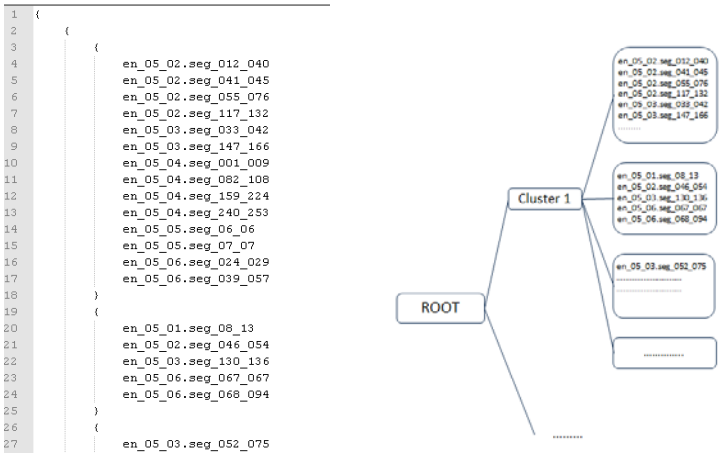
Figure 4.2: A part of text file parsed from XML file

==========
The purpose of this Act is to take necessary measures in connection with the import, export, and recovery of stolen cultural property in order to ensure proper implementation of the Convention on the Means of Prohibiting and Preventing the Illicit Import, Export and Transfer of Ownership of Cultural Property (hereinafter referred to as the "Convention").In this Act, the term "cultural property" means domestic cultural property and property which a foreign government that is a State Party to the Convention (hereinafter referred to as a "foreign government") has designated pursuant to Article 1 of the Convention.In this Act, the term "domestic cultural property" means property which is among items belonging to the categories that are enumerated in (a) through (k) of Article 1 of the Convention and has been designated as Important Cultural Property pursuant to the provisions of Paragraph 1 of Article 27 of the Act for the Protection of Cultural Properties (Act No. 214 of 1950), as Important Tangible Folk Cultural Property pursuant to the provisions of Paragraph 1 of Article 78 of that Act, or as a Historic Site, Place of Scenic Beauty, or Natural Monument pursuant to the provisions of Paragraph 1 of Article 109 of that Act.
==========
....
==========
In the instance described in the first sentence of the preceding paragraph, the victim shall compensate the possessor for the price paid for the said property.The Government of Japan shall endeavor, through educational, public awareness and other activities, to deepen public understanding of the prevention of the illicit import, export, and transfer of ownership of cultural property and also to obtain public cooperation in this regard.
==========

Figure 4.3: A part of text being segmented

Step 2: Construct Hierarchical Tree of Segments

In the second step, we model complete graph $G = (V, E, W)$ from set of segments, with vertices are candidate words (e.g., nouns, verbs, adjectives, adverbs, cardinals), and edge weight is text similarity computed in semantic aspect as described in [9] (see Appendix A). Next, we apply Algorithm 1 and existing unsupervised clustering algorithm Affinity Propagation [14] (see Appendix B) to construct HTS. Initial threshold $\theta = 0.5$ and increase coefficient $\gamma = 50\%$. Figure 4.4a shows a part of HTS which is understood as represented in 4.4b.



(a) Output of HTS (b) Visualization of HTS

Figure 4.4: Two messages used in Affinity Propagation

Step 3: Build Hierarchical Table of Indexes

In the last step, keyphrases are extracted from the segments of leaf nodes of HTS to generate HTI using Algorithm 2 (see Figure 4.5).

```
<?xml version="1.0" encoding="UTF-8"?>
<HIERARCHICAL_TABLE_OF_INDEXES>
  <HIERTABLEINDEX Level="0">
    <KEYPHRASES>
      <Keyphrase Position="en_08_03_10_11,en_08_03_04_09,">foreign cultural property</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">foreign cultural property pursuant</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">domestic cultural property pursuant</Keyphrase>
      <Keyphrase Position="en_08_03_01_03,en_08_03_04_09,en_08_03_12_13,">cultural property</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">foreign affairs pursuant</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">cultural affairs</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">foreign government</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">foreign governments</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">foreign affairs</Keyphrase>
      <Keyphrase Position="en_08_03_01_03,en_08_03_04_09,">article</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">foreign trade act</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">foreign exchange</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">import approval pursuant</Keyphrase>
      <Keyphrase Position="en_08_03_01_03,en_08_03_04_09,">cultural properties</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">minister</Keyphrase>
      <Keyphrase Position="en_08_03_01_03,en_08_03_12_13,">export</Keyphrase>
      <Keyphrase Position="en_08_03_12_13,">instance</Keyphrase>
      <Keyphrase Position="en_08_03_12_13,">japan</Keyphrase>
      <Keyphrase Position="en_08_03_12_13,">government</Keyphrase>
      <Keyphrase Position="en_08_03_12_13,">first sentence</Keyphrase>
      <Keyphrase Position="en_08_03_01_03,en_08_03_10_11,">act no.</Keyphrase>
      <Keyphrase Position="en_08_03_01_03,">transfer</Keyphrase>
      <Keyphrase Position="en_08_03_12_13,">illicit import</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">technology</Keyphrase>
      <Keyphrase Position="en_08_03_04_09,">notification</Keyphrase>
      <Keyphrase Position="en_08_03_10_11,">years</Keyphrase>
```

Figure 4.5: The output of HTI

4.2 Evaluation

The evaluation of experiment is judged by human. The experiments will be evaluated by two criteria:

i How good the summary is;

ii How quick the relevant sections can be navigated to.

Evaluation of the Summary

To evaluate the summary, specifically the quality of extracted keyphrases, we illustrate the proposed approach on a document named *Act on Controls on the Illicit Export and Import and other matters of Cultural Property* in category of *Education and Culture*.

The experiment is run in both English and Japanese version of law. A piece of the result in Japanese is illustrated in the Figure 4.6, where the leftmost box is the root containing the highest ranked indexes for all, it is decomposed into branches with more specific indexes added into lower tier. The demonstration in English version is provided in Appendix C.

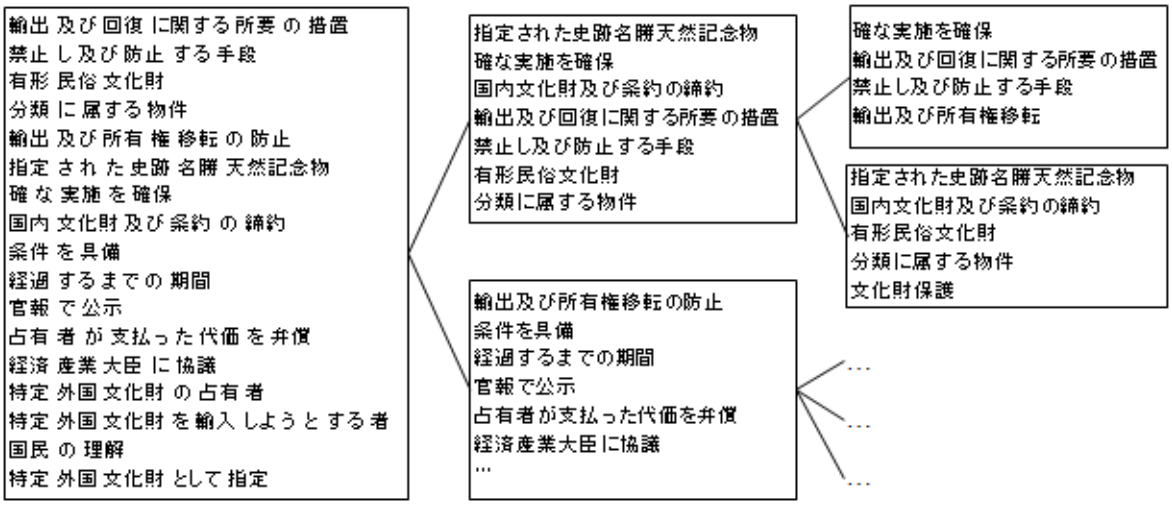


Figure 4.6: The illustration of output for Japanese law

Table 4.1 describes the result and manual evaluation on the generating of HTI for the given document. In each language, the total number of keyphrases is the number of all keyphrases in HTI generated by proposed approach. The HTI is then showed to human and get the respond of which keyphrase in HTI is acceptable or it is important to the main ideas of the text. Because of the characteristics of language, the phrase in English and Japanese content different kind of part-of-speeches, it causes an approximately 5% different between the rate of English and Japanese.

Language	Total number of keyphrases	Keyphrases accepted by human	Precision
English	150	68	45.3%
Japanese	96	39	40.6%

Table 4.1: Result of keyphrases extraction

Evaluation of Navigation

The evaluation of the navigation on hierarchical tree is implemented in English version, we use 08 documents from category *Tax and Financial affairs*, which are list in Table 4.2.

No.	Law Name
1	National Tax Collection Act
2	Act on General Rules for National Taxes
3	Income Tax Act(Limited to the provisions related to nonresidents and foreign corporations)
4	Order for Enforcement of the Income Tax Act(Limited to the provisions related to nonresidents and foreign corporations)
5	Ordinance for Enforcement of the Income Tax Act(Limited to the provisions related to nonresidents and foreign corporations)
6	Corporation Tax Act(Limited to the provisions related to foreign corporations)
7	Order for Enforcement of the Corporation Tax Act(Limited to the provisions related to foreign corporations)
8	Ordinance for Enforcement of the Corporation Tax Act(Limited to the provisions related to foreign corporations)

Table 4.2: List of laws used in evaluation of navigation

We provide a questionnaire of five questions (Figure 4.8), each question is following by four choices. The questionnaire is distributed to two groups of people, one look up the answers for questions in original plain text in PDF format using search function (Control + F) using PDF reader; the other look up the answers by navigating on the extracted keyphrases on the root level (Figure 4.7).

The result of this survey is shown in Table 4.3. From this result, we see that the average time to search the interest sections in HTI is 9.5 minutes. This result is very competitive to 13.8 minutes of using original documents and search linearly.

	On original document	On HTI
Average time	13.8 minutes	9.5 minutes

Table 4.3: Avarage time to find answer when search on original text and HTI

national taxes	application income tax subject
tax office	delinquent tax collection procedure
national tax	tax prepayment
tax payment grace period	coal tax
income tax payment	aviation fuel tax
consumption tax	power-resources development tax
district director	life insurance contract
payable tax	retirement pension fund
stamp tax	.....

Figure 4.7: Some extracted keyphrases in the root node (in English)

QUESTIONS

1. *Benificiary certificate* is entrusted with:

☐ Japanese government ☑ Business office located in Japan

☐ Business office ☐ Others

2. How long is the maximum time of *Tax Payment Grace Period*

☑ 1 year ☐ 2 years

☐ 6 months ☐ 3 months

3. Which one is the *tax rate* for *corporation tax* in case of *Foreign Corporation*

☐ 5% ☐ 10%

☐ 20% ☑ 30%

4. *Foreign trust company* is a company which is

☑ A *trust company* that is a *foreign corporation*

☐ A *trust company* located in a foreign country

☐ A *trust company* which head office located in foreign country, and branchs located in Japan

☐ A company whose *trust property* belong to a *foreign corporation*

5. *Delinquent taxpayer* may be granted a *grace period* by

☐ The *tax agent*

☑ The *district director* of the tax office

☐ Business office where taxpayer works

☐ Others

Figure 4.8: The questionnaire to evaluate the navigation of HTI (with answers)

Chapter 5

Conclusions and Future Work

Conclusions

The thesis studies an automatic process to generate a hierarchical table of indexes for multiple documents. The main contributions are:

1. Introduced an unsupervised framework to generate hierarchical table of indexes for multiple documents;

2. Proposed an approach to construct hierarchical tree of segments by unsupervised clustering algorithm with the depth and wide of the hierarchical structure is drawn automatically;

3. Proposed an approach to extract keyphrases from multiple text segments by combining sentence dependencies and graph rank based method algorithm, and then generate hierarchical table of indexes from the tree of segments.

On the aspect of the content, the problem of *overlapping* in content, which multi-document summarization is usually encounter, is solved because the segments with similar topic are grouped together. In addition, the hierarchical structure, where the keyphrases in lower tier is more details, also solves the problem of *supplement* in content between (part of) documents. The remain challenge which is *contradiction* is need more study on it.

The experiment is applied for both English and Japanese in contribution to Legal Engineering. The preliminary result of summary is provided as the illustration for our approach. And searching information on the hierarchical summary is evaluated better than searching on original plain documents.

We realize that because of the different in the transformation in word surface of languages, the English keyphrases can be the combination of *adjective + noun*, where *adjective* can be the form of *V-ing* or *V-ed*. And the Japanese keyphrases can be the combination of *verb + noun*, *verb +の + noun* etc,.

Future Work

Though considering to semantic aspect of words by using WordNet, the text similarity metric [9] that is used to calculate the distance between the segments is a little bit slow, cause by linear matching block of text from a pair of segments and by looking up for semantic distance of words from WordNet.

So far, many existing methods in NLP are mainly focused on processing English. However, every language or specific data have its owns characteristics, and those methods are not suitable or they need some modifications when being applied to other languages. It is obvious that language-dependent software may cause some limitations. Topic model may be applicable to overcome the limitations. Topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. By applying statistics, this approach gives us advantages:

(i) Firstly, it does not need much previous knowledge of language (language independence);

(ii) Secondly, it may be adaptable to the change of the language or data set (data driven).

To meet such needs, in future work, we plan:

1. To apply statistical model to constructing the hierarchical structure and combine the supportive knowledge into extracting keyphrases while generating the table of indexes;

2. To explore another text similarity will be explored to find a faster computation metric for text similarity;

3. To consider approach to treat the contrasting of information between (parts of) documents.

Bibliography

[1] Doug Beeferman, Adam Berger, and John Lafferty. Statistical models for text segmentation. *Mach. Learn.*, 34(1-3):177–210, February 1999.

[2] David M. Blei and Pedro J. Moreno. Topic segmentation with an aspect hidden markov model. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 343–348, New York, NY, USA, 2001. ACM.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[4] S. R. K. Branavan, Pawan Deshpande, and Regina Barzilay. Generating a table-of-contents. In *Proc. of ACL '07*, pages 544–551, Prague, Czech Republic, June 2007.

[5] Zheng Chen and Heng Ji. Graph-based clustering for computational linguistics: a survey. In *Proceedings of the 2010 Workshop on Graph-based Methods for Natural Language Processing*, TextGraphs-5, pages 1–9, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[6] Jen-Tzung Chien and Chuang-Hua Chueh. Topic-based hierarchical segmentation. *IEEE Transactions on Audio, Speech & Language Processing*, 20(1):55–66, 2012.

[7] Freddy Y. Y. Choi. Advances in domain independent linear text segmentation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, NAACL 2000, pages 26–33, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[8] Prabhakar Raghavan Christopher D. Manning and Hinrich Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[9] Courtney Corley and Rada Mihalcea. Measuring the semantic similarity of texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, EMSEE '05, pages 13–18, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[10] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[11] Jacob Eisenstein. Hierarchical text segmentation from multi-scale lexical cohesion. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North*

American Chapter of the Association for Computational Linguistics, NAACL '09, pages 353–361, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[12] Jacob Eisenstein and Regina Barzilay. Bayesian unsupervised topic segmentation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 334–343, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[13] Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. Domain-specific keyphrase extraction. In *IJCAI*, pages 668–673, 1999.

[14] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[15] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 1606–1611, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[16] Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 562–569, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[17] M.A.K Halliday and Ruqayia Hasan. *Cohesion in English*. Longman, London, 1976.

[18] Marti A. Hearst. Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL '94, pages 9–16, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

[19] Marti A. Hearst. Texttiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64, March 1997.

[20] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM.

[21] Hideki Kozima. Text segmentation based on similarity between words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, ACL '93, pages 286–288, Stroudsburg, PA, USA, 1993. Association for Computational Linguistics.

[22] T. K. Landauer and S. T. Dutnais. A solution to platos problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240, 1997.

[23] Dawn Lawrie, W. Bruce Croft, and Arnold Rosenberg. Finding topic words for hierarchical summarization. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 349–357, New York, NY, USA, 2001. ACM.

[24] Dawn J. Lawrie and W. Bruce Croft. Generating hierarchical summaries for web searches. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 457–458, New York, NY, USA, 2003. ACM.

[25] C. Leacock and M. Chodorow. *Combining local context and WordNet sense similiarity for word sense disambiguation*. The MIT Press, US, 1998.

[26] Michael E Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM.

[27] Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[28] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 620–628, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[29] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 257–266, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[30] Igor Malioutov and Regina Barzilay. Minimum cut model for spoken lecture segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, Sydney, Australia, July 2006. Association for Computational Linguistics.

[31] Charles T. Meadow, Bert R. Boyce, and Donald H. Kraft. *Text Information Retrieval Systems*. Academic Express, second edition, 2000.

[32] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *EMNLP*, pages 404–411, 2004.

[33] Jesús Oliva, José Ignacio Serrano, María Dolores del Castillo, and Ángel Iglesias. Symss: A syntax-based measure for short-text semantic similarity. *Data Knowl. Eng.*, 70(4):390–405, April 2011.

[34] Rebecca J. Passonneau and Diane J. Litman. Intention-based segmentation: Human reliability and correlation with linguistic cues. In *ACL*, pages 148–155, 1993.

[35] Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the 4th international conference on Computational linguistics and intelligent text processing*, CICLing'03, pages 241–257, Berlin, Heidelberg, 2003. Springer-Verlag.

[36] Matthew Purver, Thomas L. Griffiths, Konrad P. Körding, and Joshua B. Tenenbaum. Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 17–24, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[37] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, IJCAI'95, pages 448–453, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[38] Jeffrey C. Reynar. Statistical models for topic segmentation. In *ACL*, 1999.

[39] G. Salton and M. E. Lesk. Computer evaluation of indexing and text processing. *J. ACM*, 15(1):8–36, January 1968.

[40] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, August 1988.

[41] George Tsatsaronis, Iraklis Varlamis, and Michalis Vazirgiannis. Text relatedness based on a word thesaurus. *J. Artif. Int. Res.*, 37(1):1–40, January 2010.

[42] Peter D. Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, pages 491–502, London, UK, UK, 2001. Springer-Verlag.

[43] Peter D. Turney. Learning to extract keyphrases from text. *CoRR*, cs.LG/0212013, 2002.

[44] Masao Utiyama and Hitoshi Isahara. A statistical model for domain-independent text segmentation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, pages 499–506, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics.

[45] Nguyen Viet Cuong, Nguyen Le Minh, and Shimazu Akira. Learning to generate a table-of-contents with supportive knowledge. In *IEICE Transactions on Information and Systems*, pages 423–431, Japan, March 2011.

[46] Xiaojun Wan and Jianguo Xiao. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, AAAI'08, pages 855–860. AAAI Press, 2008.

[47] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

[48] Lu Zhiqiang, Shao Werimin, and Yu Zhenhua. Measuring semantic similarity between words using wikipedia. In *Proceedings of the 2009 International Conference on Web Information Systems and Mining*, WISM '09, pages 251–255, Washington, DC, USA, 2009. IEEE Computer Society.

Appendix A

Semantic Similarity of Text

We describe the metric used to compute the semantic similarity of texts [9] when constructing HTS. Given pair of segments, the algorithm to compute the text similarity is as follows:

1. Create the *sets of open-class-words* for nouns, verbs, adjectives, adverbs, and cardinals;

2. Determine pair of similar words across the set corresponding to the same open-class in the two segments.

   - For each noun (verb) in the set of nouns (verbs), try to identify the noun (verb) in the other segment that has the highest semantic similarity *maxSim* using word similarity described in Section 2.3.
   - The similarity of other word classes: adjectives, adverbs, cardinals is computed using lexical similarity.

3. Compute *directional* measure of similarity of each segment with respect to the other segment using this formula:

$$sim(T_i, T_j)_{T_i} = \frac{\sum_{pos}(\sum_{w_k \in \{WS_{pos}\}} (maxSim(w_k) \times idf_{w_k}))}{\sum_{w_k \in \{T_{i_{pos}}\}} idf_{w_k}}$$

4. The similarity between two segments $sim \in [0, 1]$ is the bidirectional similarity calculated by average function:

$$sim(T_i, T_j) = \frac{sim(T_i, T_j)_{T_i} + sim(T_i, T_j)_{T_j}}{2}$$

Appendix B

Affinity Propagation Clustering Algorithm

Affinity Propagation clustering algorithm [14] is unsupervised clustering approach which self-determine the number of clusters. Affinity propagation takes as input a collection of real-valued similarities between data points, where the similarity $s(i,k)$ indicates how well the data point with index k is suited to be the exemplar for data point i. There are many ways to determine the similarity $s(i,k)$, depend on the purpose:

(i) When the goal is to minimize squared error, each similarity is set to a negative squared error (Euclidean distance): For points x_i and x_k, $s(i,k) = -\|x_i - x_k\|^2$.

(ii) If exemplar-dependent model available: $s(i,k)$ can be set to the log-likelihood of data point i given that its exemplar is data point k.

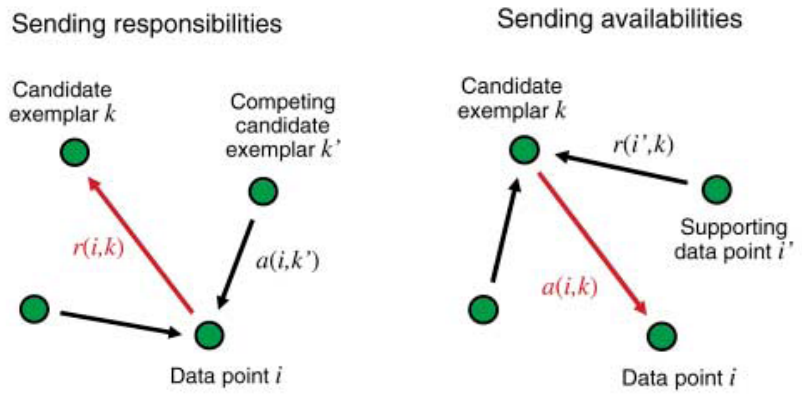
(iii) Set similarity by hand.

Rather than requiring that the number of clusters be pre-specified, affinity propagation takes as input a real number $s(k,k)$ for each data point k so that data point with larger value of $s(k,k)$ are more likely to be chosen as exemplars. These values are referred to as *"preferences."* The number of identified exemplars (number of clusters) is influenced by the value of the input preferences, but also emerges from the message-passing procedure.

There are two kinds of message exchanged between data points, and each takes into account a different kind of competition. Messages can be combined at any stage to decide which points are exemplars and, for every other point, which exemplar it belongs to. Two messages are:

(i) The *"responsibility"* $r(i,k)$, sent from data point i to candidate exemplar point k, reflects the accumulated evidence for how well-suited point k is to serve as the exemplar for point i, taking into account other potential exemplars for point i (Figure B.1a)

$$a(i,k) \leftarrow s(i,k) - \max_{k\prime s.t. k\prime \neq k} (a(i,k\prime) + s(i,k\prime))$$



(a) The responsibility $r(i,k)$ (b) The availability $a(i,k)$

Figure B.1: Two messages used in Affinity Propagation

(ii) The *"availability"* $a(i,k)$, sent from candidate exemplar point k to point i, reflects the accumulated evidence for how appropriate it would be for point i to choose point k as its exemplar, taking into account the support from other points that point k should be an exemplar (Figure B.1b).

$$\begin{cases} a(i,k) \leftarrow \min\{0, r(k,k) + \sum\limits_{i's.t.i' \notin \{i,k\}} \max\{0, r(i',k)\}\} & \text{for } i \neq k \\ a(k,k) \leftarrow \sum\limits_{i's.t.i' \neq k} \max\{0, r(i',k)\} & \text{for } i = k \end{cases}$$

At any point during affinity propagation, availabilities and responsibilities can be combined to identify exemplars. For point i, the value k that maximizes $a(i,k)+r(i,k)$ either identifies point i as an exemplar if $k = i$, or identifies the data point that is the exemplar for point i.

The message-passing procedure may be terminated after a fixed number of iterations, after changes in the messages fall below a threshold, or after the local decisions stay constant for some number of iterations. When updating the messages, a damping factor λ is added to each message to avoid numerical oscillations may arise in some cases:

$$a^{t+1}(i,k) = (1-\lambda) + \lambda \times a^t(i,k)$$

$$r^{t+1}(i,k) = (1-\lambda) + \lambda \times r^t(i,k)$$

In the experiment, damping factor is set $\lambda = 0.5$ by default. The availabilities of all points are initialized as 0. At each iteration, the affinity propagation process runs the following steps:

1. Update all responsibilities given the availabilities;

2. Update all the availabilities given the responsibilities;

3. Combine the availabilities and responsibility to monitor the exemplar decisions and terminate the algorithm when the decisions do not change for $T = 10$ times.

Fingure B.2 illustrates the Affinity Propagation on 25 two-dimensional data points, using negative Euclidean distance as similarity metric.

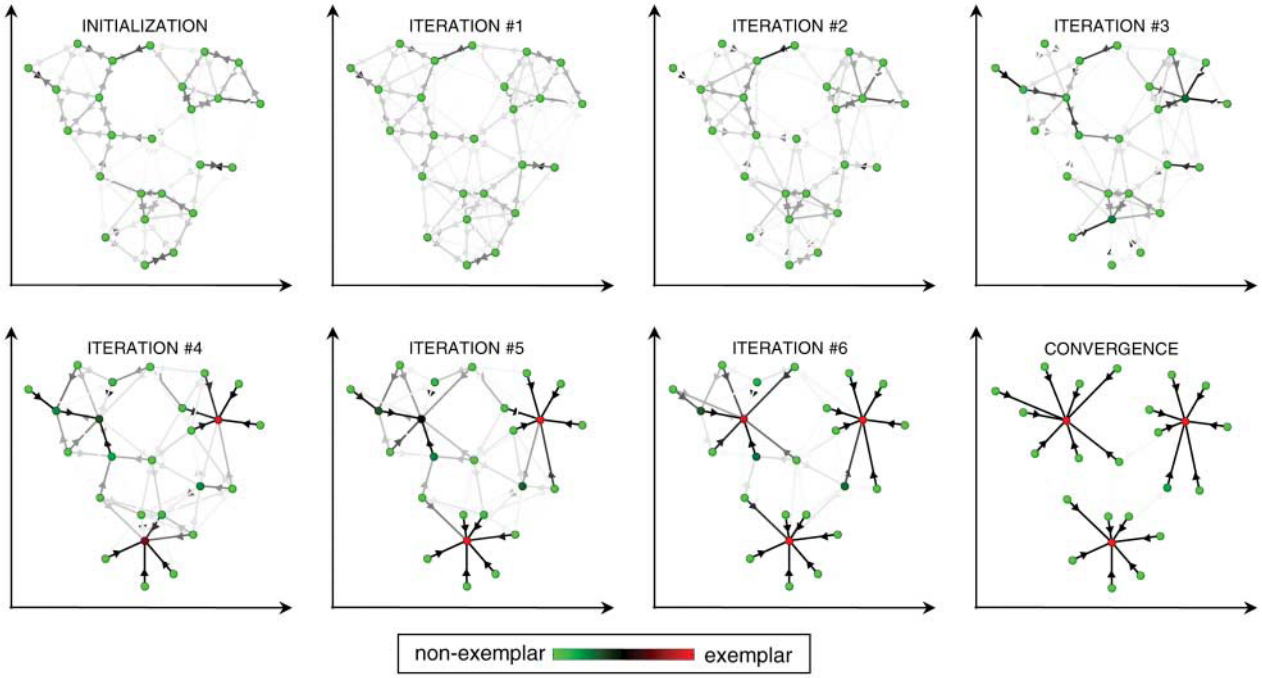
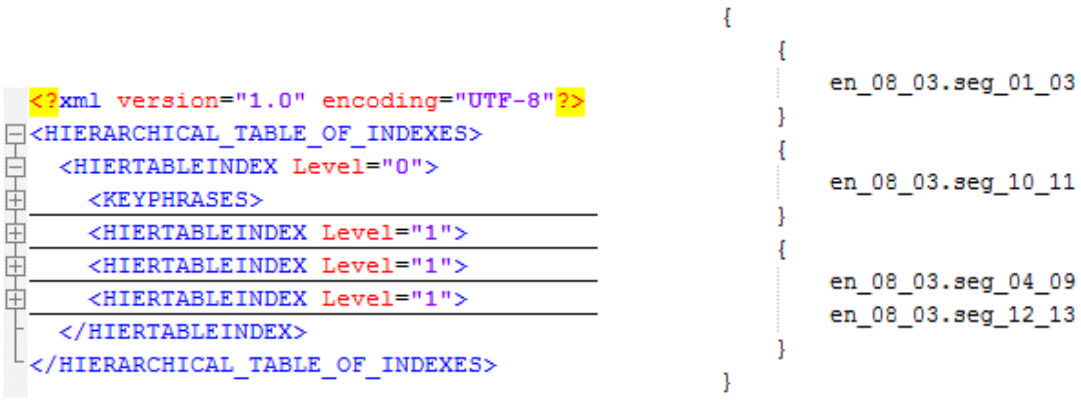


Figure B.2: The illustration of Affinity Propagation

Appendix C

Output of Hierarchical Table of Indexes

This Appendix provides an example in English version for output HTI of document *Act on Controls on the Illicit Export and Import and other matters of Cultural Property* in category of *Education and Culture*.



(a) In XML view (b) In tree view

Figure C.1: The overall look of output HTI

```
<HIERTABLEINDEX Level="0">
  <KEYPHRASES>
    <Keyphrase Position="en_08_03_10_11,en_08_03_04_09,">foreign cultural property</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign cultural property pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">domestic cultural property pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,en_08_03_04_09,en_08_03_12_13,">cultural property</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign affairs pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">cultural affairs</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign government</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign governments</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign affairs</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign trade act</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign exchange</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">import approval pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,en_08_03_04_09,">cultural properties</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">minister</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,en_08_03_12_13,">export</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">instance</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">japan</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">government</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">first sentence</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,">transfer</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">illicit import</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">technology</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">notification</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,">illicit import</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,">important cultural property pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,">natural monument pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">content</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,">convention</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">possessor</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">inclusive</Keyphrase>
    <Keyphrase Position="en_08_03_01_03,">important tangible folk cultural property</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">connection</Keyphrase>
  </KEYPHRASES>
```

Figure C.2: The keyphrases of HTI at the root node (tier 0)

```
<HIERARCHICAL_TABLE_OF_INDEXES>
  <HIERTABLEINDEX Level="0">
    <HIERTABLEINDEX Level="1">
      <KEYPHRASES>
        <Keyphrase Position="en_08_03_01_03,">natural monument pursuant</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">important cultural property pursuant</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">cultural properties</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">act no.</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">cultural property</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">important tangible folk cultural property</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">transfer</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">convention</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">article</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">illicit import</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">export</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">protection</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">proper implementation</Keyphrase>
      </KEYPHRASES>
      <KEYPHRASES Level="1">
        <Keyphrase Position="en_08_03_01_03,">natural monument pursuant</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">important cultural property pursuant</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">convention</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">cultural properties</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">cultural property</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">important tangible folk cultural property</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">illicit import</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">export</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">act no.</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">transfer</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">scenic beauty</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">article</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">protection</Keyphrase>
        <Keyphrase Position="en_08_03_01_03,">proper implementation</Keyphrase>
      </KEYPHRASES>
    </HIERTABLEINDEX>
```

Figure C.3: The keyphrases of HTI at first branch (of tier 1)

```
<HIERTABLEINDEX Level="1">
  <KEYPHRASES>
    <Keyphrase Position="en_08_03_10_11,">civil code</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">foreign cultural property</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">possessor</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">years</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">total period</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">conditions</Keyphrase>
  </KEYPHRASES>
  <KEYPHRASES Level="1">
    <Keyphrase Position="en_08_03_10_11,">civil code</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">years</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">foreign cultural property</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">possessor</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">recovery</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">article</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">total period</Keyphrase>
    <Keyphrase Position="en_08_03_10_11,">conditions</Keyphrase>
  </KEYPHRASES>
</HIERTABLEINDEX>
```

```
<HIERTABLEINDEX Level="1">
  <KEYPHRASES>
    <Keyphrase Position="en_08_03_04_09,">foreign affairs pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign cultural property pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">domestic cultural property pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign cultural property</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign affairs</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">cultural affairs</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">import approval pursuant</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign government</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign trade act</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,en_08_03_12_13,">cultural property</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">foreign exchange</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">minister</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">first sentence</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">prevention</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">government</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">japan</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">technology</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">instance</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">inclusive</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">notification</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">illicit import</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">effect</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">cultural properties</Keyphrase>
    <Keyphrase Position="en_08_03_12_13,">export</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">sports</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">culture</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">content</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">connection</Keyphrase>
    <Keyphrase Position="en_08_03_04_09,">application mutatis mutandis</Keyphrase>
  </KEYPHRASES>
  <KEYPHRASES Level="1">
  <KEYPHRASES Level="1">
</HIERTABLEINDEX>
```

Figure C.4: The keyphrases of HTI at the second and third branch (of tier 1)

Publications

[1] Le Thi Ngoc Tho, Nguyen Le Minh, Akira Shimazu. A Study on Hierarchical Table of Indexes for Multi-documents. JapTAL 2012.