An Improved Lexical Chain Method for Automatic Text Summarization using Semantic Related Terms

단어 의미적 연관성을 고려한 개선된 어휘체인기반의
자동 문서요약 방법

2016년 8월 25일

조선대학교 대학원

컴퓨터공학과

Htet Myet Lynn

An Improved Lexical Chain Method for Automatic Text Summarization using Semantic Related Terms

지도교수 김 판 구

이 논문을 공학석사학위신청 논문으로 제출함.

2016년 8월

조선대학교 대학원

컴퓨터공학과

Htet Myet Lynn

Htet Myet Lynn의 석사학위논문을 인준함

위원장 조선대학교 교수 정일용 (인)

위 원 조선대학교 교수 정현숙 (인)

위 원 조선대학교 교수 김판구 (인)

2016년 8월

조선대학교 대학원

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

An Improved Lexical Chain Method for Automatic Text Summarization using Semantic Related Terms

Htet Myet Lynn
Advisor : Prof. Pankoo Kim, Ph.D
Department of Computer Engineering
Graduate School of Chosun University

Summarization is a challenging task that need to understand the content of the document to determine the importance information of the text. Automatic Summarization is the procedure of lessening a text document using an intelligent system with complex algorithms to form a short summary of it which retains the most important information of the text. Lexical cohesion is a way identifying connected portions of the text according to the relations between the words in the document or text. Lexical cohesive relations between words in a document can be described using lexical chains. Lexical chains are applied in various Natural Language Processing (NLP) and Information Retrieval (IR) applications. The fundamental task to perform in constructing lexical chain is to extract the best appropriate candidate keywords of the text as the chains which contain the salient portions of the document are relied on them.

Thus, we extend our research on automatic keyword extraction for extracting candidate terms of the given text before approaching to summarization. Keywords are a set of keywords or keyphrases that capture the primary information or topic discussed in the text. Keywords are widely used to define queries in IR systems as they are easy to define, revise, remember, and share. Furthermore, keywords are essential Search Engine Optimization (SEO) elements

for every search engines, yet they are matched against with users' search query keywords. They can empower document browsing by providing a short summary, improve information retrieval, and be employed in generating indexes for a large text corpus.

In current thesis, we proposed a new approach for automatic text summarization using lexical chain with semantic relatedness keywords. In contrast, the new method of extracting keywords is also implemented for constructing the efficient lexical chain. Instead of constructing a lexical chain with every noun phrases in the text, the result of our experimental results show that the extracted candidate keywords of the text delivers a better efficient summary with a better performance. Thus, we have built a system to extract the promising keywords from the text which is based on the characteristics of the manually assigned keywords. The system consists of a generator to produce the possibility distributions of three distinct features of assigned keywords; the occurrence of a term in title, the occurrence of a term in a first sentence, and the higher score of average TF.IDF score of a term. Then, those distributions are applied to Markov chain process of three stages to assign the label for each N-gram term in the text. Extracted unigram candidate keywords are selected to build the lexical chains of the text. There are also three distinct relation criteria to connect the terms according to their relationships to other candidate keywords using WordNet; hypernym, hyponym and synonym. Then we apply the method to score and extract the salient portions of the document. Our experimental results prove that the efficient summaries can be extracted for the above tasks.

요약

단어 의미적 연관성을 고려한 개선된 어휘체인기반의 자동 문서요약 방법

Htet Myet Lynn
Advisor : Prof. Pankoo Kim, Ph.D
Department of Computer Engineering
Graduate School of Chosun University

문서를 요약한다는 것은 그 문서의 일관성을 유지하면서 중복을 제거하고, 응축된 정보를 생산하는 것을 말하며, 자동문서 요약 기술은 컴퓨터를 사용해서 문서 내 중요한 부분을 유지하고, 중복된 내용을 제거함으로써 처리하고자 하는 대용량의 문서를 자동적이고 효율적으로 처리하는 방법을 말한다.

“어휘 결합(Lexical Cohesion)”은 어휘의 관계(상하 어휘관계, 유의 어휘관계 등)를 바탕으로 하나의 문서 내의 등장하는 단어와 단어 사이의 관계를 분석하는 방법이다. 이러한 “어휘 결합 관계”는 어휘 사슬(Lexical Chain)을 이용하여 나타낼 수 있다. 어휘 사슬은 자연언어처리(Natural Language Processing) 및 정보검색(Information Retrieval)기술에 다양하게 활용되고 있으며, 적절한 후보키워드를 추출하는 것이 프로그램의 성능을 좌우하기 때문에, 어휘 사슬을 구성하기 위해 적절한 후보키워드를 추출하는 것이 가장 중요한 작업이라고 할 수 있다.

본 연구는 단어의 의미적 연관성을 고려하여 구성한 후보키워드의 어휘사슬을 기반으로 개선된 자동문서 요약방법에 관한 연구로, 효율적인 어휘 사슬을 구성하기 위해 새로운 키워드추출방법을 제안하였다. 본 논문에서 문서 내의 키워드 추출을 위해 “제목에 등장하는 단어”, “문서의 첫 번째 문장에 등장하는 단어”, “TF-IDF 가중치가 높게 측정되는 단어” 세 가지의 키워드 특징을 정의하였으며, 키워드가 갖는 조건부 확률 값을 활용해 전이 행렬(transition matrices)을 생성함

으로써, 마르코프 연쇄(Markov Chain)에 적용을 통해 후보키워드를 추출한다.

추출된 후보키워드는 워드넷(WordNet) 상에서 정의된 단어의 상하위어 관계, 동의어 관계를 고려하여 후보 키워드 간의 연결을 통해 어휘 사슬을 구성하였으며 이를 통해 자동문서 요약을 수행하게 된다. 본 논문의 실험결과에 따르면, 제안한 방법에 의해 추출한 후보키워드로 어휘 사슬을 구성하는 것이 문서 내 모든 명사구에 대한 어휘 사슬을 구성하는 것보다 향상된 성능을 보였으며, 더욱 효율적으로 문서를 요약할 수 있음을 증명하였다.

Ⅰ. INTRODUCTION

The first chapter states the clarification of the motivation of this thesis, and it will illustrate the actuality and aim of the thesis. Finally, Section B gives a brief overview of the following chapters.

A. Motivation

Today's world is all about online information on the internet across the globe. The World Wide Web comprises billions of documents, and it is growing at an incredible pace. Applications that provide timely access to, and digest of, various sources are required in order to lessen the information overload users are encountering. These obstacles have sparked interest in the development of automatic summarization systems. Such systems are designed to take a single article or multiple articles, a cluster of news articles, or an email thread as input, and execute a concise and efficient summary of the most important information. Recent years have been the development of numerous summarization applications and tools for news, email threads, and professional medical information, scientific articles, spontaneous dialogues, and videos. These systems have already been shown to assist people, and to enhance other automatic applications and boundaries.

B. Outline

The outline of th thesis is organized as follows:

Chapter II analyses the problem by stating the existing approaches in single document text summarization and automatic keyword extraction. This provides an overview of the related works, and explains the choice of our conceptual design.

Chapter III defines the conceptual design and exploited heuristic. The detailed description of the used statistical measures for our new automatic keyword and construction of Transition Probability Distribution Generator is provided in this section. The following section introduces the development of conventional lexical chain and scoring chain techniques are explained, and the detail information of extracting summary sentences is included.

Chapter IV provides the evaluation results for our proposed keyword extraction method, and the evaluation method called ROUGE is described as we apply it for evaluating our system.

Chapter V gives a brief summary of our proposal and presents our future research directions.

II. BACKGROUND CONCEPTS

A. SUMMARIZATION

The area of summarization has been explored by the NLP society for nearly the last half century. Radev et al. [1] defines that a summary as a short text that is executed from a single or multiple texts, that highlights the important information of the original text(s), and that should not be longer than half of the original text(s), and normally significantly less than the original ones. This straightforward definition catches three important aspects that described research on automatic summarization:

- Summaries may be generated from a single or multiple documents,
- Summaries should maintain notable information,
- Summaries must be short.

Even if we agree on these facts, it seems from the literature that any approach to provide a more elaborate description for the task would end in conflict within the community. In fact, many procedures differ on the manner of their own problem fabrications. We can start by indicating some common terms in the summarization dialect: extraction is the procedure of determining important sections of the text and generating them precisely; abstraction aims to produce important information of the text in a new way; fusion combines extracted parts cohesively, and compression intends to exclude the unimportant parts of the text [1]. Earliest occurrences of research on summarizing scientific documents proposed standards for extracting prominent sentences from text using features like word and phrase frequency [2], position in the text [3] and key phrases[4]. Various attempts published since then has focused on other domains, mostly on news data. Many approaches discoursed the problem by constructing systems depending on the type of the required summary. While extractive summarization is mostly concerned with the summary content

extracted from the original text, and it is normally relying on extraction of sentences, abstractive summarization emphasizes on the form, scheming to produce a new summary with grammatical arrangements which usually demands the advanced language generation techniques. A critical issue that will definitely drive future research on summarization is evaluation. During the last 25 years, many evaluation system competitions like TREC, DUC and MUC have produced sets of training data and have initiated baselines for performance levels. However, ROUGE has become a common scheme to evaluate summarization systems. [50]

The rest of this topic is arranged as follows: the following section describes document summarization, focusing on extractive techniques, where a few extractive approaches that pioneered the field are also considered. Commonly, the flow of information in a given document is not uniform, which means that some parts of the text are more important than others. The major challenge in summarization relies on distinguishing the more informative parts of a document from the less ones. But there have been circumstances of research describing the automatic creation of abstracts, most work presented in the literature depends on verbatim extraction of sentences to distinguish the problem of single-document summarization. In this section, we describe some well known extractive techniques. First, we look at early work from the 1950s and 60s that kicked off research on summarization. Second, we look on approaches including machine learning techniques published in the late 2000s to today [54][55][56]. Finally, we briefly describe some techniques that use a more complex natural language analysis to tackle the problem.

With the advent of machine learning approaches, a series of seminal publications appeared that employed statistical approaches to produce document extracts. While the most systems assumed feature independence and relied on naive-Bayes methods, others have emphasized on the appropriate features, and on learning algorithms which has no independence assumptions. Other crucial approaches involved hidden Markov models and log-linear models to enhance the extractive summarization. In contrast, a very recent paper used neural

networks and third party features (like common words in search engine queries), word co-occurrence, and conditional random fields to produce better purely extractive single document summarization. We next describe all these approaches in more detail.

1. Existing Approaches

(i) Naive Bayes Methods

Kupiec et al. [5] describe a technique based on Edmundson [4] which is able to learn from training data. The classification function classifies each sentence as deserving for extraction or not, using a Naive-Bayes classifier. For example, tet s be a particular sentence, S the set of sentences extracted as summary, and the features. Assuming independence of the features:

$$P(s \in S | F_1, F_2, .. F_k) = \frac{\prod_{i=1}^{k} P(F_i | s \in S).P(s \in S)}{\prod^{k_{i=1}} P(F_i)} \qquad (1)$$

The distinct features were similarly related to Edmundson's method [4], but additionally included the length of sentence and the presence of uppercase words. Each sentence score was scored according to (1), and only the top n sentences were extracted. In evaluating the system, a corpus of technical documents with manual abstracts was used in the following way. For each sentence in the abstract, the researchers analyzed its match with the actual document sentences and created a mapping manually. The system summaries were then evaluated against this mapping. Feature analysis of the system showed that a system using only the position and the features, along with the sentence length as sentence feature to be performed best.

Aone et al. [6] also a naive-Bayes classifier, but with more features. They introduced a system called DimSum that relies on features like term frequency (tf) and inverse document frequency (idf) to derive significant words. The idf was computed from a large corpus of the same domain as the given documents. Statistically acquired two-noun word collocations (bigrams) were applied as units for counting, together with single words. A named-entity tagger was also

applied and each entity was considered as a single token. They also employed some thin discourse analysis like reference to same entities in the text but still maintain the cohesion. Synonyms and morphological variants were merged when considering lexical terms, the former approach being identified by using WordNet [7]. The corpora used in the experiments were from newswire, some of which belonged to the TREC evaluations.

(ii) Hidden Markov Models

With the previous approaches, that were mostly feature-based and non-sequential techniques, Conroy and O'leary[8] introduced the problem of extracting a sentence from a document using a hidden Markov model (HMM). The fundamental motivation for using a sequential model is to discover for local dependencies between sentences. However, only three features were considered, the position of the sentence within the document, number of terms in the sentence, and likeliness of the sentence terms given the document terms.

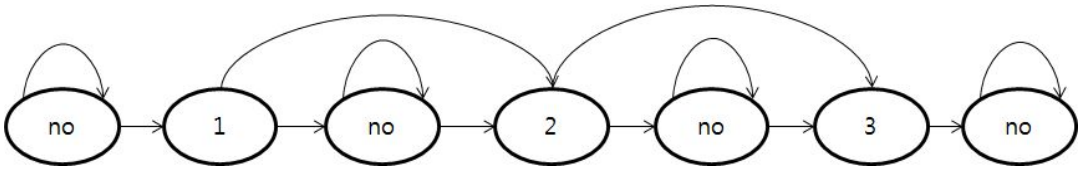

Figure 1.Markov model to extract to three summary sentences from a document(ConroyandO'leary,2001)

The HMM model was constructed as follows: it included 2s + 1 states, changeable between s (summary states) and s+1 (non-summary state). The authors allowed "hesitation" only in non-summary state and "skipping next state" only in summary states. Figure 1 describes an example HMM with 7 nodes, corresponding to s = 3. Using the TREC dataset for training corpus, the authors acquired the maximum-likelihood estimate for each transition

probability, forming the transition matrix estimate M, whose element (i, j) is the empirical probability of transition from state i to j. Combined with each state i was an output function, bi(O) = Pr(O j state i) where O is an observed vector of features. A simplifying assumption was made that the features are multi variate normal. The output function of each state was estimated by using the training data to compute the maximum likelihood estimate of its mean and covariance matrix. They estimated means of 2s+1, but presumed that all of the output functions shared a common covariance matrix. The evaluation was done by comparing with human generated extracts.

(iii) Neural Networks and Third Party Features

In 2001 and 2002, DUC has granted a process of creating a summary with 100 words of a single news article. The systems, however, with the best performance in the evaluations could not surpass the baseline with numerical significance. This highly rated baseline has been analyzed by Nenkova[9] and complies to the selection of the first n sentences of a news article. After 2002, the task of single-document summarization for news articles was lowered from DUC. Svore et al. [10] introduced a technique with neural networks and the use of third party data sets to tackle the problem of extractive summarization, outperforming the baseline with statistical significance.

The authors used a dataset of 1365 documents collected from CNN news web site, which consists of the title, time stamp, three or four human generated story highlights and the article or content. Three machine generated summaries are considered for the task. The human extracted summaries were not verbatim extractions from the article itself. They were evaluated the system using two metrics. The first one linked the three highlights produced by the system, concatenated the three human generated highlights, and compared these two blocks. And the second metric considered the ordering and compared the sentences on an individual level.

B. Lexical Chain

Human summarizers fabricate a constructed mental representation (theme) of the document and integrate the document based on the theme to produce the summary. The conventional computational techniques used word count measure [11] to determine the theme of the document. The motivation of this approach was that frequent words with the critical information of the text. One primary drawback of this proposal is that it does not emphasizes the importance of a word in the given context. Lack of consideration fails to capture the "context" or the "theme" of the document. For example,

1. **"Mr. Jimmy has invented an anesthetic machine. This device restrains the rate at which an anesthetic is pumped into the blood".**
2. **"Mr. Jimmy has invented an anesthetic machine. The doctor spent two years on this research".**

Both texts contain the same frequency of the words "Mr. Jimmy" and "machine", but the first text emphasizes the machine whereas the second one highlights Mr. Jimmy. This distinction can only be made by considering the relation between the words in the text (e.g. machine and device in first text).

The cohesion of Halliday and Hasan [13], enables us to capture the "theme" of the document. It can be determined as the property of the text to attach together as one large grammatical unit, based on relations between words. For example,

3. Wash and core six cooking potatoes.

4. Put them into the dish.

In the above set of sentences, the second sentence refers to the potatoes in the first one. This property of cohesiveness is not visible between un-related sentences. For example,

5.Wash and core six cooking potatoes.

6.Toronto is the biggest city in Canada.

Cohesion relations affect the comprehensibility of the text [14]. The cohesive structure can be described as graphs with elements of the text as the nodes and relations between the elements as edges connecting the nodes. Boldness of the information can then be determined based on the interacted relations of the nodes in the graph.

Halliday and Hasan [13] separated cohesive relations into the following categories:

Reference: reference relations, commonly, include the usage of pronouns in order to refer an entity mentioned in the preceding or the following text. In the following example, he and Mark both refer to the same person "Mark".

7. Mark went to England. He had to attend a conference.

Substitution: the relations in which one particular phrase or word is replaced with an article such as one or several etc. In the following example, "several" is substituted to refer the word "car".

8. I bought a new car yesterday. And there were several I could have had.

Ellipsis: the relations established by rejection of certain phrases or words. In the following example, the word "distant" is not mentioned for the second time.

9. New York is as distant from San Francisco as Boston is [distant] from London.

Conjunction: the relations attained by using connectors to describe the relationships between statements.

10. He gave me the directions but I lost it.

11. When you have done, we shall leave.

1. WordNet 3.1

WordNet 3.1 is a large lexical database of Englisha words that are linked together by their semantic relationships. It is like a supercharged dictionary or thesaurus with a graph structure. Moreover, nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets). Synsets are words interlinked by means of conceptual-semantic and the lexical relations. The subsequent network of relevantly related words and concepts can be directed with the browser. WordNet is freely and openly available for download. The structure of WordNet makes it a useful tool for computational linguistics and natural language processing.

WordNet apparently resembles a thesaurus, which means that it groups words together based on their lexical meanings. However, there are still some important distinctions. Firstly, WordNet interconnects not just word forms, strings of letters, but specific meanings or senses of words. Words, eventually, that are found in close proximity to one another in the network are semantically disambiguate. The second obstacle which WordNet labels the semantic relations among words, whilst the groups of words in a thesaurus do not follow any definite pattern other than similar meanings.

In WordNet, each “word form” represents some explicit lexical "meaning". Some word forms represents several meanings and some meanings can be represented by various forms. Table 1 shows the concept of lexical matrix which used to map word meanings to word forms. The entry 'Ei,j' in the lexical matrix symbolizes that word form 'WFj' refers to the meaning 'Mi'.

Table 1. Mapping between words form and lexical meanings

Word Meaning	Word Forms			
	WF1	WF2	WF3	WFn
M1	E1,1	E1,2		
M2	E2,1		E2,3	
M3		E3,2		
Mj	Ej,1	Ej,2		Ej,n

Word forms representing to the same underlying concept can be determined as synonymous for instance {WF1.WF2}. WordNet arranges the word forms belonging to the same syntactic category that refer to the common underlying concept into synonym sets called synsets. As an example, the synset notation defines its definition as standard, criterion, measure, touchstone, and it refers to the lexical meaning "a basis of comparison". Word forms that refer to more than one underlying concept are called polysemous (WF1). Table 2 represents the number of synsets and number of unique strings, and the total word-sense pairs, and Table 3 indicates number of Monosemous and Polysemous, and their senses, Table 4 shows the number of average polysemy including monosemous and the number of average polysemy without monosemous in WordNet 3.1.

Table 2. Number of Unique Strings, Sysnsets and Total Word-Sense Pairs in WordNet 3.1

POS	Unique Strings	Synsets	Total Word-Sense Pairs
Noun	117798	82115	146312
Verb	11529	13767	25047
Adjective	21479	18156	30002
Adverb	4481	3621	5580
Totals	155287	117659	206941

Table 3. Statistics of words and senses of Monosemous and Polysemous in WordNet 3.1

POS	Monosemous Words and Senses	Polysemous Words	Polysemous Senses
Noun	101863	15935	44449
Verb	6277	5252	18770
Adjective	16503	4976	14399
Adverb	3748	733	1832
Totals	128391	26896	79450

Table 4. The average number of Polysemy with and without Monosemous words

POS	Average Polysemy with Monosemous Words	Average Polysemy without Monosemous Words
Noun	1.24	2.79
Verb	2.17	3.57
Adjective	1.40	2.71
Adverb	1.25	2.50

WordNet connects the synsets by certain lexico-semantic relations in Table 5. The most significant relation is hypernym and hyponym relation for a term, in which a synset is a whole class or member of class of another synset. Hypernym and hyponym relation organizes nouns and verbs into 11 and 512 hierarchies. The hierarchical construction of synsets can be seen in Figure2. Generality of concepts increases while traversing upwards in the hierarchical structure. Figure 3 shows the WordNet entry for the word modification.

Table 5.Sample WordNet Relations

Relation	Examples
Synonym	weather - atmospheric condition
Hypernym/Hyponym	car - vehicle
Antonym	good - bad
Meronym/Holonym	steering - navigation

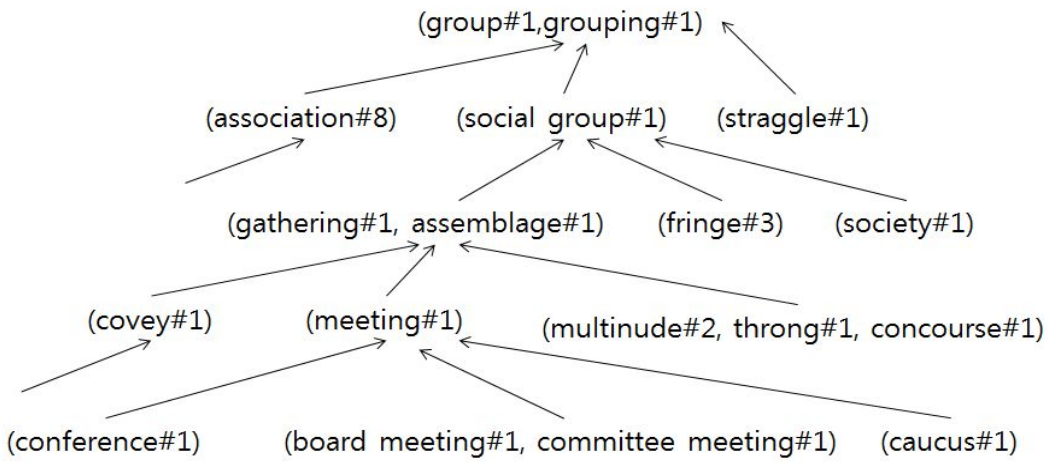

Figure 2. WordNet hierarchical structure

Gloss of each synset contains definition(s), comment(s) and some example(s) for the underlying lexical concept. For example, the gloss of the synset of weather, weather condition, atmospheric condition includes the definitions such as the meteorological conditions: temperature and wind and clouds and precipitation, and that is followed by example sentences such as "they were hoping for good weather", "every day we have weather conditions and yesterday was no exception".

The definitions of the gloss can be used to describe the interacted relations between two concepts not directly related using direct WordNet relations. For example, let us consider the words “dormitory” and “university”. There is no straightforward relation in WordNet between the two words although the relation can be identified by humans. According to the gloss of the word dormitory, the meaning of it is "a college or university building containing living quarters for students", so that we can execute a relation between the two words (we also concede the relation between dormitory and students from the same the semantic definition).

Lesk [15] complied the existence of the gloss concepts of a word in the current surroundings to lessen the sense of the word being used in current

context. Banerjee and Pedersen [16] examined more further and measured the semantic relatedness between two concepts based on their gloss definition overlap within WordNet lexical database. Harabagiuand Moldovan [17] emphasized the related gloss concepts to gather the information not explicitly stated in the text.

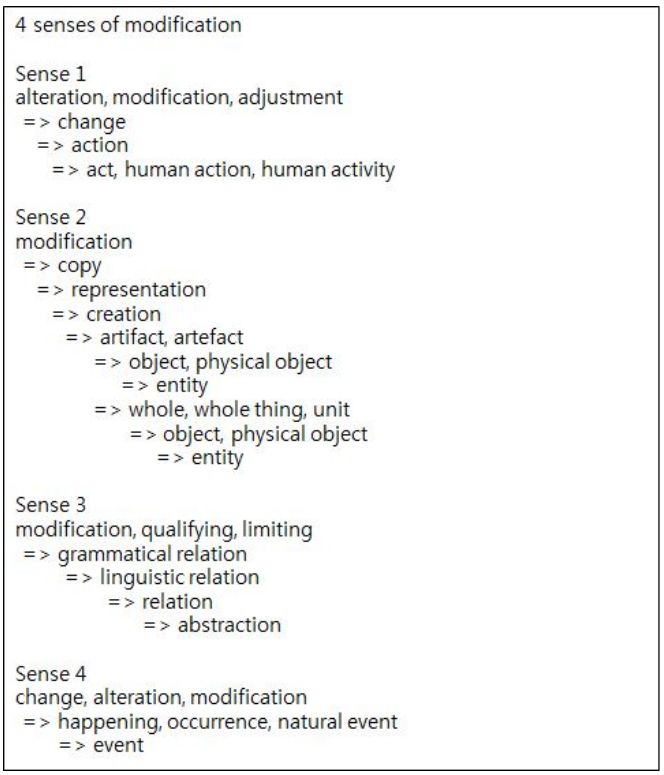

Figure 3. WordNet entry for the word modification

C. AUTOMATIC KEYWORD EXTRACTION

Automatic keyword extraction is the process to identify a small set of words, key phrases, keywords, or key segments from a text or document that defines the meaning of the document [18]. It should be processed by systematically and with either minimal or no human intervention, relies on the model. The goal of automatic keyword extraction is to apply the capacity and performance of

computation to the problems of access and discover ability, adding value to information organization and retrieval without the substantial costs and drawbacks associated with human indexers [19].

1. Motivation

As a result of rapid growth of diffusion of information technology, the spread of stored information across the Internet has been hastily increasing day after day. Large amount of stored information can be found in various forms of data contents such as metadata, text documents, images, audio files and so on. Among those data types, documents are known as unstructured data, and there are several types of documents in divergent ways of writing methods which can be collected from different sources like social media and news websites, or personal blogs. Not only text analytics processes within Natural Language Processing (NLP) to deal with such documents for taking further NLP operations, but also the efficient of common search engines rely on the amount of those collected documents. Keyword extraction plays a vital role in document retrieval, text mining, document categorization or classification, automatic summarizing, indexing, and other processes of deriving high-quality information from text summarily and accurately.

Keywords are a set of the most informative and relevant words or phrases of a document which imply the context of the entire document. Furthermore, keywords are essential Search Engine Optimization (SEO) elements for every search engines, yet they are matched against with users' search query keywords. Also they are extremely momentous for document surrogates built with metadata which contains general information of its respective data objects such as title, abstract or summary and so on. In order that, assigning the right keywords to a document is the main task and the most crucial step to perform further advanced procedures. In case of failure in distributing the right keywords in a first place, the future proceedings will be on tough road with a lot of time consuming and costly to manually extract the right ones from the beginning. There are several methods to extract the keywords from single

document the main concept of the most common technique is scoring each term in a tex twith a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

The proposed method acquires the most significant characteristics (features) of keywords to construct the model and the system relies on the probability distributions of each feature generated by Transition Probability Distribution Generator(TPDG). The TPDG values are calculated based on Naive Bayes' Bayesian probability terminology for each feature for each n-gram term, thus, not only the characteristics of a keyword but its properties can also be learnt by the model, and will be represented in probability statistics values. But then those probabilities are used as transition matrices for each different state where the three distinct features; the TFIDF score of each term over the average score, the occurrence of a term within the first sentence of the text and the occurrence of a term in the title of the text, represent as conditions for a state space (Keyword, Non-Keyword) in a Markov Chain to estimate and label a term can be in either 'Keyword' or 'Non-Keyword' state[58]. The detail description of the proposed method is described in subsections below.

2. Existing Approaches

There are several methods to extract the keywords from single document the main concept of the most common technique is scoring each term in a text with TFIDF score, a numerical statistic which reflects how important a word is to a document in a collection or corpus. The weighting score algorithm has been widely applied to many various text processing approaches as a fundamental mechanism yet enough powerful method to score the important terms which carry the notable information of the text. Term Frequency (TF) indicates the number of occurrences of a term within a text, and Inverse Document Frequency (IDF) is a measure of how much information the word provides, that is, whether, the term is common or rare across all documents,

was proposed by Karen Sparck Jones in 1972. The multiplication of both terms increases proportionally to the number of times a word appears in the document, but is neutralized by the frequency of the word in the corpus to adjust for the fact that some of the terms appear more frequently in general. There were several both supervised and unsupervised approaches taken to the automatic keyword extraction have been proposed.

Existing methods for automatic keyword extraction can be separated in four categories, i.e. simple statistics, linguistics, machine learning and other mixed approaches. Simple statistics approach like N-gram statistical information to automatic index the document [20]. N-gram is language and domain-independent and other approaches include word frequency[21], TFIDF [22], word co-occurrence [23], and PAT-tree [24], etc. Linguistics approaches principally emphasize on sentences and document including the lexical analysis [25], syntactic analysis [26], discourse analysis [27][28][52][53] and etc. Some supervised machine learning approaches for keyword extraction are implemented to learn a model from training set and applies the model to extract the keywords from test documents. Such approaches enclose Naive Bayes [29], SVM [30], etc. The other mixed approaches combine the approaches mentioned above together or use some heuristic knowledge to extract keywords [31].

(i) Linguistic Approaches

The linguistic approaches utilize the linguistic features of the words, sentences or text. Methods which emphasize to linguistic features such as part-of-speech, syntactic structure and semantic qualities tend to increase the value, functioning sometimes as filters for inappropriate keywords.

Plaset al. [32] applied two lexical resources for evaluation, the EDR electronic dictionary, and WordNet of Princeton University. Both provide densely populated lexicons including semantic relationships and links such as IS-A and PART-OF relations and concept polysemy. During automatic keyword extraction process from multiple dialogue episodes, the advantages of using the lexical resources

are compared to a pure statistical method and relative frequency ratio. Hulth [33] has tested a few different methods of incorporating linguistics into keyword extraction. Terms are considered as keywords based on 3 features: the term frequency in a document (TF), collection frequency (IDF), the position of its first occurrence in a document and its part-of-speech tag. The results showed that the use of linguistic features indicate the remarkable improvement of the automatic keyword extraction. However, some of the linguistic methods are combined methods, mixing some linguistic methods with common statistical measures such as term frequency and inverse document frequency.

(ii) Machine Learning Approaches

Automatic Keyword extraction can be seen as supervised learning from the examples or training corpus. The machine learning mechanism works as follows. First a set of training documents is provided to the system, each of which has a range of keywords which are carefully extracted by humans manually. After that, the gained knowledge is applied to find keywords from new documents or test documents depending on the model whish was built according to the result of training section. The Keyphrase Extraction Algorithm (KEA) [34] compromises the machine learning techniques and naive Bayes formula for domain-based extraction of technical keyphrases. Suzukiet al. [35] used spoken language processing techniques to extract keywords from news using an encyclopedia and newspaper articles as a guide for relevance.

The process is divided into two phases: term-weighting and keyword extraction. A set of feature vectors, firstly, is generated from different encyclopedia domains. Then the same procedure is processed on a corpus of newspaper articles. The vectors from encyclopedia are matched with the vectors of the article using a similarity computation in order to distinguish the latter into different domains, after thay which they are separated, producing the final set of feature vectors. In the second phrase, the keyword extraction process, a segment is analysed for the most relevant topic is preferred for it in order to

use the pre-existing feature vectors. The phoneme recognition software is implemented to do the analysis, looking for the best fit between a segment's vectors and that of one of the encyclopedia domains. When the best relevant domain is chosen, its keywords are then assigned to the news segment.

(iii) Mixed Approaches

The other approaches about keyword extraction mainly blend the methods mentioned above or use some heuristic knowledge in the task of keyword extraction, such as the position, length, layout feature of the words, html tags around of the words, etc[36]. The overview of the related works discloses that the automatic keyword extraction is faster and less expensive than human extraction. Moreover, the researchers claim that it achieves the precision of the human extracted keywords. Currently, existing solutions, however, for automatic keyword extraction needs either training examples or domain specific knowledge. Some approaches do not have this additional information. We apply the statistical measures to the automatical keyword extraction as they are domain independent and have limited requirements.

Ⅲ. PROPOSED METHOD & SYSTEM IMPLEMENTATION

Summarization, as carried out by humans, can be divided into 2 stages mentioned in Jones in 1993:

- Building of intermediate representation
- Synthesis of intermediate representation to generate summary

According to the researches, the lexical chains can be used as an efficient intermediate representation for the textual documents. We implemented a system to compute lexical chains as an intermediate representation for the source and to summarize the text in shorten form by extracting satisfied certain sentences. As the distinct nouns and noun phrases of the text are the main factor to construct the efficient lexical chain, we developed a new approach, for extracting and assigning the most relevant keywords/terms of the text, which assigns the keyword or non-keyword state for each term in text and seeks the most important keywords using our Transition Probability Distribution Generator (TPDG) which learns the characteristics of the keywords upon on three distinct features. The detail information of TPDG is discussed in section below. The overview architecture of our system can be seen in Figure 4 and Figure 5.

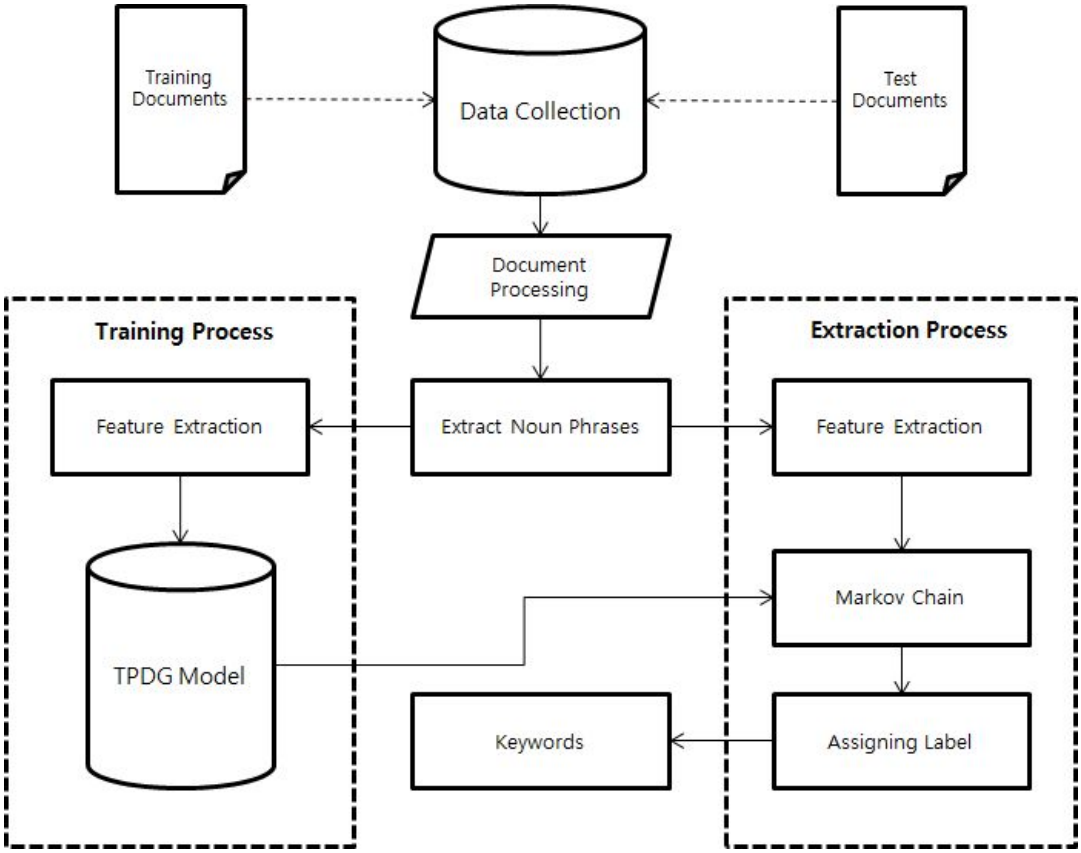

Figure 4.Automatic Keyword Extraction using TPDG Model

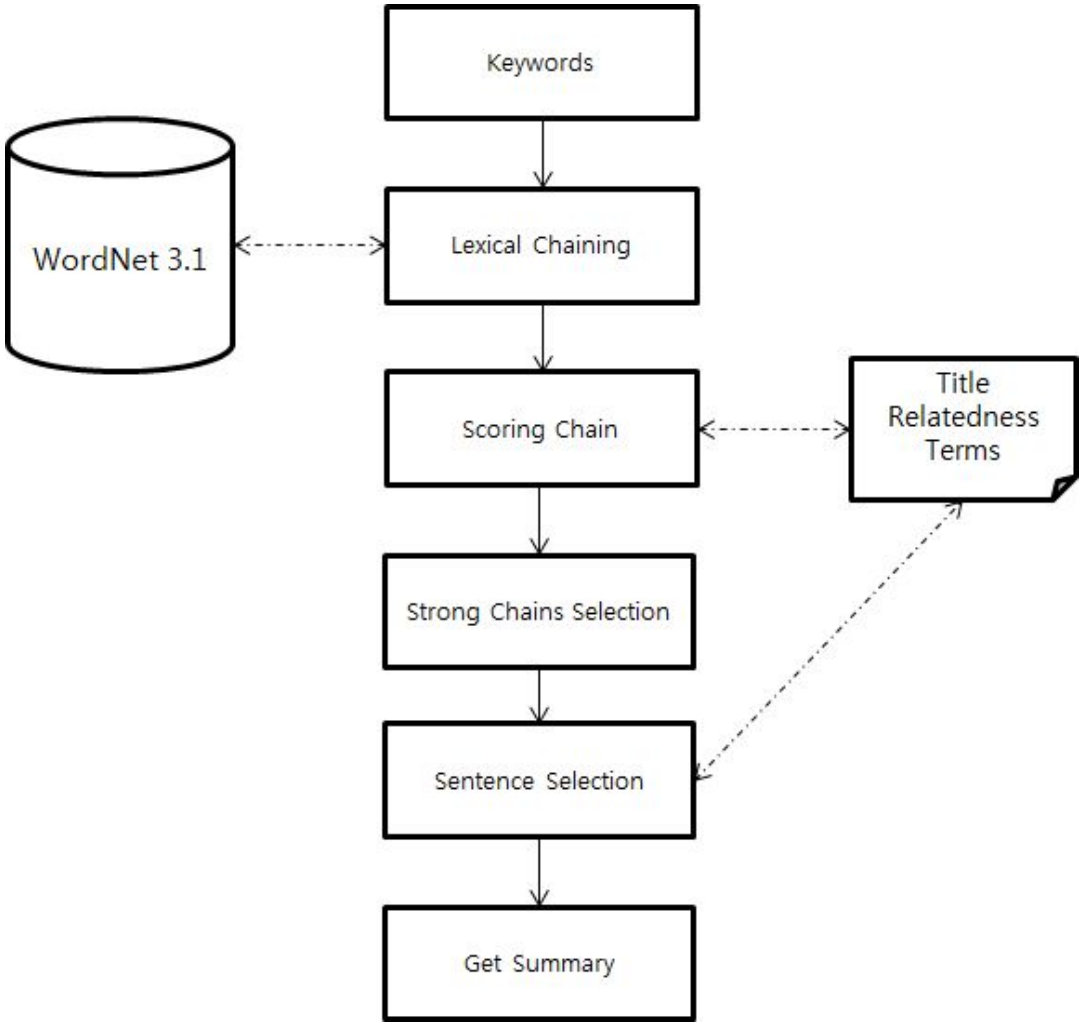

Figure 5.Extracting Summary Sentences from Lexical Chains

A. Data Collection

There are training corpus and test corpus in the data collection. Both data sets are stored in specific way of for different procedure, and collected from various sources where the characteristics and style of writing were different from each other as are from news media like The New York Times, CNN, BBC News and technology news media like Tech Crunch and Mashable and so on. A training document consists of 3 main information; Title, Contents and

manually assigned Keywords according to the range of its n-gram terms within a text while the assigned Keywords information will be excluded for Training documents. In training document, the data is stored in three set of information. The title should be the first line of the text, then the content is stored until the last sentence which are the manually assigned keywords for the given document. The sample format of training document is shown in Figure 6. Likewise, the document for test data or extraction process is stored in two sets of information; it contains Title in the first sentence, and it is followed by its content, but the manual assigned keyword section will be excluded for test data (Figure 7).

```
Uber Invites Developers to Build Apps for Customized Passenger Distractions

Uber, the huge ride-hailing service, delivers millions of rides to passengers.
Now the company wants to give people something to do while they're in the car.
Uber has long been selective about how it works with partner apps and companies.
.
.
.

Uber, app developers, UberRush,new integration
ideas,passengers,smartphone,app,developers,platform,idea,integrations,smartphone app,delivery service,new
integration,test release,handful retailers,the user experience,repeat business affinity,the company brand
```

Figure 6. A sample format of training document

```
iPhone SE: thoughts on going back to a smaller phone

I've been using the new iPhone SE for a couple days now, after having a 6s for a while, and I have to be
honest: going back to using a small phone feels weird.
I'm convinced that Apple has to be aiming this new phone at people who either love small phones or want a
reasonably-priced upgrade from a three-year-old iPhone.
Let's face it, a new iPhone is tempting.
It's a new iPhone! But if you've already graduated to a bigger phone, this phone might not be for you.
.
.
.
```

Figure 7. A sample format of test document

B. Document Preprocessing and Methods

Preprocessing method plays a very important role in text mining techniques and applications. It is the first step in the text mining process or for any further text processing performances. The documents in both training and test corpus are performed some essential fundamental text processing tasks to transform into representation of bag-of-words for n-gram terms of a document ignoring the term order.In this section, we discuss the four key steps of preprocessing namely of our proposed method, sentence segmentation, stop words removal, word tokenization, and Part-of-Speech (POS) Tagging. The NLTK open source tool for Python 2.7 is applied to tackle those preprocessing text for both training and extraction process.

1. Natural Language Toolkit (NLTK)

NLTK is the most leading platform for building Python programs to work with human language data. It includes user friendly interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Thanks to a user friendly guide commencing thr programming fundamentals together with the topics in computational linguistics, and the comprehensive API documentation, NLTK is suitable for linguists, engineers, students, researchers, and industry users and so on. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project. NLTK has been called a wonderful tool for teaching, and working in, computational linguistics using Python, and an amazing library to play with natural language.

NLTK was primarily established in 2001 being part of a computational

linguistics course in the Department of Computer and Information Science at the University of Pennsylvania. After that, it has been utilized and expanded. Now, it has been accepted in courses in many universities, and serves as the basis of many research projects. Table6 illustrates the most important NLTK functionality.

Table 6. Language processing tasks and corresponding NLTK modules with example of functionality

Language Processing Task	NLTK Modules	Functionality
Accessing corpora	nltk.corpus	Standardized interfaces to corpora and lexicons
String processing	nltk.tokenize, nltk.stem	Tokenizers, sentence tokenizers, stemmers
Collocation discovery	nltk.collocations	t-test, chi-squared, point-wise mutual information
Part-of-speech tagging	nltk.tag	n-gram, backoff, Brill, HMM, TnT
Classification	nltk.classify, nltk.cluster	Decision tree, maximum entropy, naive Bayes, EM, k-means
Chunking	nltk.chunk	Regular expression, n-gram, named entity
Parsing	nltk.parse	Chart, feature-based, unification, probabilistic, dependency
Semantic interpretation	nltk.sem, nltk.inference	Lambda calculus, first-order logic, model checking
Evaluation metrics	nltk.metrics	Precision, Recall, Agreement coefficients
Probability and estimation	nltk.probability	Frequency distributions, smoothed probability distributions
Applications	nltk.app, nlotk.chat	Graphical concordancer, parsers, WordNet browser, chatbots
Linguistic fieldwork	nltk.toolbox	Manipulate data in SIL Toolbox format

NLTK was designed with 4 fundamental goals:

Simplicity: Providing an intuitive framework along with substantial building

blocks, giving users a practical knowledge of NLP without getting deserted in the tedious house-keeping usually associated with processing annotated language data.

Consistency: providing a uniform framework with consistent interfaces and the data structures, and easily predictable method names

Extensibility: To offer a structure into which new software modules can be easily integrated including alternative implementations and competing approaches to the same task

Modularity: providing the components which can be used independently without requiring to understand the rest of the toolkit[57].

2. Sentence Segmentation

The text is available only as a stream of characters. Before tokenizing the text into words, it is needed to segment into separated sentences. In English and some other languages, using punctuation, particularly the full stop "." or period character is a reasonable approximation. However, even in English this problem is not efficient due to the use of the full stop character for abbreviations, which may or may not also terminate a sentence. As an example "Dr." is cannot be considered a sentence in "Dr. John Williams went to the hospital in Central Street." When performing on a plain text, tables of abbreviations which include the periods can help obstruct the incorrect assignment of sentence boundaries. Figure 8 shows an example of its use in segmenting the text of a sample document.

```
>>sent_seg = nltk.sent_tokenize('doc1.txt')
>>print sent_seg

['GoPro stock dropped 23.34 percent 10.87 Wednesday...',
'The stock trading time low.', 'The stock briefly jumped 11.20 trading halted.',
'The company market cap 2 billion hour values.',
'GoPro stock dropping the company faced mounting pressure….',
'As the company laying 7 percent 1 500 people 105 people.',
'As today press release states GoPro previously experienced…']
```

Figure 8. A Sample of Sentence Segmentation algorithm

3. Stop words Removal

Stop words are a division of natural language. The motive that stop-words should be removed from a text is that they make the text look heavier and less important for analysts. Removing stop words reduces the dimensionality of term space. The most common words in text documents are articles, prepositions, and pro-nouns, etc. that does not give the meaning of the documents. These words are treated as stop words. Example for stop words; the, in, a, an, with, etc. Stop words are removed from documents because those words are not measured as keywords in text mining applications [37]. There are four types of stop word removal methods; the classic method, methods based on Zipf'sLaw(Z-Methods), the mutual information method (MI), and term based random sampling (TBRS). In our system, we apply the first, classic method which is based on removing stop words obtained from pre-compiled lists [38].

4. Word Tokenization

Word tokenization is the problem of dividing a string of written language into its component words. Not only In English but also many other languages using some form of Latin alphabet, the space is a good approximation of a word divider (word delimiter). Tokenization is a kind of pre-processing in a sense; an identification of basic units to be processed. It is conventional to concentrate on

pure analysis or generation while taking basic units for granted. Yet without these basic units clearly segregated it is impossible to carry out any analysis or generation. After tokenizing the words, Part-of-Speech tagging process can be carried out for each term in a sentence of the text in next step. Figure 9 indicates the sample result of word tokenization in the system carried out by NLTK.

```
>> token_word = [nltk.sent_tokenize(t) for t in text]

[['GoPro', 'stock', 'dropped', '23.34', 'percent', '10.87',
'Wednesday', 'afternoon', 'the', 'company', 'announced', 'Q4',
'worse', 'expected', '.'], ['The', 'stock', 'trading', 'time',
'low', '.'], ['The', 'stock', 'briefly', 'jumped', '11.20',
'trading', 'halted', '.'], ['The', 'company', 'market', 'cap', '2',
'billion', 'hour', 'values', '.'],…]]
```

Figure 9. A Sample Output of Word Tokenization

5. Part-Of-Speech(POS) Tagging

A part-of-speech tagger, or POS tagger, processes a sequence of words, and attaches a part of speech tag to each word. Tagged corpora use many different conventions for tagging words. It also known as grammatical tagging or word-category disambiguation, and it is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its context. As am example, its relationship with adjacent and related words in a phrase, sentence, or paragraph. A simplified form of this is normally taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs, etc[59]. For example,

"They refused to permit us to obtain the refuse permit"

POS Tag: [('They', 'PRP'), ('refused', 'VBP'), ('to', 'TO'), ('permit', 'VB'), ('us', 'PRP'), ('to', 'TO'), ('obtain', 'VB'), ('the', 'DT'), ('refuse', 'NN'), ('permit', 'NN')]

6. Noun Phrase Extraction

As the result of evidences that noun phrases within the text mostly contain the important information or context of the document, and the keywords are primarily formed as a noun phrase according to the surveys, the program is limited to extract noun phrases for n-grams (where 1<= n <= 4) matching the POS patterns. The process of extracting noun phrase can also be known as Noun Phrase Chunking as it involves recognizing the chunks that consist of noun phrases (NPs). One of the most beneficial roots of information for NP-chunking is part-of-speech tags. This is one of the encouragements for performing part-of-speech tagging in our information extraction system. In order to create an Noun Phrase, we will first define a chunk grammar, consisting of rules that indicate how sentences should be chunked. Each string with assigned POS tag from previous stage is applied 35 grammatical rules of noun phrase pattern in regular expression to extract the noun phrases by matching over the exact POS tags. Table 7 shows an example of POS patterns of Noun Phrase for n-grams applied on our system.

Table 7. POS Patterns of Noun Phrase for N-grams

N-gram parameter	POS patterns for Noun Phrase
Unigram/1-gram	“NNS”
	“NN”
	“NNP”
Bigram/2-gram	“NNP” + “NNS”
	“WP$” + “NNS”
	“CD” + “NNS”
	“NN” + “NN”
	“NN” + “NNS”
	“NNP” + “CD”
	“NNP” + “NNP”
	“NNP” + “NNPS”
	“NNP” + “NN”
	“NNP” + “VBZ”
	“DT” + “NNS”
	“DT” + “NN”
	“DT” + “NNP”
	“JJ” + “NN”

	“JJ” + “NNS”
	“PRP$” + “NNS”
	“PRP$” + “NN”
Trigram/3-gram	“NN” + “NN” + “NN”
	“NN” + “NNS” + “NN”
	“NNP” + “NNP” + “NNP”
	“JJ” + “NN” + “NNS”
	“PRP$” + “NN” + “NN”
	“DT” + “JJ” + “NN”
	“DT” + “CD” + “NNS”
	“DT” + “VBG” + “NN”
	“DT” + “NN” + “NN”
	“DT” + “NNP” + “NN”
	“DT” + “NNP” + “NNP”
	“DT” + “JJ” + NN”
	“NNP” + “NNP” + “VBZ”
4-gram	“DT” + “NNP” + “NNP” + “NNP”
	“DT” + “NNP” + “NN” + “NN”

C. Feature Extraction

The proposal is intended to interact with web contents which are quite different from technical writings and format. Web contents commonly do not include Abstract, Introduction, and keyword section like in technical papers. Thus, we rather hold the most possible significant features of keyword to lessen the scope of broad characteristics. Three significant features consists of the weight of a term over the average score of the text using TFIDF method, the occurrence in first sentence of the text and the occurrence in title. First of all, the extracted unigram noun phrase are assigned by their corresponding weight using TFIDF method below.

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \qquad (2)$$

, where tf(t,d) represents the term frequency(tf) of term (t) in document (d), and idf(t, D) is inverse document frequency of term (t) in the corpus (D). Instead of getting the raw frequency of each term in a document, the log normalization form of term frequency is applied to reduce the effect of multiple occurrences of a term.

$$tf(t,d) = 1 + \log(ftd) \qquad (3)$$

, where (ftd) is raw frequency of term (t). Likewise, the inverse document frequency of each term is also applied into log normalization form as below.

$$idf(t,D) = \log(1 + \frac{N}{n_t}) \qquad (4)$$

, where N is the total number of documents in corpus and nt is the number of documents term t occurs. TFIDF scores for n-grams more than unigram is simply the total sum of the score of each member in a term. The TFIDF score of each term is calculated and compared to threshold (average TFIDF score) to denote whether it scores higher than the average score or lower. The average TFIDF score is computed as the sum of total TFIDF scores in document divied by the total number of terms in the given document. If the weight of a term scores higher than the threshold, the term is recorded with its corresponding

weight and its TFIDF feature representation becomes 1 or 0 for terms with lower score.

Another important characteristic of a keyword is the occurrence of a term in the first sentence. Most of the web contents generally introduce the main idea or context of the document in a very first sentence of first paragraph. Finally, the most accurate and effective way to distinguish a term a keyword is its appearance in title of the document. The representations for both features are also be denoted in binary values (0 or 1).

D. Constructing Transition Probability Distribution Generator (TPDG)

The idea of building TPDG Model is to produce the posterior probabilities of state/class (Keyword/Not Keyword) given predictor (features) using Naive Bayesian classifier algorithm based on Bayes' Theorem. Bayes' Theorem provides a way of calculating the posterior probability, P (S|F), from P(S), P(F), and P(F|S), where P(S|F) is posterior probability of State (S) given Features (F), P(S) is the prior probability of State (S), P(F) is the prior probability of Features(F), and P(F|S) is the likelihood which is the probability of Features(F) given State(S). Naive Bayes classifier assumes that the effect of the value of a feature (F) on a given state (S) is independent of the values of other features. This assumption is also called class conditional independence. In TPDG, the generator produces the posterior probabilities of Keyword or Not Keyword state of each given feature for each n-grams term by learning the characteristic of the keywords from training corpus. The Bayes' posterior probability of being Keyword given terms with over average TFIDF score can be written as:

$$P(K|TFIDF) = \frac{P(TFIDF|K)P(K)}{P(TFIDF)} \qquad (5)$$

where P(TFIDF|K) is the likelihood which is the probability of TFIDF feature given Keyword state, and it can be written as;

$$P(TFIDF|K) = \frac{N_{ktfidf}}{N_k} \quad (6)$$

, where is the number of terms with over average TFIDF score being identified as Keyword, and is the number of terms which are Keyword. Since there are certain likelihood probabilities for other possible conditions, we defined the likelihoods in total of four probabilities for TFIDF feature, and they can be described as:

$$P(TFIDF|NK) = \frac{N_{ktfidf}}{N_{nk}} \quad (7)$$

$$P(NTFIDF|K) = \frac{N_{nktfidf}}{N_{nk}} \quad (8)$$

$$P(NTFIDF|NK) = \frac{N_{nkntfidf}}{N_{nk}} \quad (9)$$

, where K represents keyword state, NK is Not Keyword state, is the number of terms with over average TFIDF score which are Not Keywords, and is the number of terms which are Not Keywords. P(TFIDF) is the prior probability of TFIDF score feature, where the sum of both likelihood of keywords and non-keywords given TFIDF feature.

$$P(TFIDF) = P(TFIDF|K) + P(TFIDF|NK) \quad (10)$$

While P(NTFIDF) is the prior probability of non-TFIDF conditions where the sum of both likelihood of keywords and non-keywords give non-TFIDF condition.

$$P(NTFIDF) = P(NTFIDF|K) + P(NTFIDF|NK) \quad (11)$$

Therefore, we have 4 likelihoods, 2 prior probabilities of feature given states for each feature. It means as the proposal emphasizes on 3 distinct features of keyword mentioned in Section C, we will get 12 likelihoods and 6 prior probabilities of feature give states in total. But then, depending on those probabilities the system is able to calculate the 2 prior probabilities of the states P(K) and P(NK). Finally, the system distributes 4 posterior probabilities

for each feature, 12 in total for all 3 features to build 2x2 dimensional matrices as transition distributions to be interacted with Markov Chain process, and those transition distribution matrices for each feature for n-grams (T) is built as can be seen inFigure 10, the entries in the first column in the matrix indicates the probabilities of the Keyword state of both given feature is active or discharge by assigning the corresponding posterior probabilities, and the second column represents the probabilities of the Not Keyword state of given feature is active or discharge.

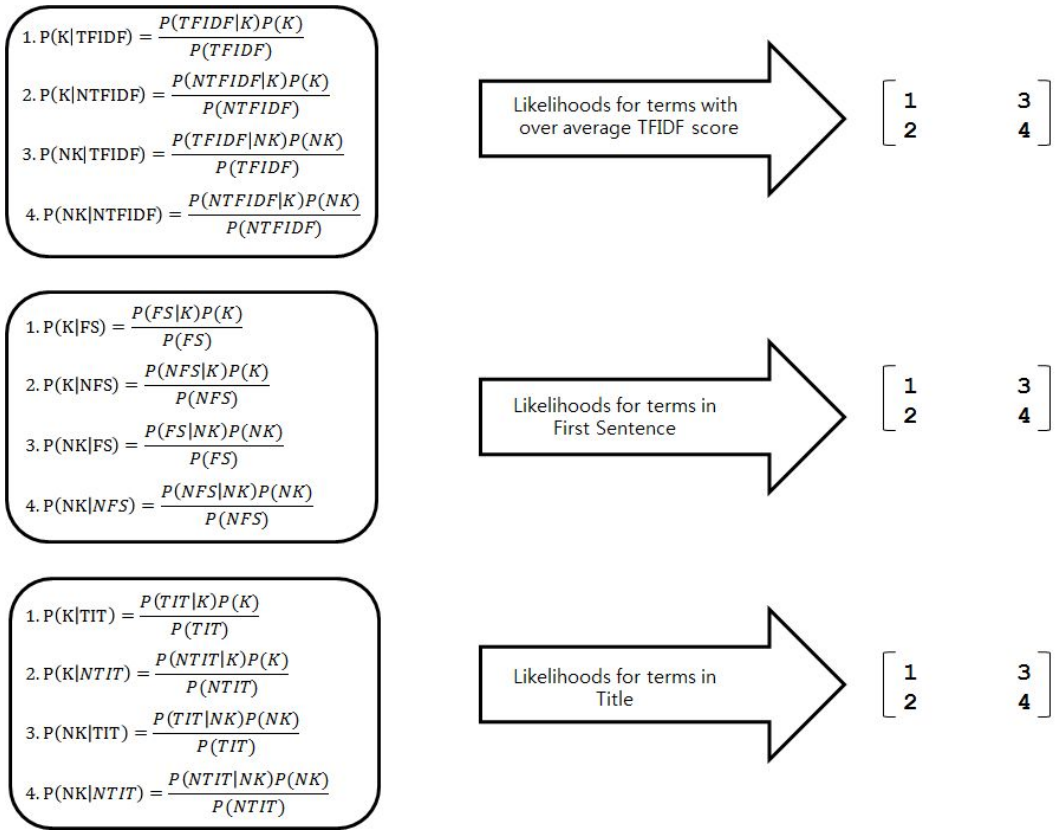

Figure 10. Transition matrices built with Likelihoods for each distinct feature

Finally, in total of 12 transition matrices (T) for n-grams are distributed for the extraction process. However, the transition matrix(T) do not follow the property of conventional Markov Chain model as the sum of the entries of each

row are not add up to 1, yet the actual probabilities of every possible condition of a keyword according to manually indexed training data are contained.

E. Assigning Keywords with Markov Chain using TPDG

Extraction process consists of a few same basis tasks as mentioned in Section 3, the corpus in test data are applied those similar procedures such as Preprocessing in Section B including sentence segmentation, Stopwords removal, word tokenization, POS tagging and Noun Phrase chunking. Then it proceeds to feature extraction in order to transform the document into set of noun phrases with their corresponding properties according to the significant features.

The Markov Chain model [51] is applied for extracting keywords on constructed document representation and it can be described as follows: We have a set of states S = {K, NK}. The process starts in one of these states according to their corresponding TFIDF values, and move successively from one state to another based on the probability values of each stage. There will be 3 stages of process in the system as the proposal method includes three best important features; TFIDF score, Occurrence in first sentence, Occurrence in Title. We assume that the occurrence of a term in a title has the most accurate characteristic of a keyword while the appearance of a term in first sentence quite lower than that, yet it is still better to assign a keyword rather than the other features like the length of a term, appearance in the last sentence, and so on. Hence, we designed three stages in an order.

1. TFIDF score
2. Occurrence of the term in first sentence
3. Occurrence of the term in title

The first entry of initial matrix(I) of a model is the TFIDF score of the term, and the second entry is the average TFIDF score within the document.

$$I = [TFIDF, avgTFIDF] \qquad (12)$$

As a term's initial matrix is compiled, the first stage of the process begins with TFIDF probability distributions which is generated by TPDG in Section D. The process returns the result in a 1x2 dimensional matrix, and again proceeds to second stage with the generated probability distributions for the condition of occurrence of the term in first sentence. Then another new conditional matrix is produced to operate the final stage of the process with the distributions of the occurrence of the term in title. The extraction process, Figure 11, is described as follows:

1. Ctfidf = I * Ttfidf (13)
2. Cfs = Ctfidf * Tfs (14)
3. Ctit = Cfs * Ttit (15)

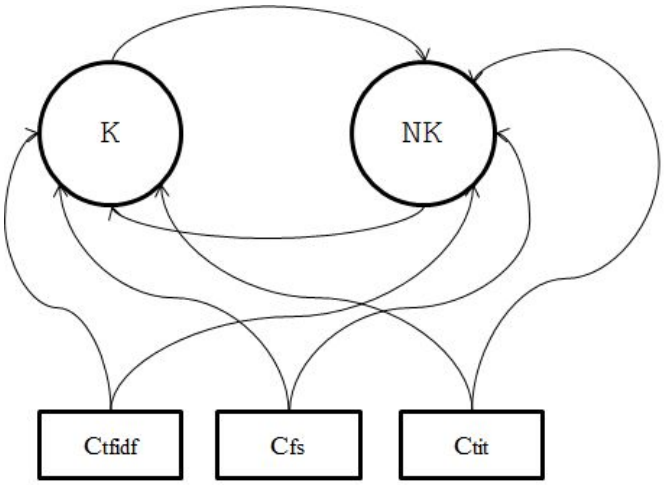

Figure 11.Markov Chain process of Keyword Extraction

, where C is the condition of a term, I is the initial state matrix, and T is the Transition matrix for each distinct feature generated by TPDG in Section D.

Lastly, the probability of a term being in keyword state is calculated by

multiplication of current state distribution matrix with the 3rd feature's probability matrix, and is defined in 1x2 matrix of blocks where the first entry represents the probability of being in Keyword state and the second entry defines the probability of being in Not Keyword state. For this reason, a term is denoted as keyword [K] if the first entry of the final matrix value is higher than the second entry, while a term is labelled as Not Keyword [NK] if the second entry is higher than the first in the labelling process to extract the keywords.

F. Building Lexical Chains

After assigning the candidate keywords, the system acquires the extracted unigram keywords in order to construct the Lexical chains. Our system's building lexical chains method is based on the method of Barzilay and Elhadad. The procedure for constructing lexical chains follows these steps:

1. Select a set of candidate keywords;
2. For each candidate keyword, find an appropriate chain relying on a relatedness criterion among members of the chain;
3. If it is found, insert the word in the chain and update it accordingly.

Moreover, there are 4 distinct relations finding an appropriate chain of each candidate keywords among the members of the chain. They are repetition, hyponym, hypernym and synset. Repetition is the co-occurrence of the given term. Hyponym is a word or phrase whose semantic field is included within that of another word, its hypernym. In simpler terms, a hyponym indicates a type of relationship with its hypernym. For example, Figure 12 indicates pigeon, crow, eagle and seagull are all hyponyms of bird which is their hypernym, and which in turn, is a hyponym of animal. And, the synsetis the set of synonyms of the given term. In the pre-processing step, all words that appear as a noun entry in WordNet are collected. Relatedness of words is determined in terms of the distance between their occurrences and the form of the path connecting

them in the WordNet lexical database. A term with one of each distinct relation with the given candidate keyword is added into the chain.

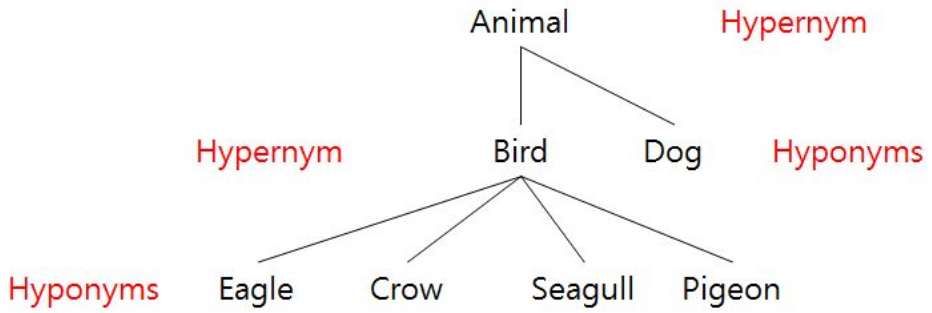

Figure 12. An example of the relationship between Hyponyms and Hypernym

If no chain is found furthermore, then a new chain is created and the candidate word is inserted with all its possible senses in WordNet. But the implemented greedy disambiguation method in this algorithm has some limitations described by the following example:

> **Mr.** *Kenny is the* **person** *that invented an anesthetic* **machine** *which supports the* **micro-computers** *in order to control the rate at which an anesthetic is pumped into the blood. Those* **machines** *are nothing new whilst his* **device** *compromises two* **micro-computers** *to achieve much closer monitoring of the* **pump** *feeding the anesthetic into the patient.*

Let's consider that we have our candidate keywords for the paragraph above as, {"Mr.", "person", "machine", "micro-computers", "machines", "device", "micro-computers", "pump"}. The lexical chain for the word "Mr." is first created ["Mr.", sense {mister, Mr.}]. "Mr." belongs only to one synset, so it is disambiguated from the beginning. The word "person" is related to this chain in the sense "a human being" by a synset relation, so the chain now contains two entries:

[lex "Mr.", sense {mister, Mr., man}]

[lex "person", sense {individual, someone, person, man, mortal, human, soul}].

But for the word "person", the previous relation with "Mr." is added, and the word "machine" which has the lexical relation as "an efficient person" is also inserted into its chain. So the chain for word "person" is generated as:

["person", sense {person, individual, someone, man, mortal, human, soul}].

["Mr.", sense {mister, Mr., man}]

["machine", sense {an efficient person}].

The lexical chain members for the word "machine" are "micro-computers" which is the hyponym of the "machine", and "machines" which is the plural noun of it, and "device" which also is the hyponym of the machine according to the synset relation of WordNet. The sample of lexical chains for word "Mr.", "Person" and "Machine" can be seen in Figure 13 [12].

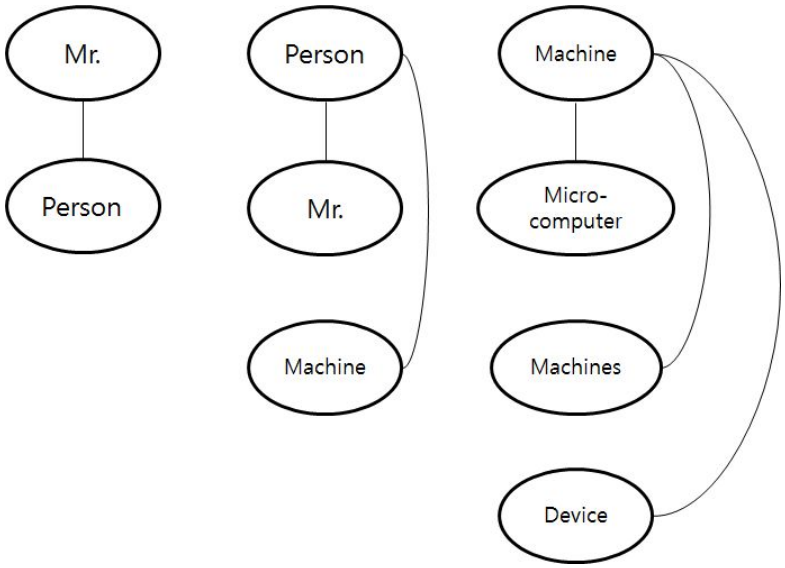

Figure 13.The lexical chains for word "Mr.", "Person" and "Machine"

G. Scoring Chains

Our scoring method for chains differs from the conventional method of Barzilay and Elhadad [12]. Our system emphasizes on the occurrence of the most important keywords of the content such as terms involved in title, and terms which are lexically related to title terms. We distinguish the set of exact terms in title and the set of related terms with title for our candidate unigram keywords, and then we set the value of 1 for each member term of a set of exact title term and 0.5 for members of set of title related terms. The following parameters are good predictors of the strength of a chain:

Length: The number of occurrences of members of the chain

Homogeneity index: 1 - the number of distinct occurrences divided by the length.

$$Score(Chain) = Length * HomogeneityIndex \quad (13)$$

After scoring a chain according to the formula above, the title relatedness terms values must be combined into its score chain. The title relatedness terms which we categorized in two groups; the exact term or co-occurrence of title term in a sentence, we set as 1, and the synonyms or related terms of title terms are set to 0.5 respectively. Then, the number of times of appearance of each member of that two groups, their set value will be added to the chain accordingly.

H. Strong Chain Selection

After scoring each lexical chain of the document, the strong chains are appointed upon their weight values whether their score higher than "Strength Criterion" value. The "Strength Criterion" value can be described as below which is applied in Barzilay and Elhadad's method:

$$Score(Chain) > Average(Scores) + 2 * Standard\,Deviation(Scores) \quad (16)$$

These are preliminary results but they are notably confirmed by their experiments on the 30 texts we analyzed extensively. The method was experimented with different normalization methods for the score function, but they do not seem to improve the results. Extending the empirical analysis in the future, and to support the formal learning methods to determine a good scoring function. According to the experiments of Barzilay and Elhadad, the average number of strong chains selected by this selection method was 5 for texts of 1055 words on average (474 words minimum, 3198 words maximum), when 32 chains were originally generated on average.

I. Sentence Selection

After the strong chains have been selected, the next pace of the summarization algorithm is to extract full sentences from the original document respected to the chain distribution. The sentences with the occurrence of each member of the strong chains are extracted, but then duplication of the exact same sentences will be ignored. Each of the sentences is considered the involvement of the title terms or the related title terms in it. The length of sentence and the number of title terms of related title terms occurrence within a sentence commit the ranking process in descending order. The very first best 7 sentences will be selected as summary sentences of the give text.

Ⅳ. EXPERIMENTAL EVALUATION

A. Evaluation Measures for Keyword Extraction

For our experiment in keyword extraction (TPDG), we compared our proposed keyword extraction method with three recent automatic keyword extraction approaches which are word co-occurrence method called Rapid Automatic Keyword Extraction (RAKE) [46] which used to extract keywords of a single document domain independently, and it basically relies on the term frequency, term degree and ratio of degree to frequency, and the second system is the CRF model [47], and the Support Vector Machine approach [48]. Our experimental data set contains in total of 600 various documents from different sources in terms of the writing style and format.

1. Experimental Results

In the evaluation, we followed the contingence table from C.Zhang's CRF Model [47]. There are two types of keyword collection which are manually extracted by humans, and the collection extracted by the keyword extraction methods. Each extracted keyword collection include two possible states which are keywords and non-keywords state. Table 8 shows the contingence table on the result of keywords extraction and manual assignment keywords.

Table 8. Contingence table of Extraction and Manual Assignment

	Manually assigned Keywords	Manually assigned Non-keywords
Keywords extracted by system	a	b
Non-keywords extracted by system	c	d

For our experiments, we processed the evaluations according to the general measuring method used in the Information Retrieval evaluation, i.e. Precision (P), Recall (R) and F1-Measure. The evaluation measures are defined as follows:

$$P = \frac{a}{a+b} \tag{19}$$

$$R = \frac{a}{a+c} \tag{20}$$

$$F_1(P,R) = \frac{2PR}{P+R} \tag{21}$$

The proposal outperforms well by producing better performance and meaningful keywords. We evaluate the performance of the four keyword extraction methods, i.e. CRF, SVM, RAKE, and our proposed method, by using the formulas for Precision, Recall and F1-measure mentioned above.

Table 9 shows the results of four keyword extraction models. According to F1-Score in Table 9, we can see that our proposed model outperforms the other approaches, and the result of the F1-Score comparison is: proposed model>RAKE>CRF>SVM.

Table 9. Performance Evaluation of Keyword Extraction

Model	Precision	Recall	F1-Score
SVM	0.8017	0.3327	0.4653
CRF	0.6637	0.4196	0.5125
RAKE	0.4929	0.5958	0.5329
Proposed model	0.5216	0.6304	**0.5709**

Figure 14 is a sample of a web document, while Figure 15 shows the sample keyword extraction of our system from the give text.

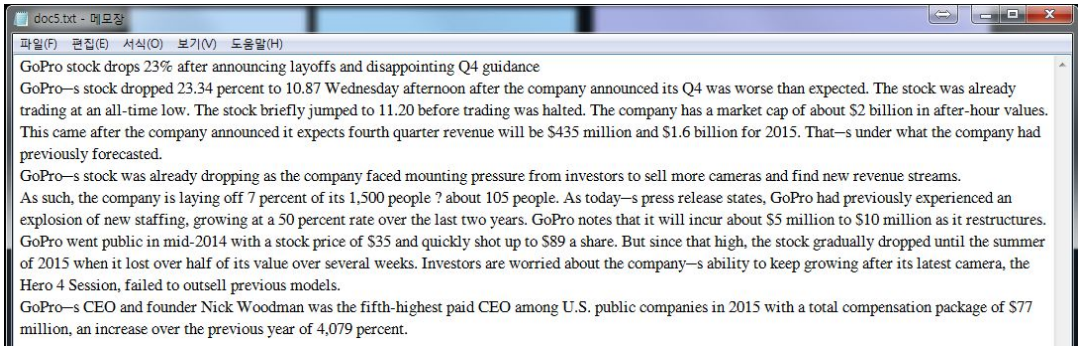

Figure 14. Sample of a Web Document

Figure 15. Candidate Keywords produced by the Proposed System

B. Evaluation Measures for Summarization

In evaluation of our proposed summarization, the evaluation methods can be broadly classified into two categories [39] intrinsic and extrinsic. The most common methods of evaluation, rate the summaries based on their ability to perform certain task (Information retrieval etc). Those methods of evaluation determine the quality of the summaries based on the overlap with human generated "ideal summaries". In the evaluation, precision and recall are the mainly used measures computed based on the number of units (sentences, words, etc) common to both system-generated and ideal summaries. Precision (P) is defined as the percentage of summary generated by system in common with the ideal summary.

$$P = \frac{|SystemSummary| \cap |ReferenceSummary|}{|SystemSummary|} \tag{22}$$

Recall (R) is defined as the ratio of the number of units (sentences or words) of the summaries extracted from systems in common with the ideal summaries to total number of units in the ideal summaries.

$$R = \frac{|SystemSummary| \cap |ReferenceSummary|}{|ReferenceSummary|} \tag{23}$$

Another formula, F-measure, is a composition score that uses (3 factor to weight the relative importance of precision and recall measures:

$$F-measure = \frac{(1+\beta^2)R^*P}{R+\beta^2 P} \tag{24}$$

We evaluated our summarization techniques using the test data collected from different sources such CNN, BBC UK, TechCrunch, New York Time and so on. We performed the automatic evaluation of summaries of our proposed system using ROUGE [40].

1. ROUGE

ROUGE (Recall-Oriented Understudy of Gisting Evaluation) is a system includes a collection of measures to evaluate the summaries by comparing them with "ideal" summaries automatically, without much of human effort. Then, the quality of the summaries is determined by the number of n-gram (sequence of n words) overlaps between the two summaries. ROUGE calculates considered in the evaluation are: ROUGE-N (n=1, 2, 3, 4).

ROUGE-L, a recall based measure, is measured by the number of n-gram overlaps (n = 1, 2, 3, 4) between the reference and system generated summaries. It is computed as follows:

$$= \frac{\sum_{S \in ReferenceSummaries} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in ReferenceSummaries} \sum_{gram_n \in S} Count(gram_n)} \quad (25)$$

, where n stands for the length of the n-gram, $gram_n$ and $Count_{match}(gram_n)$ is the maximum number of n-grams common to both summaries. When the several multiple references are used for evaluation, pairwise summary-level ROUGE-N score between the candidate summary "s" and every reference summary "r" is first computed.

Given M references, the best score over M sets of M-1 references is computed. The final score is the average of the M ROUGE-N scores using different M-1 references. This method is also known as Jacknifing procedure and, it is applied to all ROUGE measures in the ROUGE evaluation package for our evaluation process.

2. Experimental Results

To test the efficiency of our system, we have performed a preliminary set of experiments. We experimented our system with two most recent and well-known summarizers; TextRank [44] (Baseline A) and Automatic Text

Summarizer [49] (Baseline B). We have used a corpus of 600 news contents from several different sources such as CNN, BBC, the New York Times and so on. In order to evaluate the system with ROUGE method, we provided two manually human extracted summaries for each contents Reference summaries, and Baseline A, Baseline B and our proposed system for three System summaries. Figure 16 shows the sample summary of 6 sentences from a document produced by our system. And the original text can be seen in Figure 14.

```
############################
GoPro stock drops 23% after announcing layoffs and disappointing Q4 guidance

########## SUMMARY ##########

GoPro  s stock dropped 23.34 percent to 10.87 Wednesday afternoon after the c
ompany announced its Q4 was worse than expected.

GoPro  s stock was already dropping as the company faced mounting pressure fr
om investors to sell more cameras and find new revenue streams.

GoPro went public in mid 2014 with a stock price of  35 and quickly shot up t
o  89 a share.

The stock was already trading at an all time low.

But since that high  the stock gradually dropped until the summer of 2015 whe
n it lost over half of its value over several weeks.

The stock briefly jumped to 11.20 before trading was halted.
```

Figure 16. Sample Summary of A Document produced by the Proposed System

We have experimented the effect of n-gram words using ROUGE evaluation by changing the parameter(s) of n value. Table 10 shows that our system outperforms than the baseline methods according to the F-score value. And then, the sum of all Recall, Precision and F-score gives the highest score among the compared systems for ROUGE-1 evaluation.

Table 10. Comparison of ROUGE-1 Evaluation of Summarization Methods

System	Recall	Precision	F-score
Baseline A	5.0500	6.0249	5.3705
Baseline B	4.5273	**6.3084**	5.3395
Our system	**5.0504**	5.9864	**5.9864**

The result of ROUGE-2 evaluation of our system returns the lowest score compared to baseline methods in Table 11, while the baseline B performs better than the first baseline, and proposed system according to it's F-score for ROUGE-3 evaluation in Table 12.

Table 11. Comparison of ROUGE-2 Evaluation for Automatic Summarization

System	Recall	Precision	F-score
Baseline A	**4.1113**	4.6723	4.3526
Baseline B	3.7204	**5.5362**	**4.3978**
Our system	4.0498	4.9395	4.1939

Table 12. Comparison of ROUGE-3 Evaluation of Summarization Methods

System	Recall	Precision	F-score
Baseline A	**3.9512**	4.3972	4.1135
Baseline B	3.4723	**5.2058**	**4.1147**
Our system	3.8652	4.7591	4.0110

Table 13 shows our system outperforms the rest of the baseline systems in ROUGE-4 evaluation, we also measure the quality of the summaries generated with respect to various quality in mean coverage, and it surpasses the baseline methods in Table 14.

Table 13. Comparison of ROUGE-4 Evaluation of Summarization Methods

System	Recall	Precision	F-score
Baseline A	3.7234	4.2068	3.9530
Baseline B	3.3315	**5.0298**	3.9564
Our system	**3.7578**	4.6562	**3.9584**

The proposed method selects 7 most representative sentences as summaries.

To give a fair comparison, we examines the average of Recall, Precision and F-score result for each system overall ROUGE 1 to ROUGE 4. Table 14 shows the performance comparison between 2 baseline methods and our proposed system according to their average score in all measures.

Table 14. Performance Comparison between Baseline methods and proposed system

Recall					
	ROUGE1	ROUGE2	ROUGE3	ROUGE4	Avg Scr
Baseline 1	5.0500	**4.1113**	**3.9512**	3.7234	**4.2090**
Baseline 2	4.5374	3.7204	3.4723	3.3315	3.7629
Our System	**5.0504**	4.0498	3.8652	**3.7578**	4.1808
Precision					
	ROUGE1	ROUGE2	ROUGE3	ROUGE4	Avg Scr
Baseline 1	6.0249	4.6723	4.3972	4.2068	4.8253
Baseline 2	**6.3084**	**5.5362**	5.2058	5.0298	5.5200
Our System	6.5864	4.93958	5.35916	5.6562	**5.6353**
F-score					
	ROUGE1	ROUGE2	ROUGE3	ROUGE4	Avg Scr
Baseline1	5.3705	4.3526	4.1135	3.9530	4.4474
Baseline2	5.3395	**4.3978**	**4.1147**	3.9564	4.4521
Our System	**5.9864**	4.1939	4.0110	**3.9584**	**4.5374**

In Recall measure, the average score of Baseline 1 outperforms Baseline 2 and proposed method, while the average score for Precision and F-score of our proposed system surpasses the other approaches.

We compared the performance and efficiency of each system according to the size training set corpus. Using the average score of each system produced by the ROUGE evaluation method, we can see that the performance of the proposed system increased depending on the size of the training corpus. It gets better and, produced more accurate improved results as the size of training set is increased [Figure 17].

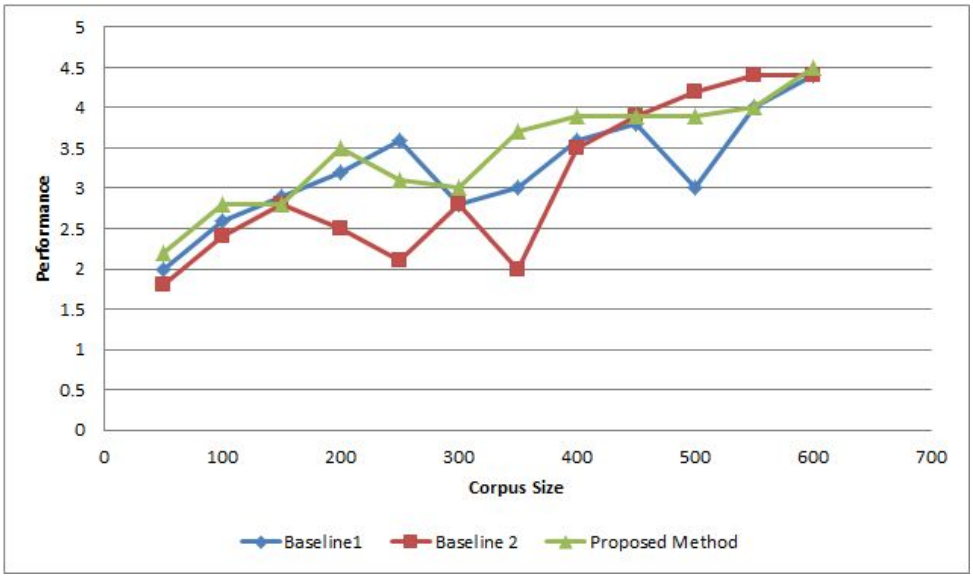

Figure 17. Performance Comparison of each system according to the size of training corpus

V. CONCLUSION AND FUTURE WORK

In this thesis, we presented a method to compute the probability distributions of keywords of the text according to their characteristics, and a method to construct lexical chains as an efficient intermediate representation of the document based on extracted candidate keywords. Along with normal WordNet relations, our method also included additional relations such as proper noun repetition and gloss relations in the computation of lexical chains.

Then, we investigated the approaches to extract sentences from the document(s) based on the distribution of lexical chains. We built a transition probability distribution generator (TPDG) for n-gram keywords which learns the characteristics of the assigned keywords from the training data set. A new method of automatic keyword extraction also featured in the system based on the Markov chains process. Among the extracted n-gram keywords, only unigrams are selected to construct the lexical chain. Instead of finding all the noun phrases from the text to build a lexical chain, collecting the most related keywords of the text gives the better performance and more efficient result. Terms involved in title, and related to title are vital for scoring and identifying the competent information within the text.

Lexical chains, until now, were mainly used to generate single document summaries. Lexical chains support to identify the themes, by clustering the document collection. We performed intrinsic evaluation to distinguish the quality of the summaries generated by our approaches. We found that our system achieved better results in keyword extraction than existing RAKE, and returns more efficient summaries than baseline method and recent systems.

Our method to compute lexical chains includes the gloss relations using the advent of the lexical database. These relations were based on the presence

of gloss concept or synonym of the gloss concept in the text. We would like to pursue further research into the methods to compute the semantic similarity based on the overlap of the concepts of gloss. The lexical chains are evaluated according to their performance in identifying the sense of the word in given context. It has been proved that concepts present in the gloss of a word play an important role in the determination of the word sense [41] [42]. We would like to compare the performance of our system in this aspect with respect to other lexical chaining methods, and our future study will be emphasized towards automatic abstractive summarization while improving the extraction of keywords of the text corpus independently.

REFERENCES

[1] Radev, D. R., Hovy, E., McKeown, K., "*Introduction to the special issue on summarization*". Computational Linguistics, vol. 28, no. 4, pages 399-408, 2002.

[2] Luhn, H.P., "*The automatic creation of literature abstracts*". IBM Journal of Research Development, vol. 2, no. 2, pages 159-165, 1958.

[3] Baxendale, P., "*Machine-made index for technical literature - an experiment*". IBM Journal of Research Development, vol. 2, no. 4, pages 354-361, 1958.

[4] Edmundson, H. P., "*New methods in automatic extracting*". Journal of the ACM, vol. 16, no. 2, pages 264-285, 1969.

[5] Kupiec, J., Pedersen, J., and Chen, F., "*A trainable document summarizer*". In Proceedings SIGIR '95, pages 68-73, New York, NY, USA, 1995.

[6] Aone, C., Okurowski, M. E., Gorlinsky, J., and Larsen, B., "*A trainable summarizer with knowledge acquired from robust nlp techniques*". In Mani, I.and Maybury, M. T., editors, Advances in Automatic Text Summarization, MIT Press, pages 71-80. 1999.

[7] Miller, G. A., "*Wordnet: a lexical database for english*". Commun. ACM, vol. 38, no. 11, pages 39-41, 1995.

[8] Conroy, J. M. and O'leary, D. P., "*Text summarization via hidden markov models*". In Proceedings of SIGIR '01, 406-407, New York, NY, USA, 2001.

[9] Nenkova, A., "*Automatic text summarization of newswire*", Lessons learned from the document understanding conference. In Proceedings of AAAI 2005, Pittsburgh, USA, 2005.

[10] Svore, K., Vanderwende, L., and Burges, C, "*Enhancing single-document summarization by combining RankNet and third-party sources*". In Proceedings of the EMNLP-CoNLL, pages 48-457, 2007.

[11] Luhn, H., "*The automatic creation of literature abstracts*". IBM Journal of Research and Development, vol. 2, no. 2, 1958.

[12] Barzilay, R. and M. Elhadad, "*Using lexical chains for text summarization*". In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th European Chapter Meeting of the Association for Computational Linguistics, Workshop on Intelligent Scalable Text Summarization, pages 10-17, Madrid, 1997.

[13] Halliday, M. and R. Hasan, "*Cohesion in English*". Longman, London, 1976.

[14] Mani, I., "*Automatic Summarization*". John Benjamins Co, Amsterdam, Philadelphia, 2001.

[15] Lesk, M.. "*Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone*". In Proceedings of the5th annual international conference on Systems documentation, pages 24-26,Toronto, Ontario, Canada. ACM Press, 1986.

[16] Banerjee, S. and T. Pedersen, "*Extended gloss overlaps as a measure of semantic relatedness*". In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pages 805-810, Acapulco, Mexico, 2003.

[17] Harabagiu, S. and D. Moldovan, "*WordNet: An Electronic Lexical Database, chapter Knowledge Processing on an Extended WordNet*". MIT press, 1998.

[18] A. Hulth. "*Improved automatic keyword extraction given more linguistic knowledge*". In Proceedings of the 2003 Conference on Emprical Methods in Natural Language Processing, Sapporo, Japan, 2003.

[19] Michael J. Giarlo, "*A comparative analysis of keyword extraction techniques*". Rutgers, The State University of New Jersey.

[20] Cohen J. D.,. "*Highlights: Language and Domain-independent Automatic Indexing Terms for Abstracting*". Journal of the American Society for Information Science, 1995, vol. 46, no. 3, pages 162-174, January 1999.

[21] Luhn H. P., "*A Statistical Approach to Mechanized Encoding and Searching of Literary Information*". IBM Journal of Research and Development, vol. 1, no. 4, pages 309-317, 1957.

[22] Salton G, Yang C. S., Yu C. T., "*A Theory of Term Importance in Automatic Text Analysis*". Journal of the American society for Information

Science, vol. 26, no. 1, pages 33-44, 1975.

[23] Matsuo Y., Ishizuka M., "*Keyword Extraction from a Single Document Using Word Co-occurrence Statistical Information*". International Journal on Artificial Intelligence Tools, vol. 13, no. 1, pages 157-169, 2004.

[24] Chien L. F., "*PAT-tree-based Keyword Extraction for Chinese Information Retrieval*". In Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1997), Philadelphia, PA, USA, pages 50-59, 1997.

[25] Ercan G, Cicekli I., "*Using Lexical Chains for Keyword Extraction. Information Processing and Management*". vol. 43, no. 6, pages 1705-1714, 2007.

[26] Hulth A., "*Improved Automatic Keyword Extraction Given More Linguistic Knowledge*". In Proceedings of the 2003 Conference on Emprical Methods in Natural Language Processing, Sapporo, Japan, pages 216-223, 2003.

[27] Dennis S. F., "*The Design and Testing of a Fully Automatic Indexing-searching System for Documents Consisting of Expository Text*". In G. Schecter eds. Information Retrieval: a Critical Review, Washington D. C.: Thompson Book Company, pages 67-94. 1967.

[28] Salton G, Buckley C.. "*Automatic Text Structuring and Retrieval - Experiments in Automatic Encyclopaedia Searching*". In Proceedings of the Fourteenth SIGIR Conference, New York: ACM, pages 21-30, 1991.

[29] Frank E., Paynter G. W., Witten I. H., "*Domain-Specific Keyphrase Extraction*". In Proceedings of the 16th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, Morgan Kaufmann, pages 668-673, 1999.

[30] Zhang K., Xu H., Tang J., Li J. Z., "*Keyword Extraction Using Support Vector Machine*". In Proceedings of the Seventh International Conference on Web-Age Information Management (WAIM2006), Hong Kong, China, pages 85-96, 2006.

[31] Keith H. J. B., "*Phraserate: An Html Keyphrase Extractor*". Technical Report, University of California, Riverside, vol. 1, no. 16, 2002.

[32] Plas L., Pallotta V., Rajman M., Ghorbel H., "*Automatic keyword extraction*

from spoken text, A comparison of two lexical resources: the EDR and WordNet". Proceedings of the 4th International Language Resources and Evaluation, European Language Resource Association, 2004.

[33] Hulth A., "*Improved automatic keyword extraction given more linguistic knowledge".* In Proceedings of the 2003 Conference on Emprical Methods in Natural Language Processing, Sapporo, Japan, 2003.

[34] Witten I., Paynte G., Frank E., Gutwin C., Nevill-Manning C., "*KEA: practical automatic key phrase extraction".* In Proceedings of the 4th ACM Conference on Digital Library, 1999.

[35] Suzuki Y., Fukumoto F., Sekiguchi Y., "*Keyword extraction of radio news using term weighting with an encyclopedia and newspaper articles",* SIGIR, 1998.

[36] Keith J. B., Humphreys. "*Phrase rate: An HTML keyphrase extractor".* Technical Report. 2002.

[37] Porter M. F., "*An Algorithm for Suffix Stripping".* Program, vol. 14, no. 3, pages 130-137, 1980.

[38] Ms. Anjali G. J., "*A Comparative Study of Stemming Algorithms".* Anjali Ganesh Jivani et al, Int. J. Comp. Tech. Appl., vol. 2, no. 6, ISSN:2229-6093, 1930-1938.

[39] Mani I., Maybury M., "*Advances in Automatic Text Summarization".* MIT Press, 1999.

[40] Lin C.-Y., "*Rouge: A package for automatic evaluation of summaries".* In Marie-Francine Moens, S. S., editor, Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, pages 74-81, Barcelona, Spain, 2004.

[41] Lesk M., "*Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone".* In Proceedings of the5th annual international conference on Systems documentation, pages 24-26, Toronto, Ontario, Canada. ACM Press, 1986.

[42] Banerjee S., Pedersen T., "*Extended gloss overlaps as a measure of semantic relatedness".* In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pages 805-810, Acapulco, Mexico, 2003.

[43] Feifan L., Deana P., Fei L., Yang L., "*Unsupervised Approaches for Automatic Keyword Extraction Using Meeting Transcripts*". Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL, pages 620 - 628, Boulder, Colorado, June 2009.

[44] Rada M., Paul T., "*TextRank: Bringing Order into Texts*". In Proceedings of the Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, Barcelona, Spain, July 2004.

[45] Martin D., Karel J., "*Automatic Keyphrase Extraction based on NLP and Statistical Methods*". In Proceedings of the Dateso 2011: Annual International Workshop on Databases, Texts, Specifications and Objects, Pisek, Czech Republic, April 20, 2011.

[46] Rose, S., Engel, D., Cramer, N., & Cowley, W.. "*Automatic Keyword Extraction from Individual Documents*". In M. W. Berry & J. Kogan (Eds.), Text Mining: Theory and Applications: John Wiley & Sons, 2010.

[47] Chengzhi Z., Huilin W., Yao L., Dan W., Yi L., Bo W., "*Automatic Keyword Extraction from Documents Using Conditional Random Fields*". Journal of Computational Information Systems, vol. 4, no. 3, pages 1169-1180, 2004.

[48] Zhang K., Xu H., Tang J., Li J. Z.. "*Keyword Extraction Using Support Vector Machine*". In Proceedings of the Seventh International Conference on Web-Age Information Management (WAIM2006), pages 85-96, Hong Kong, China, pages 85-96, 2006.

[49] Annapurna P. Patil, S. D., Syed A. A. A., Tanay A., Varun B., "*Automatic text summarizer*". In proceedings of 2014 International Conference on Advances in Computing, Communications and Informatics(ICACCI), pages 1530-1534, 2014.

[50] Dipanjan D., Andre F. T. M., "*A Survey on Automatic Text Summarization*". Technical Report, 2007.

[51] Charles M. G., Laurie S. J., "*Introduction to Probability*". pages 40-470, 1997.

[52] Fang J., Guogh L,. Xue Y., Xiang Y., "*Semantic-based Keyword Extraction Method for Document*". In proceedings of International Journal of u- and

e-Service, Sclence and Technology, vol. 8, no. 5, pages 37-46, 2015.

[53] Luis M., Wang L., Isabel T., Chris D., Alan W. B., Anatole G., David M. M., Joao P. N., Jaime C., "*Automatic Keyword Extraction on Twitter*". In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers), Beijing, China, pages 637-643, 2015.

[54] Xiong S., Luo Y., "*A new approach for multi-document summarization based on latent semantic analysis*". In Proceedings of the Seventh International Symposium on Computational Intelligence and Design, ISCID 14, vol. 1, pages 177-180, Hangzhou, China, December, 2014.

[55] Yang G., Wen D., Kinsuhuk, Chen N. S., Sutinen E., "*A novel contextual topic model for multi-document summarization*". Expert Systems with Applications, vol. 42, no. 3, pages 1340-1352, 2015.

[56] Meena Y. K., Gopalani D., "*Domain independent framework for automatic text summarization*". Procedia Computer Science, vol. 48, pages 722-727, 2015.

[57] Steven B., Ewan K., Edward L., "Analyzing Text with the Natural Language Toolkit: Natural Language Processing with Python". 2009.

[58] https://en.wikipedia.org/wiki/Tf-idf.

[59] https://en.wikipedia.org/wiki/Part-of-speech_tagging.

ACKNOWLEDGMENTS

I, paying homage to the supreme triple Gems, the Buddha, Dhamma and Sangha, and give great respects to my parents and my teachers. Creating a Master Thesis is not an individual experience; rather it takes place in a social context and includes several persons, whom I would like to thank sincerely.

First and foremost, this thesis is dedicate--d to my parents and, I would like to thank for their endless love, support and encouragement. Thank you both for giving me strength to reach for the stars and chase my dreams. My little brother, cousins, friends, relatives and beloved ones deserve my wholehearted thanks as well. There are many people without whose support and contributions, this thesis would not be possible.

I would like to sincerely indicate my gratitude on my supervisor, Prof. Kim Pankoo, for his guidance and support throughout this study, and especially for his confidence in me. I would also like to thank my seniors, Choi Junho, Choi Chang, Choi Dongjin, Ko ByeongKyu, Kim JeongIn, Lee Eunji and my lab mates Lee Jaeuk, Cha Junseok, Han SungMin, Lee Mungyu and Hong Taekeun, who offered me not only the environment of working in an energetic laboratory but the joyful student life outside of the campus as well.

Last but not least, I would like to thank my friends and colleagues at Chosun University for their encouragement and moral support which made my stay in studies in Gwangju more enjoyable. Everybody kept me going, and this thesis would not have been possible without them.

An Improved Lexical Chain Method for Automatic Text Summarization using Semantic Related Terms

단어 의미적 연관성을 고려한 개선된 어휘체인기반의
자동 문서요약 방법

2016년 8월 25일

조선대학교 대학원

컴퓨터공학과

Htet Myet Lynn

An Improved Lexical Chain Method for Automatic Text Summarization using Semantic Related Terms

지도교수 김 판 구

이 논문을 공학석사학위신청 논문으로 제출함.

2016년 8월

조선대학교 대학원

컴퓨터공학과

Htet Myet Lynn

Htet Myet Lynn의 석사학위논문을 인준함

위원장 조선대학교 교수 정일용 (인)

위 원 조선대학교 교수 정현숙 (인)

위 원 조선대학교 교수 김판구 (인)

2016년 8월

조선대학교 대학원

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

An Improved Lexical Chain Method for Automatic Text Summarization using Semantic Related Terms

Htet Myet Lynn
Advisor : Prof. Pankoo Kim, Ph.D
Department of Computer Engineering
Graduate School of Chosun University

Summarization is a challenging task that need to understand the content of the document to determine the importance information of the text. Automatic Summarization is the procedure of lessening a text document using an intelligent system with complex algorithms to form a short summary of it which retains the most important information of the text. Lexical cohesion is a way identifying connected portions of the text according to the relations between the words in the document or text. Lexical cohesive relations between words in a document can be described using lexical chains. Lexical chains are applied in various Natural Language Processing (NLP) and Information Retrieval (IR) applications. The fundamental task to perform in constructing lexical chain is to extract the best appropriate candidate keywords of the text as the chains which contain the salient portions of the document are relied on them.

Thus, we extend our research on automatic keyword extraction for extracting candidate terms of the given text before approaching to summarization. Keywords are a set of keywords or keyphrases that capture the primary information or topic discussed in the text. Keywords are widely used to define queries in IR systems as they are easy to define, revise, remember, and share. Furthermore, keywords are essential Search Engine Optimization (SEO) elements

for every search engines, yet they are matched against with users' search query keywords. They can empower document browsing by providing a short summary, improve information retrieval, and be employed in generating indexes for a large text corpus.

In current thesis, we proposed a new approach for automatic text summarization using lexical chain with semantic relatedness keywords. In contrast, the new method of extracting keywords is also implemented for constructing the efficient lexical chain. Instead of constructing a lexical chain with every noun phrases in the text, the result of our experimental results show that the extracted candidate keywords of the text delivers a better efficient summary with a better performance. Thus, we have built a system to extract the promising keywords from the text which is based on the characteristics of the manually assigned keywords. The system consists of a generator to produce the possibility distributions of three distinct features of assigned keywords; the occurrence of a term in title, the occurrence of a term in a first sentence, and the higher score of average TF.IDF score of a term. Then, those distributions are applied to Markov chain process of three stages to assign the label for each N-gram term in the text. Extracted unigram candidate keywords are selected to build the lexical chains of the text. There are also three distinct relation criteria to connect the terms according to their relationships to other candidate keywords using WordNet; hypernym, hyponym and synonym. Then we apply the method to score and extract the salient portions of the document. Our experimental results prove that the efficient summaries can be extracted for the above tasks.

요약

단어 의미적 연관성을 고려한 개선된 어휘체인기반의 자동 문서요약 방법

Htet Myet Lynn
Advisor : Prof. Pankoo Kim, Ph.D
Department of Computer Engineering
Graduate School of Chosun University

문서를 요약한다는 것은 그 문서의 일관성을 유지하면서 중복을 제거하고, 응축된 정보를 생산하는 것을 말하며, 자동문서 요약 기술은 컴퓨터를 사용해서 문서 내 중요한 부분을 유지하고, 중복된 내용을 제거함으로써 처리하고자 하는 대용량의 문서를 자동적이고 효율적으로 처리하는 방법을 말한다.

“어휘 결합(Lexical Cohesion)”은 어휘의 관계(상하 어휘관계, 유의 어휘관계 등)를 바탕으로 하나의 문서 내의 등장하는 단어와 단어 사이의 관계를 분석하는 방법이다. 이러한 “어휘 결합 관계”는 어휘 사슬(Lexical Chain)을 이용하여 나타낼 수 있다. 어휘 사슬은 자연언어처리(Natural Language Processing) 및 정보검색(Information Retrieval)기술에 다양하게 활용되고 있으며, 적절한 후보키워드를 추출하는 것이 프로그램의 성능을 좌우하기 때문에, 어휘 사슬을 구성하기 위해 적절한 후보키워드를 추출하는 것이 가장 중요한 작업이라고 할 수 있다.

본 연구는 단어의 의미적 연관성을 고려하여 구성한 후보키워드의 어휘사슬을 기반으로 개선된 자동문서 요약방법에 관한 연구로, 효율적인 어휘 사슬을 구성하기 위해 새로운 키워드추출방법을 제안하였다. 본 논문에서 문서 내의 키워드 추출을 위해 “제목에 등장하는 단어”, “문서의 첫 번째 문장에 등장하는 단어”, “TF-IDF 가중치가 높게 측정되는 단어” 세 가지의 키워드 특징을 정의하였으며, 키워드가 갖는 조건부 확률 값을 활용해 전이 행렬(transition matrices)을 생성함

으로써, 마르코프 연쇄(Markov Chain)에 적용을 통해 후보키워드를 추출한다.

추출된 후보키워드는 워드넷(WordNet) 상에서 정의된 단어의 상하위어 관계, 동의어 관계를 고려하여 후보 키워드 간의 연결을 통해 어휘 사슬을 구성하였으며 이를 통해 자동문서 요약을 수행하게 된다. 본 논문의 실험결과에 따르면, 제안한 방법에 의해 추출한 후보키워드로 어휘 사슬을 구성하는 것이 문서 내 모든 명사구에 대한 어휘 사슬을 구성하는 것보다 향상된 성능을 보였으며, 더욱 효율적으로 문서를 요약할 수 있음을 증명하였다.

Ⅰ. INTRODUCTION

The first chapter states the clarification of the motivation of this thesis, and it will illustrate the actuality and aim of the thesis. Finally, Section B gives a brief overview of the following chapters.

A. Motivation

Today's world is all about online information on the internet across the globe. The World Wide Web comprises billions of documents, and it is growing at an incredible pace. Applications that provide timely access to, and digest of, various sources are required in order to lessen the information overload users are encountering. These obstacles have sparked interest in the development of automatic summarization systems. Such systems are designed to take a single article or multiple articles, a cluster of news articles, or an email thread as input, and execute a concise and efficient summary of the most important information. Recent years have been the development of numerous summarization applications and tools for news, email threads, and professional medical information, scientific articles, spontaneous dialogues, and videos. These systems have already been shown to assist people, and to enhance other automatic applications and boundaries.

B. Outline

The outline of th thesis is organized as follows:

Chapter II analyses the problem by stating the existing approaches in single document text summarization and automatic keyword extraction. This provides an overview of the related works, and explains the choice of our conceptual design.

Chapter III defines the conceptual design and exploited heuristic. The detailed description of the used statistical measures for our new automatic keyword and construction of Transition Probability Distribution Generator is provided in this section. The following section introduces the development of conventional lexical chain and scoring chain techniques are explained, and the detail information of extracting summary sentences is included.

Chapter IV provides the evaluation results for our proposed keyword extraction method, and the evaluation method called ROUGE is described as we apply it for evaluating our system.

Chapter V gives a brief summary of our proposal and presents our future research directions.

II. BACKGROUND CONCEPTS

A. SUMMARIZATION

The area of summarization has been explored by the NLP society for nearly the last half century. Radev et al. [1] defines that a summary as a short text that is executed from a single or multiple texts, that highlights the important information of the original text(s), and that should not be longer than half of the original text(s), and normally significantly less than the original ones. This straightforward definition catches three important aspects that described research on automatic summarization:

- Summaries may be generated from a single or multiple documents,
- Summaries should maintain notable information,
- Summaries must be short.

Even if we agree on these facts, it seems from the literature that any approach to provide a more elaborate description for the task would end in conflict within the community. In fact, many procedures differ on the manner of their own problem fabrications. We can start by indicating some common terms in the summarization dialect: extraction is the procedure of determining important sections of the text and generating them precisely; abstraction aims to produce important information of the text in a new way; fusion combines extracted parts cohesively, and compression intends to exclude the unimportant parts of the text [1]. Earliest occurrences of research on summarizing scientific documents proposed standards for extracting prominent sentences from text using features like word and phrase frequency [2], position in the text [3] and key phrases[4]. Various attempts published since then has focused on other domains, mostly on news data. Many approaches discoursed the problem by constructing systems depending on the type of the required summary. While extractive summarization is mostly concerned with the summary content

extracted from the original text, and it is normally relying on extraction of sentences, abstractive summarization emphasizes on the form, scheming to produce a new summary with grammatical arrangements which usually demands the advanced language generation techniques. A critical issue that will definitely drive future research on summarization is evaluation. During the last 25 years, many evaluation system competitions like TREC, DUC and MUC have produced sets of training data and have initiated baselines for performance levels. However, ROUGE has become a common scheme to evaluate summarization systems. [50]

The rest of this topic is arranged as follows: the following section describes document summarization, focusing on extractive techniques, where a few extractive approaches that pioneered the field are also considered. Commonly, the flow of information in a given document is not uniform, which means that some parts of the text are more important than others. The major challenge in summarization relies on distinguishing the more informative parts of a document from the less ones. But there have been circumstances of research describing the automatic creation of abstracts, most work presented in the literature depends on verbatim extraction of sentences to distinguish the problem of single-document summarization. In this section, we describe some well known extractive techniques. First, we look at early work from the 1950s and 60s that kicked off research on summarization. Second, we look on approaches including machine learning techniques published in the late 2000s to today [54][55][56]. Finally, we briefly describe some techniques that use a more complex natural language analysis to tackle the problem.

With the advent of machine learning approaches, a series of seminal publications appeared that employed statistical approaches to produce document extracts. While the most systems assumed feature independence and relied on naive-Bayes methods, others have emphasized on the appropriate features, and on learning algorithms which has no independence assumptions. Other crucial approaches involved hidden Markov models and log-linear models to enhance the extractive summarization. In contrast, a very recent paper used neural

networks and third party features (like common words in search engine queries), word co-occurrence, and conditional random fields to produce better purely extractive single document summarization. We next describe all these approaches in more detail.

1. Existing Approaches

(i) Naive Bayes Methods

Kupiec et al. [5] describe a technique based on Edmundson [4] which is able to learn from training data. The classification function classifies each sentence as deserving for extraction or not, using a Naive-Bayes classifier. For example, tet s be a particular sentence, S the set of sentences extracted as summary, and the features. Assuming independence of the features:

$$P(s \in S | F_1, F_2, .. F_k) = \frac{\prod_{i=1}^{k} P(F_i | s \in S) . P(s \in S)}{\prod^{k_{i=1}} P(F_i)} \qquad (1)$$

The distinct features were similarly related to Edmundson's method [4], but additionally included the length of sentence and the presence of uppercase words. Each sentence score was scored according to (1), and only the top n sentences were extracted. In evaluating the system, a corpus of technical documents with manual abstracts was used in the following way. For each sentence in the abstract, the researchers analyzed its match with the actual document sentences and created a mapping manually. The system summaries were then evaluated against this mapping. Feature analysis of the system showed that a system using only the position and the features, along with the sentence length as sentence feature to be performed best.

Aone et al. [6] also a naive-Bayes classifier, but with more features. They introduced a system called DimSum that relies on features like term frequency (tf) and inverse document frequency (idf) to derive significant words. The idf was computed from a large corpus of the same domain as the given documents. Statistically acquired two-noun word collocations (bigrams) were applied as units for counting, together with single words. A named-entity tagger was also

applied and each entity was considered as a single token. They also employed some thin discourse analysis like reference to same entities in the text but still maintain the cohesion. Synonyms and morphological variants were merged when considering lexical terms, the former approach being identified by using WordNet [7]. The corpora used in the experiments were from newswire, some of which belonged to the TREC evaluations.

(ii) Hidden Markov Models

With the previous approaches, that were mostly feature-based and non-sequential techniques, Conroy and O'leary[8] introduced the problem of extracting a sentence from a document using a hidden Markov model (HMM). The fundamental motivation for using a sequential model is to discover for local dependencies between sentences. However, only three features were considered, the position of the sentence within the document, number of terms in the sentence, and likeliness of the sentence terms given the document terms.

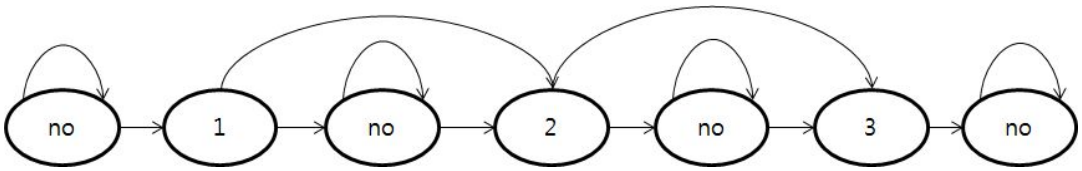

Figure 1.Markov model to extract to three summary sentences from a document(ConroyandO'leary,2001)

The HMM model was constructed as follows: it included 2s + 1 states, changeable between s (summary states) and s+1 (non-summary state). The authors allowed "hesitation" only in non-summary state and "skipping next state" only in summary states. Figure 1 describes an example HMM with 7 nodes, corresponding to s = 3. Using the TREC dataset for training corpus, the authors acquired the maximum-likelihood estimate for each transition

probability, forming the transition matrix estimate M, whose element (i, j) is the empirical probability of transition from state i to j. Combined with each state i was an output function, bi(O) = Pr(O j state i) where O is an observed vector of features. A simplifying assumption was made that the features are multi variate normal. The output function of each state was estimated by using the training data to compute the maximum likelihood estimate of its mean and covariance matrix. They estimated means of 2s+1, but presumed that all of the output functions shared a common covariance matrix. The evaluation was done by comparing with human generated extracts.

(iii) Neural Networks and Third Party Features

In 2001 and 2002, DUC has granted a process of creating a summary with 100 words of a single news article. The systems, however, with the best performance in the evaluations could not surpass the baseline with numerical significance. This highly rated baseline has been analyzed by Nenkova[9] and complies to the selection of the first n sentences of a news article. After 2002, the task of single-document summarization for news articles was lowered from DUC. Svore et al. [10] introduced a technique with neural networks and the use of third party data sets to tackle the problem of extractive summarization, outperforming the baseline with statistical significance.

The authors used a dataset of 1365 documents collected from CNN news web site, which consists of the title, time stamp, three or four human generated story highlights and the article or content. Three machine generated summaries are considered for the task. The human extracted summaries were not verbatim extractions from the article itself. They were evaluated the system using two metrics. The first one linked the three highlights produced by the system, concatenated the three human generated highlights, and compared these two blocks. And the second metric considered the ordering and compared the sentences on an individual level.

B. Lexical Chain

Human summarizers fabricate a constructed mental representation (theme) of the document and integrate the document based on the theme to produce the summary. The conventional computational techniques used word count measure [11] to determine the theme of the document. The motivation of this approach was that frequent words with the critical information of the text. One primary drawback of this proposal is that it does not emphasizes the importance of a word in the given context. Lack of consideration fails to capture the "context" or the "theme" of the document. For example,

1. **"Mr. Jimmy has invented an anesthetic machine. This device restrains the rate at which an anesthetic is pumped into the blood".**
2. **"Mr. Jimmy has invented an anesthetic machine. The doctor spent two years on this research".**

Both texts contain the same frequency of the words "Mr. Jimmy" and "machine", but the first text emphasizes the machine whereas the second one highlights Mr. Jimmy. This distinction can only be made by considering the relation between the words in the text (e.g. machine and device in first text).

The cohesion of Halliday and Hasan [13], enables us to capture the "theme" of the document. It can be determined as the property of the text to attach together as one large grammatical unit, based on relations between words. For example,

3. Wash and core six cooking potatoes.

4. Put them into the dish.

In the above set of sentences, the second sentence refers to the potatoes in the first one. This property of cohesiveness is not visible between un-related sentences. For example,

5.Wash and core six cooking potatoes.

6.Toronto is the biggest city in Canada.

Cohesion relations affect the comprehensibility of the text [14]. The cohesive structure can be described as graphs with elements of the text as the nodes and relations between the elements as edges connecting the nodes. Boldness of the information can then be determined based on the interacted relations of the nodes in the graph.

Halliday and Hasan [13] separated cohesive relations into the following categories:

Reference: reference relations, commonly, include the usage of pronouns in order to refer an entity mentioned in the preceding or the following text. In the following example, he and Mark both refer to the same person "Mark".

7. Mark went to England. He had to attend a conference.

Substitution: the relations in which one particular phrase or word is replaced with an article such as one or several etc. In the following example, "several" is substituted to refer the word "car".

8. I bought a new car yesterday. And there were several I could have had.

Ellipsis: the relations established by rejection of certain phrases or words. In the following example, the word "distant" is not mentioned for the second time.

9. New York is as distant from San Francisco as Boston is [distant] from London.

Conjunction: the relations attained by using connectors to describe the relationships between statements.

10. He gave me the directions but I lost it.

11. When you have done, we shall leave.

1. WordNet 3.1

WordNet 3.1 is a large lexical database of Englisha words that are linked together by their semantic relationships. It is like a supercharged dictionary or thesaurus with a graph structure. Moreover, nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets). Synsets are words interlinked by means of conceptual-semantic and the lexical relations. The subsequent network of relevantly related words and concepts can be directed with the browser. WordNet is freely and openly available for download. The structure of WordNet makes it a useful tool for computational linguistics and natural language processing.

WordNet apparently resembles a thesaurus, which means that it groups words together based on their lexical meanings. However, there are still some important distinctions. Firstly, WordNet interconnects not just word forms, strings of letters, but specific meanings or senses of words. Words, eventually, that are found in close proximity to one another in the network are semantically disambiguate. The second obstacle which WordNet labels the semantic relations among words, whilst the groups of words in a thesaurus do not follow any definite pattern other than similar meanings.

In WordNet, each “word form” represents some explicit lexical "meaning". Some word forms represents several meanings and some meanings can be represented by various forms. Table 1 shows the concept of lexical matrix which used to map word meanings to word forms. The entry 'Ei,j' in the lexical matrix symbolizes that word form 'WFj' refers to the meaning 'Mi'.

Table 1. Mapping between words form and lexical meanings

Word Meaning	Word Forms			
	WF1	WF2	WF3	WFn
M1	E1,1	E1,2		
M2	E2,1		E2,3	
M3		E3,2		
Mj	Ej,1	Ej,2		Ej,n

Word forms representing to the same underlying concept can be determined as synonymous for instance {WF1.WF2}. WordNet arranges the word forms belonging to the same syntactic category that refer to the common underlying concept into synonym sets called synsets. As an example, the synset notation defines its definition as standard, criterion, measure, touchstone, and it refers to the lexical meaning "a basis of comparison". Word forms that refer to more than one underlying concept are called polysemous (WF1). Table 2 represents the number of synsets and number of unique strings, and the total word-sense pairs, and Table 3 indicates number of Monosemous and Polysemous, and their senses, Table 4 shows the number of average polysemy including monosemous and the number of average polysemy without monosemous in WordNet 3.1.

Table 2. Number of Unique Strings, Sysnsets and Total Word-Sense Pairs in WordNet 3.1

POS	Unique Strings	Synsets	Total Word-Sense Pairs
Noun	117798	82115	146312
Verb	11529	13767	25047
Adjective	21479	18156	30002
Adverb	4481	3621	5580
Totals	155287	117659	206941

Table 3. Statistics of words and senses of Monosemous and Polysemous in WordNet 3.1

POS	Monosemous Words and Senses	Polysemous Words	Polysemous Senses
Noun	101863	15935	44449
Verb	6277	5252	18770
Adjective	16503	4976	14399
Adverb	3748	733	1832
Totals	128391	26896	79450

Table 4. The average number of Polysemy with and without Monosemous words

POS	Average Polysemy with Monosemous Words	Average Polysemy without Monosemous Words
Noun	1.24	2.79
Verb	2.17	3.57
Adjective	1.40	2.71
Adverb	1.25	2.50

WordNet connects the synsets by certain lexico-semantic relations in Table 5. The most significant relation is hypernym and hyponym relation for a term, in which a synset is a whole class or member of class of another synset. Hypernym and hyponym relation organizes nouns and verbs into 11 and 512 hierarchies. The hierarchical construction of synsets can be seen in Figure2. Generality of concepts increases while traversing upwards in the hierarchical structure. Figure 3 shows the WordNet entry for the word modification.

Table 5.Sample WordNet Relations

Relation	Examples
Synonym	weather - atmospheric condition
Hypernym/Hyponym	car - vehicle
Antonym	good - bad
Meronym/Holonym	steering - navigation

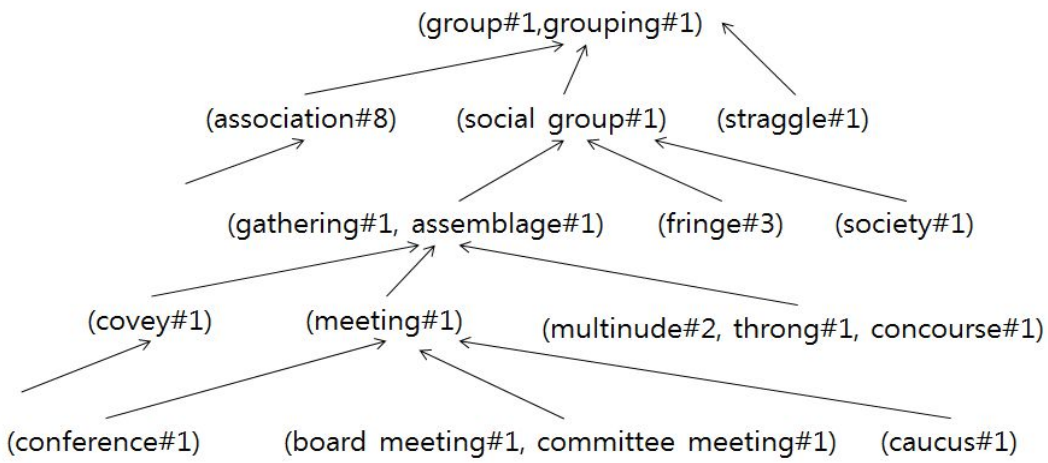

Figure 2. WordNet hierarchical structure

Gloss of each synset contains definition(s), comment(s) and some example(s) for the underlying lexical concept. For example, the gloss of the synset of weather, weather condition, atmospheric condition includes the definitions such as the meteorological conditions: temperature and wind and clouds and precipitation, and that is followed by example sentences such as "they were hoping for good weather", "every day we have weather conditions and yesterday was no exception".

The definitions of the gloss can be used to describe the interacted relations between two concepts not directly related using direct WordNet relations. For example, let us consider the words “dormitory” and “university”. There is no straightforward relation in WordNet between the two words although the relation can be identified by humans. According to the gloss of the word dormitory, the meaning of it is "a college or university building containing living quarters for students", so that we can execute a relation between the two words (we also concede the relation between dormitory and students from the same the semantic definition).

Lesk [15] complied the existence of the gloss concepts of a word in the current surroundings to lessen the sense of the word being used in current

context. Banerjee and Pedersen [16] examined more further and measured the semantic relatedness between two concepts based on their gloss definition overlap within WordNet lexical database. Harabagiuand Moldovan [17] emphasized the related gloss concepts to gather the information not explicitly stated in the text.

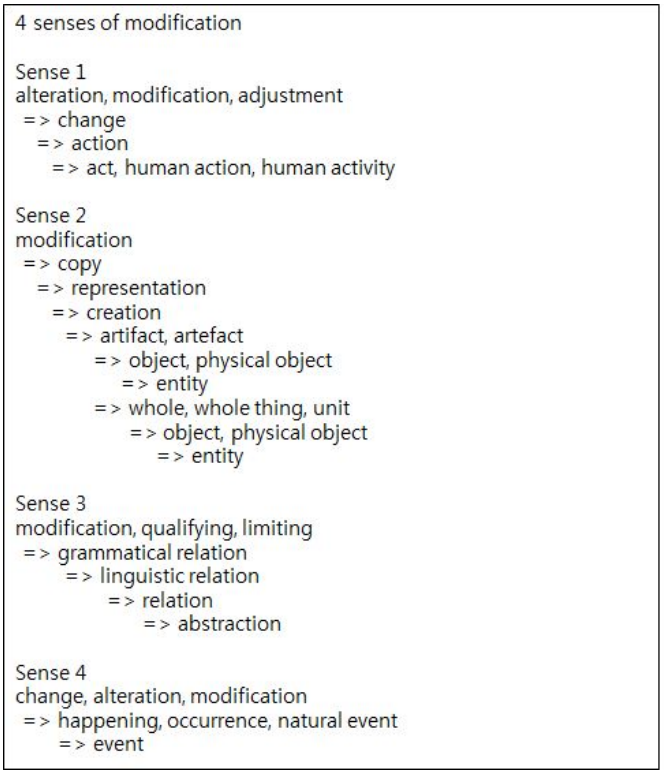

Figure 3. WordNet entry for the word modification

C. AUTOMATIC KEYWORD EXTRACTION

Automatic keyword extraction is the process to identify a small set of words, key phrases, keywords, or key segments from a text or document that defines the meaning of the document [18]. It should be processed by systematically and with either minimal or no human intervention, relies on the model. The goal of automatic keyword extraction is to apply the capacity and performance of

computation to the problems of access and discover ability, adding value to information organization and retrieval without the substantial costs and drawbacks associated with human indexers [19].

1. Motivation

As a result of rapid growth of diffusion of information technology, the spread of stored information across the Internet has been hastily increasing day after day. Large amount of stored information can be found in various forms of data contents such as metadata, text documents, images, audio files and so on. Among those data types, documents are known as unstructured data, and there are several types of documents in divergent ways of writing methods which can be collected from different sources like social media and news websites, or personal blogs. Not only text analytics processes within Natural Language Processing (NLP) to deal with such documents for taking further NLP operations, but also the efficient of common search engines rely on the amount of those collected documents. Keyword extraction plays a vital role in document retrieval, text mining, document categorization or classification, automatic summarizing, indexing, and other processes of deriving high-quality information from text summarily and accurately.

Keywords are a set of the most informative and relevant words or phrases of a document which imply the context of the entire document. Furthermore, keywords are essential Search Engine Optimization (SEO) elements for every search engines, yet they are matched against with users' search query keywords. Also they are extremely momentous for document surrogates built with metadata which contains general information of its respective data objects such as title, abstract or summary and so on. In order that, assigning the right keywords to a document is the main task and the most crucial step to perform further advanced procedures. In case of failure in distributing the right keywords in a first place, the future proceedings will be on tough road with a lot of time consuming and costly to manually extract the right ones from the beginning. There are several methods to extract the keywords from single

document the main concept of the most common technique is scoring each term in a tex twith a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

The proposed method acquires the most significant characteristics (features) of keywords to construct the model and the system relies on the probability distributions of each feature generated by Transition Probability Distribution Generator(TPDG). The TPDG values are calculated based on Naive Bayes' Bayesian probability terminology for each feature for each n-gram term, thus, not only the characteristics of a keyword but its properties can also be learnt by the model, and will be represented in probability statistics values. But then those probabilities are used as transition matrices for each different state where the three distinct features; the TFIDF score of each term over the average score, the occurrence of a term within the first sentence of the text and the occurrence of a term in the title of the text, represent as conditions for a state space (Keyword, Non-Keyword) in a Markov Chain to estimate and label a term can be in either 'Keyword' or 'Non-Keyword' state[58]. The detail description of the proposed method is described in subsections below.

2. Existing Approaches

There are several methods to extract the keywords from single document the main concept of the most common technique is scoring each term in a text with TFIDF score, a numerical statistic which reflects how important a word is to a document in a collection or corpus. The weighting score algorithm has been widely applied to many various text processing approaches as a fundamental mechanism yet enough powerful method to score the important terms which carry the notable information of the text. Term Frequency (TF) indicates the number of occurrences of a term within a text, and Inverse Document Frequency (IDF) is a measure of how much information the word provides, that is, whether, the term is common or rare across all documents,

was proposed by Karen Sparck Jones in 1972. The multiplication of both terms increases proportionally to the number of times a word appears in the document, but is neutralized by the frequency of the word in the corpus to adjust for the fact that some of the terms appear more frequently in general. There were several both supervised and unsupervised approaches taken to the automatic keyword extraction have been proposed.

Existing methods for automatic keyword extraction can be separated in four categories, i.e. simple statistics, linguistics, machine learning and other mixed approaches. Simple statistics approach like N-gram statistical information to automatic index the document [20]. N-gram is language and domain-independent and other approaches include word frequency[21], TFIDF [22], word co-occurrence [23], and PAT-tree [24], etc. Linguistics approaches principally emphasize on sentences and document including the lexical analysis [25], syntactic analysis [26], discourse analysis [27][28][52][53] and etc. Some supervised machine learning approaches for keyword extraction are implemented to learn a model from training set and applies the model to extract the keywords from test documents. Such approaches enclose Naive Bayes [29], SVM [30], etc. The other mixed approaches combine the approaches mentioned above together or use some heuristic knowledge to extract keywords [31].

(i) Linguistic Approaches

The linguistic approaches utilize the linguistic features of the words, sentences or text. Methods which emphasize to linguistic features such as part-of-speech, syntactic structure and semantic qualities tend to increase the value, functioning sometimes as filters for inappropriate keywords.

Plaset al. [32] applied two lexical resources for evaluation, the EDR electronic dictionary, and WordNet of Princeton University. Both provide densely populated lexicons including semantic relationships and links such as IS-A and PART-OF relations and concept polysemy. During automatic keyword extraction process from multiple dialogue episodes, the advantages of using the lexical resources

are compared to a pure statistical method and relative frequency ratio. Hulth [33] has tested a few different methods of incorporating linguistics into keyword extraction. Terms are considered as keywords based on 3 features: the term frequency in a document (TF), collection frequency (IDF), the position of its first occurrence in a document and its part-of-speech tag. The results showed that the use of linguistic features indicate the remarkable improvement of the automatic keyword extraction. However, some of the linguistic methods are combined methods, mixing some linguistic methods with common statistical measures such as term frequency and inverse document frequency.

(ii) Machine Learning Approaches

Automatic Keyword extraction can be seen as supervised learning from the examples or training corpus. The machine learning mechanism works as follows. First a set of training documents is provided to the system, each of which has a range of keywords which are carefully extracted by humans manually. After that, the gained knowledge is applied to find keywords from new documents or test documents depending on the model whish was built according to the result of training section. The Keyphrase Extraction Algorithm (KEA) [34] compromises the machine learning techniques and naive Bayes formula for domain-based extraction of technical keyphrases. Suzukiet al. [35] used spoken language processing techniques to extract keywords from news using an encyclopedia and newspaper articles as a guide for relevance.

The process is divided into two phases: term-weighting and keyword extraction. A set of feature vectors, firstly, is generated from different encyclopedia domains. Then the same procedure is processed on a corpus of newspaper articles. The vectors from encyclopedia are matched with the vectors of the article using a similarity computation in order to distinguish the latter into different domains, after thay which they are separated, producing the final set of feature vectors. In the second phrase, the keyword extraction process, a segment is analysed for the most relevant topic is preferred for it in order to

use the pre-existing feature vectors. The phoneme recognition software is implemented to do the analysis, looking for the best fit between a segment's vectors and that of one of the encyclopedia domains. When the best relevant domain is chosen, its keywords are then assigned to the news segment.

(iii) Mixed Approaches

The other approaches about keyword extraction mainly blend the methods mentioned above or use some heuristic knowledge in the task of keyword extraction, such as the position, length, layout feature of the words, html tags around of the words, etc[36]. The overview of the related works discloses that the automatic keyword extraction is faster and less expensive than human extraction. Moreover, the researchers claim that it achieves the precision of the human extracted keywords. Currently, existing solutions, however, for automatic keyword extraction needs either training examples or domain specific knowledge. Some approaches do not have this additional information. We apply the statistical measures to the automatical keyword extraction as they are domain independent and have limited requirements.

Ⅲ. PROPOSED METHOD & SYSTEM IMPLEMENTATION

Summarization, as carried out by humans, can be divided into 2 stages mentioned in Jones in 1993:

- Building of intermediate representation
- Synthesis of intermediate representation to generate summary

According to the researches, the lexical chains can be used as an efficient intermediate representation for the textual documents. We implemented a system to compute lexical chains as an intermediate representation for the source and to summarize the text in shorten form by extracting satisfied certain sentences. As the distinct nouns and noun phrases of the text are the main factor to construct the efficient lexical chain, we developed a new approach, for extracting and assigning the most relevant keywords/terms of the text, which assigns the keyword or non-keyword state for each term in text and seeks the most important keywords using our Transition Probability Distribution Generator (TPDG) which learns the characteristics of the keywords upon on three distinct features. The detail information of TPDG is discussed in section below. The overview architecture of our system can be seen in Figure 4 and Figure 5.

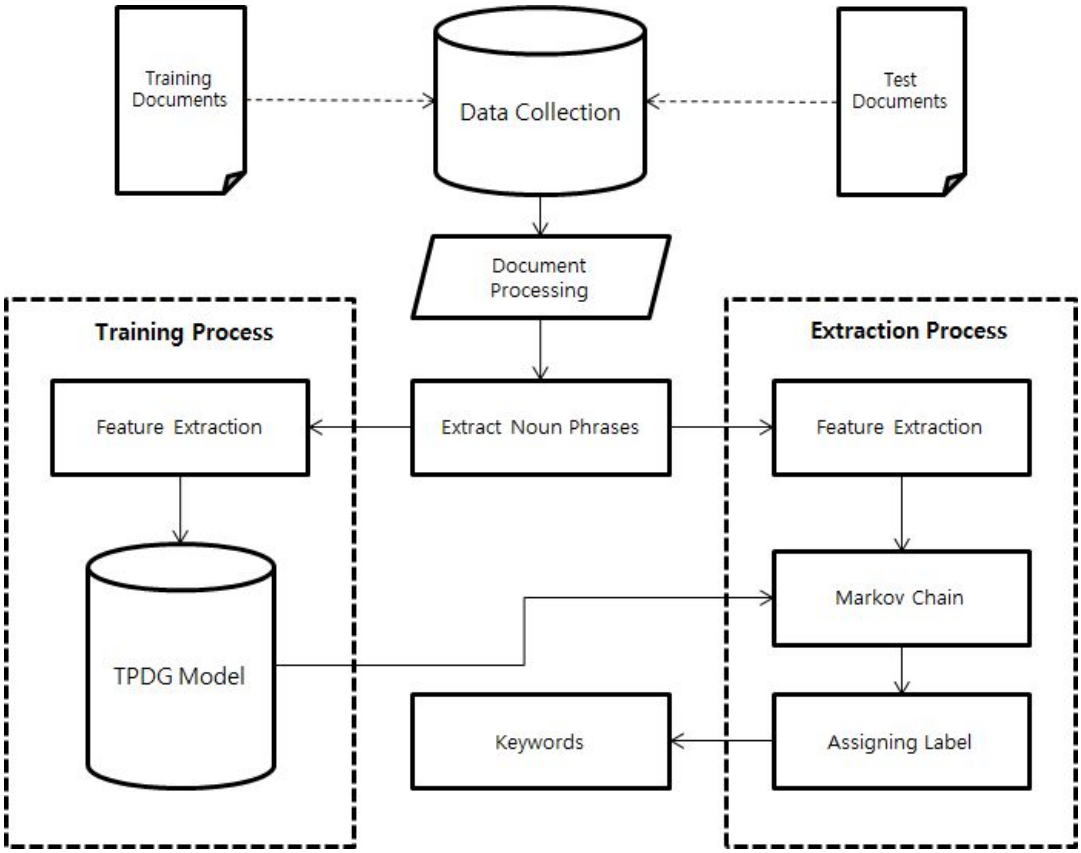

Figure 4.Automatic Keyword Extraction using TPDG Model

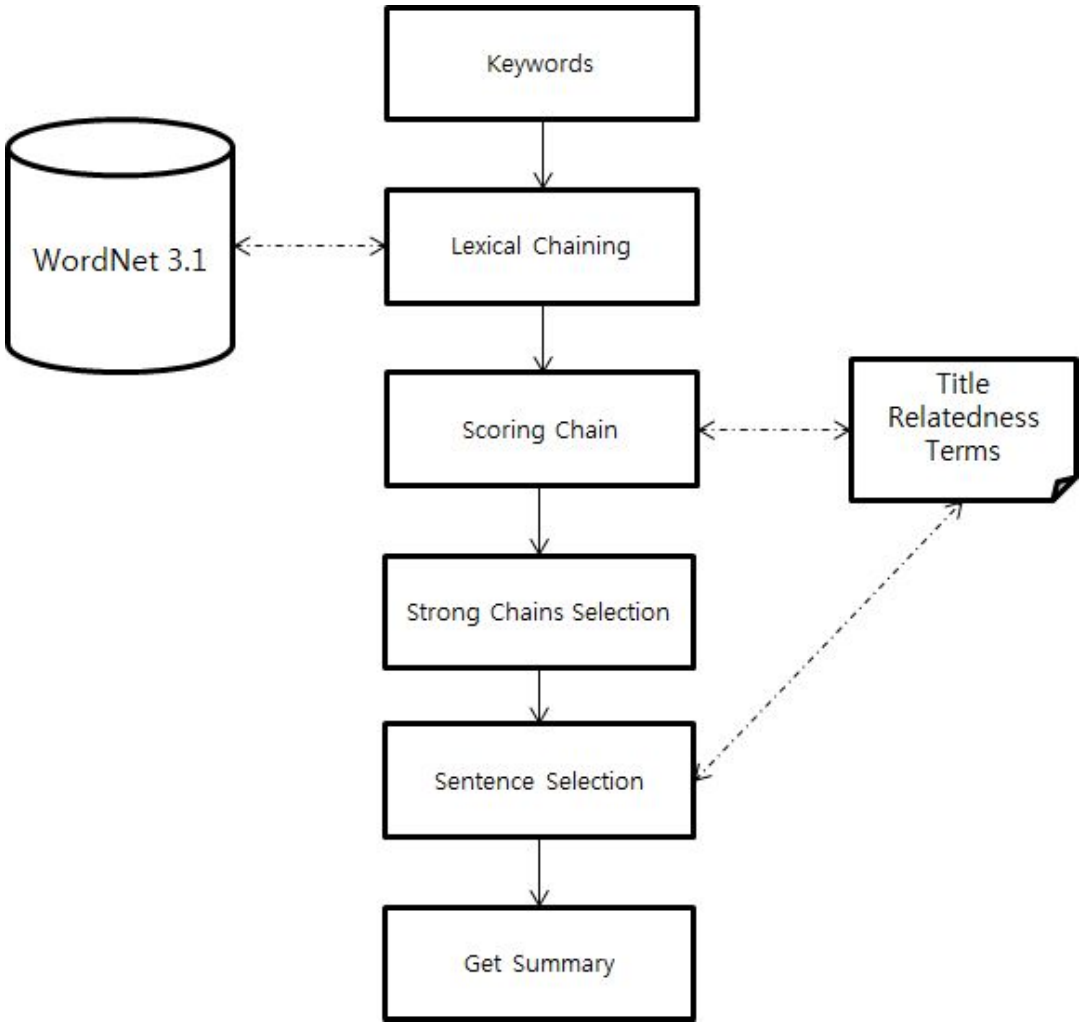

Figure 5.Extracting Summary Sentences from Lexical Chains

A. Data Collection

There are training corpus and test corpus in the data collection. Both data sets are stored in specific way of for different procedure, and collected from various sources where the characteristics and style of writing were different from each other as are from news media like The New York Times, CNN, BBC News and technology news media like Tech Crunch and Mashable and so on. A training document consists of 3 main information; Title, Contents and

manually assigned Keywords according to the range of its n-gram terms within a text while the assigned Keywords information will be excluded for Training documents. In training document, the data is stored in three set of information. The title should be the first line of the text, then the content is stored until the last sentence which are the manually assigned keywords for the given document. The sample format of training document is shown in Figure 6. Likewise, the document for test data or extraction process is stored in two sets of information; it contains Title in the first sentence, and it is followed by its content, but the manual assigned keyword section will be excluded for test data (Figure 7).

```
Uber Invites Developers to Build Apps for Customized Passenger Distractions

Uber, the huge ride-hailing service, delivers millions of rides to passengers.
Now the company wants to give people something to do while they're in the car.
Uber has long been selective about how it works with partner apps and companies.
.
.
.

Uber, app developers, UberRush,new integration
ideas,passengers,smartphone,app,developers,platform,idea,integrations,smartphone app,delivery service,new
integration,test release,handful retailers,the user experience,repeat business affinity,the company brand
```

Figure 6. A sample format of training document

```
iPhone SE: thoughts on going back to a smaller phone

I've been using the new iPhone SE for a couple days now, after having a 6s for a while, and I have to be
honest: going back to using a small phone feels weird.
I'm convinced that Apple has to be aiming this new phone at people who either love small phones or want a
reasonably-priced upgrade from a three-year-old iPhone.
Let's face it, a new iPhone is tempting.
It's a new iPhone! But if you've already graduated to a bigger phone, this phone might not be for you.
.
.
.
```

Figure 7. A sample format of test document

B. Document Preprocessing and Methods

Preprocessing method plays a very important role in text mining techniques and applications. It is the first step in the text mining process or for any further text processing performances. The documents in both training and test corpus are performed some essential fundamental text processing tasks to transform into representation of bag-of-words for n-gram terms of a document ignoring the term order.In this section, we discuss the four key steps of preprocessing namely of our proposed method, sentence segmentation, stop words removal, word tokenization, and Part-of-Speech (POS) Tagging. The NLTK open source tool for Python 2.7 is applied to tackle those preprocessing text for both training and extraction process.

1. Natural Language Toolkit (NLTK)

NLTK is the most leading platform for building Python programs to work with human language data. It includes user friendly interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Thanks to a user friendly guide commencing thr programming fundamentals together with the topics in computational linguistics, and the comprehensive API documentation, NLTK is suitable for linguists, engineers, students, researchers, and industry users and so on. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project. NLTK has been called a wonderful tool for teaching, and working in, computational linguistics using Python, and an amazing library to play with natural language.

NLTK was primarily established in 2001 being part of a computational

linguistics course in the Department of Computer and Information Science at the University of Pennsylvania. After that, it has been utilized and expanded. Now, it has been accepted in courses in many universities, and serves as the basis of many research projects. Table6 illustrates the most important NLTK functionality.

Table 6. Language processing tasks and corresponding NLTK modules with example of functionality

Language Processing Task	NLTK Modules	Functionality
Accessing corpora	nltk.corpus	Standardized interfaces to corpora and lexicons
String processing	nltk.tokenize, nltk.stem	Tokenizers, sentence tokenizers, stemmers
Collocation discovery	nltk.collocations	t-test, chi-squared, point-wise mutual information
Part-of-speech tagging	nltk.tag	n-gram, backoff, Brill, HMM, TnT
Classification	nltk.classify, nltk.cluster	Decision tree, maximum entropy, naive Bayes, EM, k-means
Chunking	nltk.chunk	Regular expression, n-gram, named entity
Parsing	nltk.parse	Chart, feature-based, unification, probabilistic, dependency
Semantic interpretation	nltk.sem, nltk.inference	Lambda calculus, first-order logic, model checking
Evaluation metrics	nltk.metrics	Precision, Recall, Agreement coefficients
Probability and estimation	nltk.probability	Frequency distributions, smoothed probability distributions
Applications	nltk.app, nlotk.chat	Graphical concordancer, parsers, WordNet browser, chatbots
Linguistic fieldwork	nltk.toolbox	Manipulate data in SIL Toolbox format

NLTK was designed with 4 fundamental goals:

Simplicity: Providing an intuitive framework along with substantial building

blocks, giving users a practical knowledge of NLP without getting deserted in the tedious house-keeping usually associated with processing annotated language data.

Consistency: providing a uniform framework with consistent interfaces and the data structures, and easily predictable method names

Extensibility: To offer a structure into which new software modules can be easily integrated including alternative implementations and competing approaches to the same task

Modularity: providing the components which can be used independently without requiring to understand the rest of the toolkit[57].

2. Sentence Segmentation

The text is available only as a stream of characters. Before tokenizing the text into words, it is needed to segment into separated sentences. In English and some other languages, using punctuation, particularly the full stop "." or period character is a reasonable approximation. However, even in English this problem is not efficient due to the use of the full stop character for abbreviations, which may or may not also terminate a sentence. As an example "Dr." is cannot be considered a sentence in "Dr. John Williams went to the hospital in Central Street." When performing on a plain text, tables of abbreviations which include the periods can help obstruct the incorrect assignment of sentence boundaries. Figure 8 shows an example of its use in segmenting the text of a sample document.

```
>>sent_seg = nltk.sent_tokenize('doc1.txt')
>>print sent_seg

['GoPro stock dropped 23.34 percent 10.87 Wednesday...',
'The stock trading time low.', 'The stock briefly jumped 11.20 trading halted.',
'The company market cap 2 billion hour values.',
'GoPro stock dropping the company faced mounting pressure….',
'As the company laying 7 percent 1 500 people 105 people.',
'As today press release states GoPro previously experienced…']
```

Figure 8. A Sample of Sentence Segmentation algorithm

3. Stop words Removal

Stop words are a division of natural language. The motive that stop-words should be removed from a text is that they make the text look heavier and less important for analysts. Removing stop words reduces the dimensionality of term space. The most common words in text documents are articles, prepositions, and pro-nouns, etc. that does not give the meaning of the documents. These words are treated as stop words. Example for stop words; the, in, a, an, with, etc. Stop words are removed from documents because those words are not measured as keywords in text mining applications [37]. There are four types of stop word removal methods; the classic method, methods based on Zipf'sLaw(Z-Methods), the mutual information method (MI), and term based random sampling (TBRS). In our system, we apply the first, classic method which is based on removing stop words obtained from pre-compiled lists [38].

4. Word Tokenization

Word tokenization is the problem of dividing a string of written language into its component words. Not only In English but also many other languages using some form of Latin alphabet, the space is a good approximation of a word divider (word delimiter). Tokenization is a kind of pre-processing in a sense; an identification of basic units to be processed. It is conventional to concentrate on

pure analysis or generation while taking basic units for granted. Yet without these basic units clearly segregated it is impossible to carry out any analysis or generation. After tokenizing the words, Part-of-Speech tagging process can be carried out for each term in a sentence of the text in next step. Figure 9 indicates the sample result of word tokenization in the system carried out by NLTK.

```
>> token_word = [nltk.sent_tokenize(t) for t in text]

[['GoPro', 'stock', 'dropped', '23.34', 'percent', '10.87',
'Wednesday', 'afternoon', 'the', 'company', 'announced', 'Q4',
'worse', 'expected', '.'], ['The', 'stock', 'trading', 'time',
'low', '.'], ['The', 'stock', 'briefly', 'jumped', '11.20',
'trading', 'halted', '.'], ['The', 'company', 'market', 'cap', '2',
'billion', 'hour', 'values', '.'],…]]
```

Figure 9. A Sample Output of Word Tokenization

5. Part-Of-Speech(POS) Tagging

A part-of-speech tagger, or POS tagger, processes a sequence of words, and attaches a part of speech tag to each word. Tagged corpora use many different conventions for tagging words. It also known as grammatical tagging or word-category disambiguation, and it is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its context. As am example, its relationship with adjacent and related words in a phrase, sentence, or paragraph. A simplified form of this is normally taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs, etc[59]. For example,

"They refused to permit us to obtain the refuse permit"

POS Tag: [('They', 'PRP'), ('refused', 'VBP'), ('to', 'TO'), ('permit', 'VB'), ('us', 'PRP'), ('to', 'TO'), ('obtain', 'VB'), ('the', 'DT'), ('refuse', 'NN'), ('permit', 'NN')]

6. Noun Phrase Extraction

As the result of evidences that noun phrases within the text mostly contain the important information or context of the document, and the keywords are primarily formed as a noun phrase according to the surveys, the program is limited to extract noun phrases for n-grams (where 1<= n <= 4) matching the POS patterns. The process of extracting noun phrase can also be known as Noun Phrase Chunking as it involves recognizing the chunks that consist of noun phrases (NPs). One of the most beneficial roots of information for NP-chunking is part-of-speech tags. This is one of the encouragements for performing part-of-speech tagging in our information extraction system. In order to create an Noun Phrase, we will first define a chunk grammar, consisting of rules that indicate how sentences should be chunked. Each string with assigned POS tag from previous stage is applied 35 grammatical rules of noun phrase pattern in regular expression to extract the noun phrases by matching over the exact POS tags. Table 7 shows an example of POS patterns of Noun Phrase for n-grams applied on our system.

Table 7. POS Patterns of Noun Phrase for N-grams

N-gram parameter	POS patterns for Noun Phrase
Unigram/1-gram	“NNS”
	“NN”
	“NNP”
Bigram/2-gram	“NNP” + “NNS”
	“WP$” + “NNS”
	“CD” + “NNS”
	“NN” + “NN”
	“NN” + “NNS”
	“NNP” + “CD”
	“NNP” + “NNP”
	“NNP” + “NNPS”
	“NNP” + “NN”
	“NNP” + “VBZ”
	“DT” + “NNS”
	“DT” + “NN”
	“DT” + “NNP”
	“JJ” + “NN”

	“JJ” + “NNS”
	“PRP$” + “NNS”
	“PRP$” + “NN”
Trigram/3-gram	“NN” + “NN” + “NN”
	“NN” + “NNS” + “NN”
	“NNP” + “NNP” + “NNP”
	“JJ” + “NN” + “NNS”
	“PRP$” + “NN” + “NN”
	“DT” + “JJ” + “NN”
	“DT” + “CD” + “NNS”
	“DT” + “VBG” + “NN”
	“DT” + “NN” + “NN”
	“DT” + “NNP” + “NN”
	“DT” + “NNP” + “NNP”
	“DT” + “JJ” + NN”
	“NNP” + “NNP” + “VBZ”
4-gram	“DT” + “NNP” + “NNP” + “NNP”
	“DT” + “NNP” + “NN” + “NN”

C. Feature Extraction

The proposal is intended to interact with web contents which are quite different from technical writings and format. Web contents commonly do not include Abstract, Introduction, and keyword section like in technical papers. Thus, we rather hold the most possible significant features of keyword to lessen the scope of broad characteristics. Three significant features consists of the weight of a term over the average score of the text using TFIDF method, the occurrence in first sentence of the text and the occurrence in title. First of all, the extracted unigram noun phrase are assigned by their corresponding weight using TFIDF method below.

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \qquad (2)$$

, where tf(t,d) represents the term frequency(tf) of term (t) in document (d), and idf(t, D) is inverse document frequency of term (t) in the corpus (D). Instead of getting the raw frequency of each term in a document, the log normalization form of term frequency is applied to reduce the effect of multiple occurrences of a term.

$$tf(t,d) = 1 + \log(ftd) \qquad (3)$$

, where (ftd) is raw frequency of term (t). Likewise, the inverse document frequency of each term is also applied into log normalization form as below.

$$idf(t,D) = \log(1 + \frac{N}{n_t}) \qquad (4)$$

, where N is the total number of documents in corpus and nt is the number of documents term t occurs. TFIDF scores for n-grams more than unigram is simply the total sum of the score of each member in a term. The TFIDF score of each term is calculated and compared to threshold (average TFIDF score) to denote whether it scores higher than the average score or lower. The average TFIDF score is computed as the sum of total TFIDF scores in document divied by the total number of terms in the given document. If the weight of a term scores higher than the threshold, the term is recorded with its corresponding

weight and its TFIDF feature representation becomes 1 or 0 for terms with lower score.

Another important characteristic of a keyword is the occurrence of a term in the first sentence. Most of the web contents generally introduce the main idea or context of the document in a very first sentence of first paragraph. Finally, the most accurate and effective way to distinguish a term a keyword is its appearance in title of the document. The representations for both features are also be denoted in binary values (0 or 1).

D. Constructing Transition Probability Distribution Generator (TPDG)

The idea of building TPDG Model is to produce the posterior probabilities of state/class (Keyword/Not Keyword) given predictor (features) using Naive Bayesian classifier algorithm based on Bayes' Theorem. Bayes' Theorem provides a way of calculating the posterior probability, P (S|F), from P(S), P(F), and P(F|S), where P(S|F) is posterior probability of State (S) given Features (F), P(S) is the prior probability of State (S), P(F) is the prior probability of Features(F), and P(F|S) is the likelihood which is the probability of Features(F) given State(S). Naive Bayes classifier assumes that the effect of the value of a feature (F) on a given state (S) is independent of the values of other features. This assumption is also called class conditional independence. In TPDG, the generator produces the posterior probabilities of Keyword or Not Keyword state of each given feature for each n-grams term by learning the characteristic of the keywords from training corpus. The Bayes' posterior probability of being Keyword given terms with over average TFIDF score can be written as:

$$P(K|TFIDF) = \frac{P(TFIDF|K)P(K)}{P(TFIDF)} \qquad (5)$$

where P(TFIDF|K) is the likelihood which is the probability of TFIDF feature given Keyword state, and it can be written as;

$$P(TFIDF|K) = \frac{N_{ktfidf}}{N_k} \quad (6)$$

, where is the number of terms with over average TFIDF score being identified as Keyword, and is the number of terms which are Keyword. Since there are certain likelihood probabilities for other possible conditions, we defined the likelihoods in total of four probabilities for TFIDF feature, and they can be described as:

$$P(TFIDF|NK) = \frac{N_{ktfidf}}{N_{nk}} \quad (7)$$

$$P(NTFIDF|K) = \frac{N_{nktfidf}}{N_{nk}} \quad (8)$$

$$P(NTFIDF|NK) = \frac{N_{nkntfidf}}{N_{nk}} \quad (9)$$

, where K represents keyword state, NK is Not Keyword state, is the number of terms with over average TFIDF score which are Not Keywords, and is the number of terms which are Not Keywords. P(TFIDF) is the prior probability of TFIDF score feature, where the sum of both likelihood of keywords and non-keywords given TFIDF feature.

$$P(TFIDF) = P(TFIDF|K) + P(TFIDF|NK) \quad (10)$$

While P(NTFIDF) is the prior probability of non-TFIDF conditions where the sum of both likelihood of keywords and non-keywords give non-TFIDF condition.

$$P(NTFIDF) = P(NTFIDF|K) + P(NTFIDF|NK) \quad (11)$$

Therefore, we have 4 likelihoods, 2 prior probabilities of feature given states for each feature. It means as the proposal emphasizes on 3 distinct features of keyword mentioned in Section C, we will get 12 likelihoods and 6 prior probabilities of feature give states in total. But then, depending on those probabilities the system is able to calculate the 2 prior probabilities of the states P(K) and P(NK). Finally, the system distributes 4 posterior probabilities

for each feature, 12 in total for all 3 features to build 2x2 dimensional matrices as transition distributions to be interacted with Markov Chain process, and those transition distribution matrices for each feature for n-grams (T) is built as can be seen inFigure 10, the entries in the first column in the matrix indicates the probabilities of the Keyword state of both given feature is active or discharge by assigning the corresponding posterior probabilities, and the second column represents the probabilities of the Not Keyword state of given feature is active or discharge.

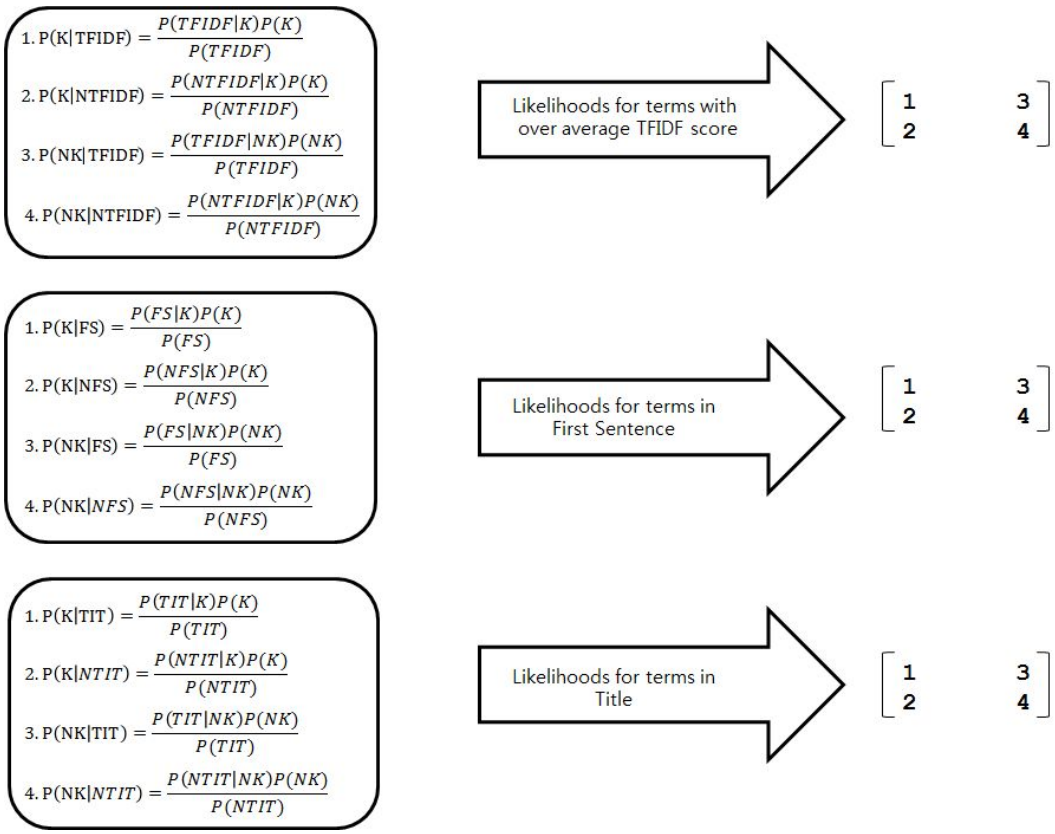

Figure 10. Transition matrices built with Likelihoods for each distinct feature

Finally, in total of 12 transition matrices (T) for n-grams are distributed for the extraction process. However, the transition matrix(T) do not follow the property of conventional Markov Chain model as the sum of the entries of each

row are not add up to 1, yet the actual probabilities of every possible condition of a keyword according to manually indexed training data are contained.

E. Assigning Keywords with Markov Chain using TPDG

Extraction process consists of a few same basis tasks as mentioned in Section 3, the corpus in test data are applied those similar procedures such as Preprocessing in Section B including sentence segmentation, Stopwords removal, word tokenization, POS tagging and Noun Phrase chunking. Then it proceeds to feature extraction in order to transform the document into set of noun phrases with their corresponding properties according to the significant features.

The Markov Chain model [51] is applied for extracting keywords on constructed document representation and it can be described as follows: We have a set of states S = {K, NK}. The process starts in one of these states according to their corresponding TFIDF values, and move successively from one state to another based on the probability values of each stage. There will be 3 stages of process in the system as the proposal method includes three best important features; TFIDF score, Occurrence in first sentence, Occurrence in Title. We assume that the occurrence of a term in a title has the most accurate characteristic of a keyword while the appearance of a term in first sentence quite lower than that, yet it is still better to assign a keyword rather than the other features like the length of a term, appearance in the last sentence, and so on. Hence, we designed three stages in an order.

1. TFIDF score
2. Occurrence of the term in first sentence
3. Occurrence of the term in title

The first entry of initial matrix(I) of a model is the TFIDF score of the term, and the second entry is the average TFIDF score within the document.

$$I = [TFIDF, avgTFIDF] \qquad (12)$$

As a term's initial matrix is compiled, the first stage of the process begins with TFIDF probability distributions which is generated by TPDG in Section D. The process returns the result in a 1x2 dimensional matrix, and again proceeds to second stage with the generated probability distributions for the condition of occurrence of the term in first sentence. Then another new conditional matrix is produced to operate the final stage of the process with the distributions of the occurrence of the term in title. The extraction process, Figure 11, is described as follows:

1. Ctfidf = I * Ttfidf (13)
2. Cfs = Ctfidf * Tfs (14)
3. Ctit = Cfs * Ttit (15)

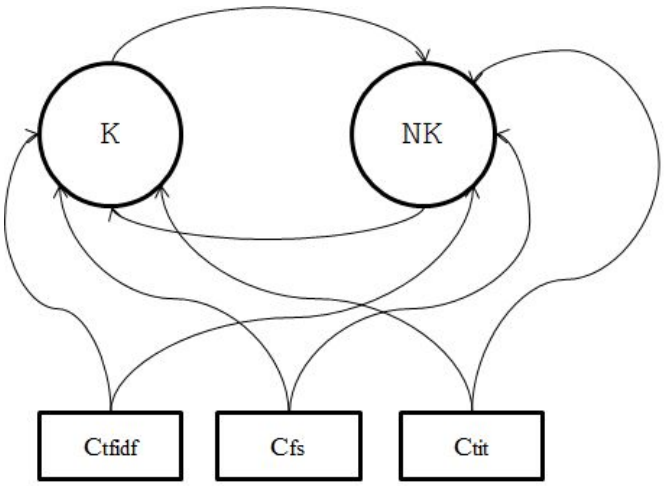

Figure 11.Markov Chain process of Keyword Extraction

, where C is the condition of a term, I is the initial state matrix, and T is the Transition matrix for each distinct feature generated by TPDG in Section D.

Lastly, the probability of a term being in keyword state is calculated by

multiplication of current state distribution matrix with the 3rd feature's probability matrix, and is defined in 1x2 matrix of blocks where the first entry represents the probability of being in Keyword state and the second entry defines the probability of being in Not Keyword state. For this reason, a term is denoted as keyword [K] if the first entry of the final matrix value is higher than the second entry, while a term is labelled as Not Keyword [NK] if the second entry is higher than the first in the labelling process to extract the keywords.

F. Building Lexical Chains

After assigning the candidate keywords, the system acquires the extracted unigram keywords in order to construct the Lexical chains. Our system's building lexical chains method is based on the method of Barzilay and Elhadad. The procedure for constructing lexical chains follows these steps:

1. Select a set of candidate keywords;
2. For each candidate keyword, find an appropriate chain relying on a relatedness criterion among members of the chain;
3. If it is found, insert the word in the chain and update it accordingly.

Moreover, there are 4 distinct relations finding an appropriate chain of each candidate keywords among the members of the chain. They are repetition, hyponym, hypernym and synset. Repetition is the co-occurrence of the given term. Hyponym is a word or phrase whose semantic field is included within that of another word, its hypernym. In simpler terms, a hyponym indicates a type of relationship with its hypernym. For example, Figure 12 indicates pigeon, crow, eagle and seagull are all hyponyms of bird which is their hypernym, and which in turn, is a hyponym of animal. And, the synsetis the set of synonyms of the given term. In the pre-processing step, all words that appear as a noun entry in WordNet are collected. Relatedness of words is determined in terms of the distance between their occurrences and the form of the path connecting

them in the WordNet lexical database. A term with one of each distinct relation with the given candidate keyword is added into the chain.

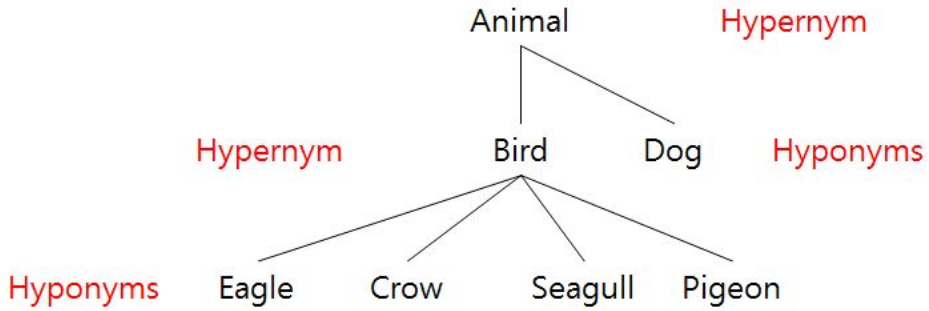

Figure 12. An example of the relationship between Hyponyms and Hypernym

If no chain is found furthermore, then a new chain is created and the candidate word is inserted with all its possible senses in WordNet. But the implemented greedy disambiguation method in this algorithm has some limitations described by the following example:

> **Mr.** *Kenny is the* **person** *that invented an anesthetic* **machine** *which supports the* **micro-computers** *in order to control the rate at which an anesthetic is pumped into the blood. Those* **machines** *are nothing new whilst his* **device** *compromises two* **micro-computers** *to achieve much closer monitoring of the* **pump** *feeding the anesthetic into the patient.*

Let's consider that we have our candidate keywords for the paragraph above as, {"Mr.", "person", "machine", "micro-computers", "machines", "device", "micro-computers", "pump"}. The lexical chain for the word "Mr." is first created ["Mr.", sense {mister, Mr.}]. "Mr." belongs only to one synset, so it is disambiguated from the beginning. The word "person" is related to this chain in the sense "a human being" by a synset relation, so the chain now contains two entries:

[lex "Mr.", sense {mister, Mr., man}]

[lex "person", sense {individual, someone, person, man, mortal, human, soul}].

But for the word "person", the previous relation with "Mr." is added, and the word "machine" which has the lexical relation as "an efficient person" is also inserted into its chain. So the chain for word "person" is generated as:

["person", sense {person, individual, someone, man, mortal, human, soul}].

["Mr.", sense {mister, Mr., man}]

["machine", sense {an efficient person}].

The lexical chain members for the word "machine" are "micro-computers" which is the hyponym of the "machine", and "machines" which is the plural noun of it, and "device" which also is the hyponym of the machine according to the synset relation of WordNet. The sample of lexical chains for word "Mr.", "Person" and "Machine" can be seen in Figure 13 [12].

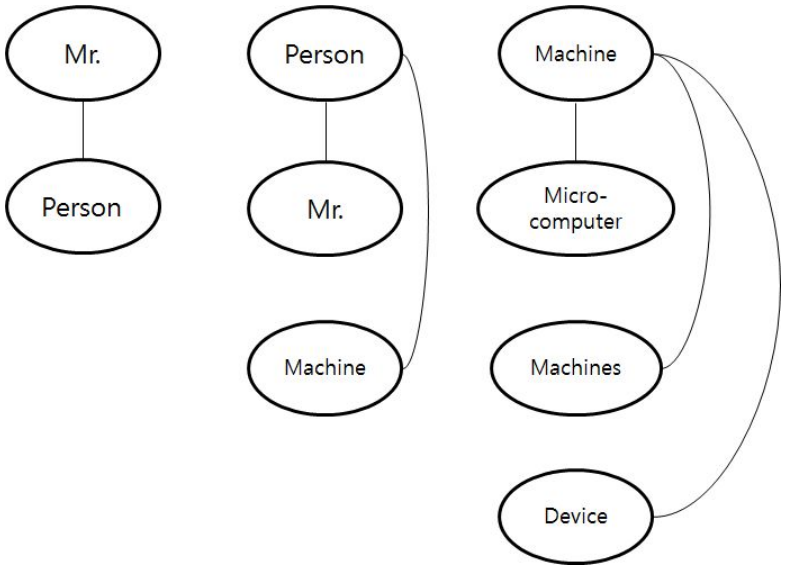

Figure 13.The lexical chains for word "Mr.", "Person" and "Machine"

G. Scoring Chains

Our scoring method for chains differs from the conventional method of Barzilay and Elhadad [12]. Our system emphasizes on the occurrence of the most important keywords of the content such as terms involved in title, and terms which are lexically related to title terms. We distinguish the set of exact terms in title and the set of related terms with title for our candidate unigram keywords, and then we set the value of 1 for each member term of a set of exact title term and 0.5 for members of set of title related terms. The following parameters are good predictors of the strength of a chain:

Length: The number of occurrences of members of the chain

Homogeneity index: 1 - the number of distinct occurrences divided by the length.

$$Score(Chain) = Length * HomogeneityIndex \quad (13)$$

After scoring a chain according to the formula above, the title relatedness terms values must be combined into its score chain. The title relatedness terms which we categorized in two groups; the exact term or co-occurrence of title term in a sentence, we set as 1, and the synonyms or related terms of title terms are set to 0.5 respectively. Then, the number of times of appearance of each member of that two groups, their set value will be added to the chain accordingly.

H. Strong Chain Selection

After scoring each lexical chain of the document, the strong chains are appointed upon their weight values whether their score higher than "Strength Criterion" value. The "Strength Criterion" value can be described as below which is applied in Barzilay and Elhadad's method:

$$Score(Chain) > Average(Scores) + 2 * Standard\,Deviation(Scores) \quad (16)$$

These are preliminary results but they are notably confirmed by their experiments on the 30 texts we analyzed extensively. The method was experimented with different normalization methods for the score function, but they do not seem to improve the results. Extending the empirical analysis in the future, and to support the formal learning methods to determine a good scoring function. According to the experiments of Barzilay and Elhadad, the average number of strong chains selected by this selection method was 5 for texts of 1055 words on average (474 words minimum, 3198 words maximum), when 32 chains were originally generated on average.

I. Sentence Selection

After the strong chains have been selected, the next pace of the summarization algorithm is to extract full sentences from the original document respected to the chain distribution. The sentences with the occurrence of each member of the strong chains are extracted, but then duplication of the exact same sentences will be ignored. Each of the sentences is considered the involvement of the title terms or the related title terms in it. The length of sentence and the number of title terms of related title terms occurrence within a sentence commit the ranking process in descending order. The very first best 7 sentences will be selected as summary sentences of the give text.

Ⅳ. EXPERIMENTAL EVALUATION

A. Evaluation Measures for Keyword Extraction

For our experiment in keyword extraction (TPDG), we compared our proposed keyword extraction method with three recent automatic keyword extraction approaches which are word co-occurrence method called Rapid Automatic Keyword Extraction (RAKE) [46] which used to extract keywords of a single document domain independently, and it basically relies on the term frequency, term degree and ratio of degree to frequency, and the second system is the CRF model [47], and the Support Vector Machine approach [48]. Our experimental data set contains in total of 600 various documents from different sources in terms of the writing style and format.

1. Experimental Results

In the evaluation, we followed the contingence table from C.Zhang's CRF Model [47]. There are two types of keyword collection which are manually extracted by humans, and the collection extracted by the keyword extraction methods. Each extracted keyword collection include two possible states which are keywords and non-keywords state. Table 8 shows the contingence table on the result of keywords extraction and manual assignment keywords.

Table 8. Contingence table of Extraction and Manual Assignment

	Manually assigned Keywords	Manually assigned Non-keywords
Keywords extracted by system	a	b
Non-keywords extracted by system	c	d

For our experiments, we processed the evaluations according to the general measuring method used in the Information Retrieval evaluation, i.e. Precision (P), Recall (R) and F1-Measure. The evaluation measures are defined as follows:

$$P = \frac{a}{a+b} \quad (19)$$

$$R = \frac{a}{a+c} \quad (20)$$

$$F_1(P,R) = \frac{2PR}{P+R} \quad (21)$$

The proposal outperforms well by producing better performance and meaningful keywords. We evaluate the performance of the four keyword extraction methods, i.e. CRF, SVM, RAKE, and our proposed method, by using the formulas for Precision, Recall and F1-measure mentioned above.

Table 9 shows the results of four keyword extraction models. According to F1-Score in Table 9, we can see that our proposed model outperforms the other approaches, and the result of the F1-Score comparison is: proposed model>RAKE>CRF>SVM.

Table 9. Performance Evaluation of Keyword Extraction

Model	Precision	Recall	F1-Score
SVM	0.8017	0.3327	0.4653
CRF	0.6637	0.4196	0.5125
RAKE	0.4929	0.5958	0.5329
Proposed model	0.5216	0.6304	**0.5709**

Figure 14 is a sample of a web document, while Figure 15 shows the sample keyword extraction of our system from the give text.

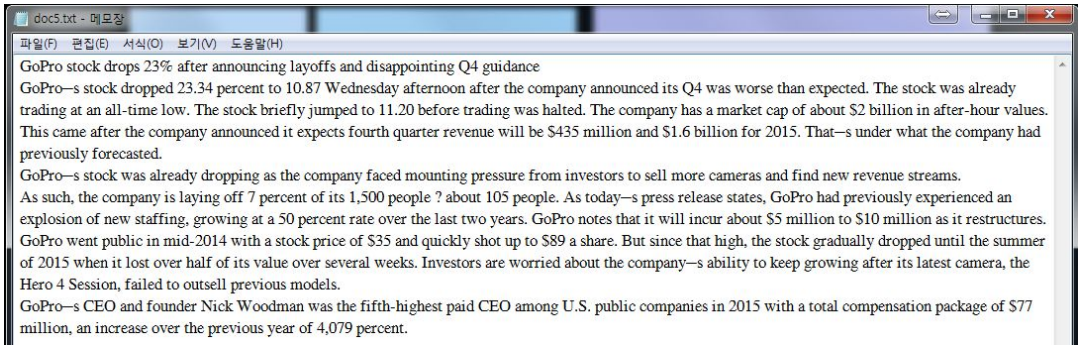

Figure 14. Sample of a Web Document

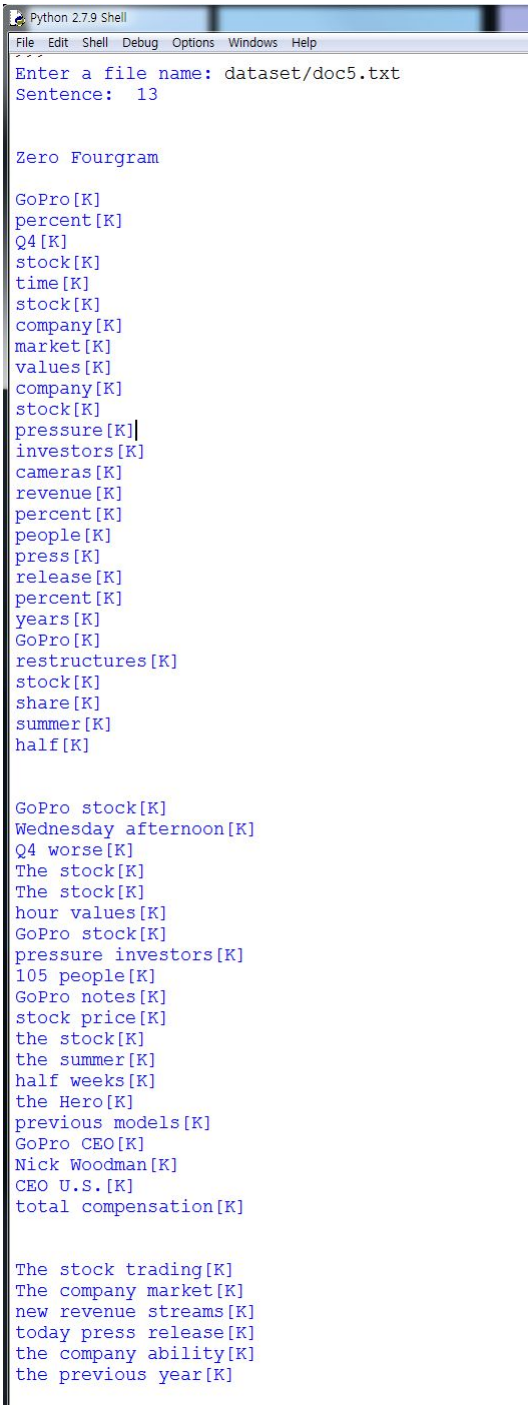

Figure 15. Candidate Keywords produced by the Proposed System

B. Evaluation Measures for Summarization

In evaluation of our proposed summarization, the evaluation methods can be broadly classified into two categories [39] intrinsic and extrinsic. The most common methods of evaluation, rate the summaries based on their ability to perform certain task (Information retrieval etc). Those methods of evaluation determine the quality of the summaries based on the overlap with human generated "ideal summaries". In the evaluation, precision and recall are the mainly used measures computed based on the number of units (sentences, words, etc) common to both system-generated and ideal summaries. Precision (P) is defined as the percentage of summary generated by system in common with the ideal summary.

$$P = \frac{|SystemSummary| \cap |ReferenceSummary|}{|SystemSummary|} \tag{22}$$

Recall (R) is defined as the ratio of the number of units (sentences or words) of the summaries extracted from systems in common with the ideal summaries to total number of units in the ideal summaries.

$$R = \frac{|SystemSummary| \cap |ReferenceSummary|}{|ReferenceSummary|} \tag{23}$$

Another formula, F-measure, is a composition score that uses (3 factor to weight the relative importance of precision and recall measures:

$$F-measure = \frac{(1+\beta^2)R^*P}{R+\beta^2 P} \tag{24}$$

We evaluated our summarization techniques using the test data collected from different sources such CNN, BBC UK, TechCrunch, New York Time and so on. We performed the automatic evaluation of summaries of our proposed system using ROUGE [40].

1. ROUGE

ROUGE (Recall-Oriented Understudy of Gisting Evaluation) is a system includes a collection of measures to evaluate the summaries by comparing them with "ideal" summaries automatically, without much of human effort. Then, the quality of the summaries is determined by the number of n-gram (sequence of n words) overlaps between the two summaries. ROUGE calculates considered in the evaluation are: ROUGE-N (n=1, 2, 3, 4).

ROUGE-L, a recall based measure, is measured by the number of n-gram overlaps (n = 1, 2, 3, 4) between the reference and system generated summaries. It is computed as follows:

$$= \frac{\sum_{S \in ReferenceSummaries} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in ReferenceSummaries} \sum_{gram_n \in S} Count(gram_n)} \quad (25)$$

, where n stands for the length of the n-gram, $gram_n$ and $Count_{match}(gram_n)$ is the maximum number of n-grams common to both summaries. When the several multiple references are used for evaluation, pairwise summary-level ROUGE-N score between the candidate summary "s" and every reference summary "r" is first computed.

Given M references, the best score over M sets of M-1 references is computed. The final score is the average of the M ROUGE-N scores using different M-1 references. This method is also known as Jacknifing procedure and, it is applied to all ROUGE measures in the ROUGE evaluation package for our evaluation process.

2. Experimental Results

To test the efficiency of our system, we have performed a preliminary set of experiments. We experimented our system with two most recent and well-known summarizers; TextRank [44] (Baseline A) and Automatic Text

Summarizer [49] (Baseline B). We have used a corpus of 600 news contents from several different sources such as CNN, BBC, the New York Times and so on. In order to evaluate the system with ROUGE method, we provided two manually human extracted summaries for each contents Reference summaries, and Baseline A, Baseline B and our proposed system for three System summaries. Figure 16 shows the sample summary of 6 sentences from a document produced by our system. And the original text can be seen in Figure 14.

```
############################
GoPro stock drops 23% after announcing layoffs and disappointing Q4 guidance

########## SUMMARY ##########

GoPro  s stock dropped 23.34 percent to 10.87 Wednesday afternoon after the c
ompany announced its Q4 was worse than expected.

GoPro  s stock was already dropping as the company faced mounting pressure fr
om investors to sell more cameras and find new revenue streams.

GoPro went public in mid 2014 with a stock price of  35 and quickly shot up t
o  89 a share.

The stock was already trading at an all time low.

But since that high  the stock gradually dropped until the summer of 2015 whe
n it lost over half of its value over several weeks.

The stock briefly jumped to 11.20 before trading was halted.
```

Figure 16. Sample Summary of A Document produced by the Proposed System

We have experimented the effect of n-gram words using ROUGE evaluation by changing the parameter(s) of n value. Table 10 shows that our system outperforms than the baseline methods according to the F-score value. And then, the sum of all Recall, Precision and F-score gives the highest score among the compared systems for ROUGE-1 evaluation.

Table 10. Comparison of ROUGE-1 Evaluation of Summarization Methods

System	Recall	Precision	F-score
Baseline A	5.0500	6.0249	5.3705
Baseline B	4.5273	**6.3084**	5.3395
Our system	**5.0504**	5.9864	**5.9864**

The result of ROUGE-2 evaluation of our system returns the lowest score compared to baseline methods in Table 11, while the baseline B performs better than the first baseline, and proposed system according to it's F-score for ROUGE-3 evaluation in Table 12.

Table 11. Comparison of ROUGE-2 Evaluation for Automatic Summarization

System	Recall	Precision	F-score
Baseline A	**4.1113**	4.6723	4.3526
Baseline B	3.7204	**5.5362**	**4.3978**
Our system	4.0498	4.9395	4.1939

Table 12. Comparison of ROUGE-3 Evaluation of Summarization Methods

System	Recall	Precision	F-score
Baseline A	**3.9512**	4.3972	4.1135
Baseline B	3.4723	**5.2058**	**4.1147**
Our system	3.8652	4.7591	4.0110

Table 13 shows our system outperforms the rest of the baseline systems in ROUGE-4 evaluation, we also measure the quality of the summaries generated with respect to various quality in mean coverage, and it surpasses the baseline methods in Table 14.

Table 13. Comparison of ROUGE-4 Evaluation of Summarization Methods

System	Recall	Precision	F-score
Baseline A	3.7234	4.2068	3.9530
Baseline B	3.3315	**5.0298**	3.9564
Our system	**3.7578**	4.6562	**3.9584**

The proposed method selects 7 most representative sentences as summaries.

To give a fair comparison, we examines the average of Recall, Precision and F-score result for each system overall ROUGE 1 to ROUGE 4. Table 14 shows the performance comparison between 2 baseline methods and our proposed system according to their average score in all measures.

Table 14. Performance Comparison between Baseline methods and proposed system

Recall					
	ROUGE1	ROUGE2	ROUGE3	ROUGE4	Avg Scr
Baseline 1	5.0500	**4.1113**	**3.9512**	3.7234	**4.2090**
Baseline 2	4.5374	3.7204	3.4723	3.3315	3.7629
Our System	**5.0504**	4.0498	3.8652	**3.7578**	4.1808
Precision					
	ROUGE1	ROUGE2	ROUGE3	ROUGE4	Avg Scr
Baseline 1	6.0249	4.6723	4.3972	4.2068	4.8253
Baseline 2	**6.3084**	**5.5362**	5.2058	5.0298	5.5200
Our System	6.5864	4.93958	5.35916	5.6562	**5.6353**
F-score					
	ROUGE1	ROUGE2	ROUGE3	ROUGE4	Avg Scr
Baseline1	5.3705	4.3526	4.1135	3.9530	4.4474
Baseline2	5.3395	**4.3978**	**4.1147**	3.9564	4.4521
Our System	**5.9864**	4.1939	4.0110	**3.9584**	**4.5374**

In Recall measure, the average score of Baseline 1 outperforms Baseline 2 and proposed method, while the average score for Precision and F-score of our proposed system surpasses the other approaches.

We compared the performance and efficiency of each system according to the size training set corpus. Using the average score of each system produced by the ROUGE evaluation method, we can see that the performance of the proposed system increased depending on the size of the training corpus. It gets better and, produced more accurate improved results as the size of training set is increased [Figure 17].

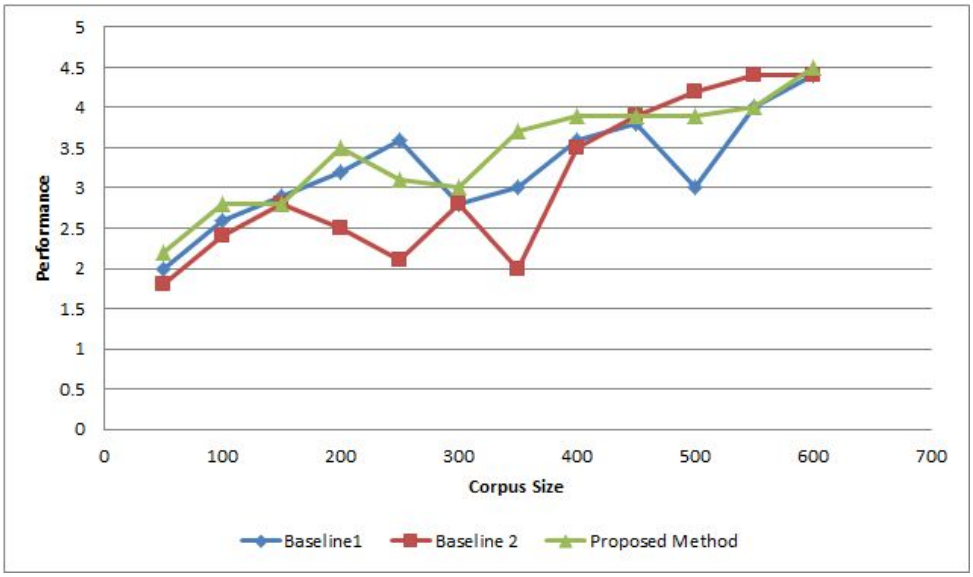

Figure 17. Performance Comparison of each system according to the size of training corpus

V. CONCLUSION AND FUTURE WORK

In this thesis, we presented a method to compute the probability distributions of keywords of the text according to their characteristics, and a method to construct lexical chains as an efficient intermediate representation of the document based on extracted candidate keywords. Along with normal WordNet relations, our method also included additional relations such as proper noun repetition and gloss relations in the computation of lexical chains.

Then, we investigated the approaches to extract sentences from the document(s) based on the distribution of lexical chains. We built a transition probability distribution generator (TPDG) for n-gram keywords which learns the characteristics of the assigned keywords from the training data set. A new method of automatic keyword extraction also featured in the system based on the Markov chains process. Among the extracted n-gram keywords, only unigrams are selected to construct the lexical chain. Instead of finding all the noun phrases from the text to build a lexical chain, collecting the most related keywords of the text gives the better performance and more efficient result. Terms involved in title, and related to title are vital for scoring and identifying the competent information within the text.

Lexical chains, until now, were mainly used to generate single document summaries. Lexical chains support to identify the themes, by clustering the document collection. We performed intrinsic evaluation to distinguish the quality of the summaries generated by our approaches. We found that our system achieved better results in keyword extraction than existing RAKE, and returns more efficient summaries than baseline method and recent systems.

Our method to compute lexical chains includes the gloss relations using the advent of the lexical database. These relations were based on the presence

of gloss concept or synonym of the gloss concept in the text. We would like to pursue further research into the methods to compute the semantic similarity based on the overlap of the concepts of gloss. The lexical chains are evaluated according to their performance in identifying the sense of the word in given context. It has been proved that concepts present in the gloss of a word play an important role in the determination of the word sense [41] [42]. We would like to compare the performance of our system in this aspect with respect to other lexical chaining methods, and our future study will be emphasized towards automatic abstractive summarization while improving the extraction of keywords of the text corpus independently.

REFERENCES

[1] Radev, D. R., Hovy, E., McKeown, K., "*Introduction to the special issue on summarization*". Computational Linguistics, vol. 28, no. 4, pages 399-408, 2002.

[2] Luhn, H.P., "*The automatic creation of literature abstracts*". IBM Journal of Research Development, vol. 2, no. 2, pages 159-165, 1958.

[3] Baxendale, P., "*Machine-made index for technical literature - an experiment*". IBM Journal of Research Development, vol. 2, no. 4, pages 354-361, 1958.

[4] Edmundson, H. P., "*New methods in automatic extracting*". Journal of the ACM, vol. 16, no. 2, pages 264-285, 1969.

[5] Kupiec, J., Pedersen, J., and Chen, F., "*A trainable document summarizer*". In Proceedings SIGIR '95, pages 68-73, New York, NY, USA, 1995.

[6] Aone, C., Okurowski, M. E., Gorlinsky, J., and Larsen, B., "*A trainable summarizer with knowledge acquired from robust nlp techniques*". In Mani, I.and Maybury, M. T., editors, Advances in Automatic Text Summarization, MIT Press, pages 71-80. 1999.

[7] Miller, G. A., "*Wordnet: a lexical database for english*". Commun. ACM, vol. 38, no. 11, pages 39-41, 1995.

[8] Conroy, J. M. and O'leary, D. P., "*Text summarization via hidden markov models*". In Proceedings of SIGIR '01, 406-407, New York, NY, USA, 2001.

[9] Nenkova, A., "*Automatic text summarization of newswire*", Lessons learned from the document understanding conference. In Proceedings of AAAI 2005, Pittsburgh, USA, 2005.

[10] Svore, K., Vanderwende, L., and Burges, C, "*Enhancing single-document summarization by combining RankNet and third-party sources*". In Proceedings of the EMNLP-CoNLL, pages 48-457, 2007.

[11] Luhn, H., "*The automatic creation of literature abstracts*". IBM Journal of Research and Development, vol. 2, no. 2, 1958.

[12] Barzilay, R. and M. Elhadad, "*Using lexical chains for text summarization*". In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th European Chapter Meeting of the Association for Computational Linguistics, Workshop on Intelligent Scalable Text Summarization, pages 10-17, Madrid, 1997.

[13] Halliday, M. and R. Hasan, "*Cohesion in English*". Longman, London, 1976.

[14] Mani, I., "*Automatic Summarization*". John Benjamins Co, Amsterdam, Philadelphia, 2001.

[15] Lesk, M.. "*Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone*". In Proceedings of the5th annual international conference on Systems documentation, pages 24-26,Toronto, Ontario, Canada. ACM Press, 1986.

[16] Banerjee, S. and T. Pedersen, "*Extended gloss overlaps as a measure of semantic relatedness*". In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pages 805-810, Acapulco, Mexico, 2003.

[17] Harabagiu, S. and D. Moldovan, "*WordNet: An Electronic Lexical Database, chapter Knowledge Processing on an Extended WordNet*". MIT press, 1998.

[18] A. Hulth. "*Improved automatic keyword extraction given more linguistic knowledge*". In Proceedings of the 2003 Conference on Emprical Methods in Natural Language Processing, Sapporo, Japan, 2003.

[19] Michael J. Giarlo, "*A comparative analysis of keyword extraction techniques*". Rutgers, The State University of New Jersey.

[20] Cohen J. D.,. "*Highlights: Language and Domain-independent Automatic Indexing Terms for Abstracting*". Journal of the American Society for Information Science, 1995, vol. 46, no. 3, pages 162-174, January 1999.

[21] Luhn H. P., "*A Statistical Approach to Mechanized Encoding and Searching of Literary Information*". IBM Journal of Research and Development, vol. 1, no. 4, pages 309-317, 1957.

[22] Salton G, Yang C. S., Yu C. T., "*A Theory of Term Importance in Automatic Text Analysis*". Journal of the American society for Information

Science, vol. 26, no. 1, pages 33-44, 1975.

[23] Matsuo Y., Ishizuka M., "*Keyword Extraction from a Single Document Using Word Co-occurrence Statistical Information*". International Journal on Artificial Intelligence Tools, vol. 13, no. 1, pages 157-169, 2004.

[24] Chien L. F., "*PAT-tree-based Keyword Extraction for Chinese Information Retrieval*". In Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1997), Philadelphia, PA, USA, pages 50-59, 1997.

[25] Ercan G, Cicekli I., "*Using Lexical Chains for Keyword Extraction. Information Processing and Management*". vol. 43, no. 6, pages 1705-1714, 2007.

[26] Hulth A., "*Improved Automatic Keyword Extraction Given More Linguistic Knowledge*". In Proceedings of the 2003 Conference on Emprical Methods in Natural Language Processing, Sapporo, Japan, pages 216-223, 2003.

[27] Dennis S. F., "*The Design and Testing of a Fully Automatic Indexing-searching System for Documents Consisting of Expository Text*". In G. Schecter eds. Information Retrieval: a Critical Review, Washington D. C.: Thompson Book Company, pages 67-94. 1967.

[28] Salton G, Buckley C.. "*Automatic Text Structuring and Retrieval - Experiments in Automatic Encyclopaedia Searching*". In Proceedings of the Fourteenth SIGIR Conference, New York: ACM, pages 21-30, 1991.

[29] Frank E., Paynter G. W., Witten I. H., "*Domain-Specific Keyphrase Extraction*". In Proceedings of the 16th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, Morgan Kaufmann, pages 668-673, 1999.

[30] Zhang K., Xu H., Tang J., Li J. Z., "*Keyword Extraction Using Support Vector Machine*". In Proceedings of the Seventh International Conference on Web-Age Information Management (WAIM2006), Hong Kong, China, pages 85-96, 2006.

[31] Keith H. J. B., "*Phraserate: An Html Keyphrase Extractor*". Technical Report, University of California, Riverside, vol. 1, no. 16, 2002.

[32] Plas L., Pallotta V., Rajman M., Ghorbel H., "*Automatic keyword extraction*

from spoken text, A comparison of two lexical resources: the EDR and WordNet". Proceedings of the 4th International Language Resources and Evaluation, European Language Resource Association, 2004.

[33] Hulth A., "*Improved automatic keyword extraction given more linguistic knowledge".* In Proceedings of the 2003 Conference on Emprical Methods in Natural Language Processing, Sapporo, Japan, 2003.

[34] Witten I., Paynte G., Frank E., Gutwin C., Nevill-Manning C., "*KEA: practical automatic key phrase extraction".* In Proceedings of the 4th ACM Conference on Digital Library, 1999.

[35] Suzuki Y., Fukumoto F., Sekiguchi Y., "*Keyword extraction of radio news using term weighting with an encyclopedia and newspaper articles",* SIGIR, 1998.

[36] Keith J. B., Humphreys. "*Phrase rate: An HTML keyphrase extractor".* Technical Report. 2002.

[37] Porter M. F., "*An Algorithm for Suffix Stripping".* Program, vol. 14, no. 3, pages 130-137, 1980.

[38] Ms. Anjali G. J., "*A Comparative Study of Stemming Algorithms".* Anjali Ganesh Jivani et al, Int. J. Comp. Tech. Appl., vol. 2, no. 6, ISSN:2229-6093, 1930-1938.

[39] Mani I., Maybury M., "*Advances in Automatic Text Summarization".* MIT Press, 1999.

[40] Lin C.-Y., "*Rouge: A package for automatic evaluation of summaries".* In Marie-Francine Moens, S. S., editor, Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, pages 74-81, Barcelona, Spain, 2004.

[41] Lesk M., "*Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone".* In Proceedings of the5th annual international conference on Systems documentation, pages 24-26, Toronto, Ontario, Canada. ACM Press, 1986.

[42] Banerjee S., Pedersen T., "*Extended gloss overlaps as a measure of semantic relatedness".* In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pages 805-810, Acapulco, Mexico, 2003.

[43] Feifan L., Deana P., Fei L., Yang L., "*Unsupervised Approaches for Automatic Keyword Extraction Using Meeting Transcripts*". Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL, pages 620 - 628, Boulder, Colorado, June 2009.

[44] Rada M., Paul T., "*TextRank: Bringing Order into Texts*". In Proceedings of the Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, Barcelona, Spain, July 2004.

[45] Martin D., Karel J., "*Automatic Keyphrase Extraction based on NLP and Statistical Methods*". In Proceedings of the Dateso 2011: Annual International Workshop on Databases, Texts, Specifications and Objects, Pisek, Czech Republic, April 20, 2011.

[46] Rose, S., Engel, D., Cramer, N., & Cowley, W.. "*Automatic Keyword Extraction from Individual Documents*". In M. W. Berry & J. Kogan (Eds.), Text Mining: Theory and Applications: John Wiley & Sons, 2010.

[47] Chengzhi Z., Huilin W., Yao L., Dan W., Yi L., Bo W., "*Automatic Keyword Extraction from Documents Using Conditional Random Fields*". Journal of Computational Information Systems, vol. 4, no. 3, pages 1169-1180, 2004.

[48] Zhang K., Xu H., Tang J., Li J. Z.. "*Keyword Extraction Using Support Vector Machine*". In Proceedings of the Seventh International Conference on Web-Age Information Management (WAIM2006), pages 85-96, Hong Kong, China, pages 85-96, 2006.

[49] Annapurna P. Patil, S. D., Syed A. A. A., Tanay A., Varun B., "*Automatic text summarizer*". In proceedings of 2014 International Conference on Advances in Computing, Communications and Informatics(ICACCI), pages 1530-1534, 2014.

[50] Dipanjan D., Andre F. T. M., "*A Survey on Automatic Text Summarization*". Technical Report, 2007.

[51] Charles M. G., Laurie S. J., "*Introduction to Probability*". pages 40-470, 1997.

[52] Fang J., Guogh L,. Xue Y., Xiang Y., "*Semantic-based Keyword Extraction Method for Document*". In proceedings of International Journal of u- and

e-Service, Sclence and Technology, vol. 8, no. 5, pages 37-46, 2015.

[53] Luis M., Wang L., Isabel T., Chris D., Alan W. B., Anatole G., David M. M., Joao P. N., Jaime C., "*Automatic Keyword Extraction on Twitter*". In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers), Beijing, China, pages 637-643, 2015.

[54] Xiong S., Luo Y., "*A new approach for multi-document summarization based on latent semantic analysis*". In Proceedings of the Seventh International Symposium on Computational Intelligence and Design, ISCID 14, vol. 1, pages 177-180, Hangzhou, China, December, 2014.

[55] Yang G., Wen D., Kinsuhuk, Chen N. S., Sutinen E., "*A novel contextual topic model for multi-document summarization*". Expert Systems with Applications, vol. 42, no. 3, pages 1340-1352, 2015.

[56] Meena Y. K., Gopalani D., "*Domain independent framework for automatic text summarization*". Procedia Computer Science, vol. 48, pages 722-727, 2015.

[57] Steven B., Ewan K., Edward L., "Analyzing Text with the Natural Language Toolkit: Natural Language Processing with Python". 2009.

[58] https://en.wikipedia.org/wiki/Tf-idf.

[59] https://en.wikipedia.org/wiki/Part-of-speech_tagging.

ACKNOWLEDGMENTS

I, paying homage to the supreme triple Gems, the Buddha, Dhamma and Sangha, and give great respects to my parents and my teachers. Creating a Master Thesis is not an individual experience; rather it takes place in a social context and includes several persons, whom I would like to thank sincerely.

First and foremost, this thesis is dedicate--d to my parents and, I would like to thank for their endless love, support and encouragement. Thank you both for giving me strength to reach for the stars and chase my dreams. My little brother, cousins, friends, relatives and beloved ones deserve my wholehearted thanks as well. There are many people without whose support and contributions, this thesis would not be possible.

I would like to sincerely indicate my gratitude on my supervisor, Prof. Kim Pankoo, for his guidance and support throughout this study, and especially for his confidence in me. I would also like to thank my seniors, Choi Junho, Choi Chang, Choi Dongjin, Ko ByeongKyu, Kim JeongIn, Lee Eunji and my lab mates Lee Jaeuk, Cha Junseok, Han SungMin, Lee Mungyu and Hong Taekeun, who offered me not only the environment of working in an energetic laboratory but the joyful student life outside of the campus as well.

Last but not least, I would like to thank my friends and colleagues at Chosun University for their encouragement and moral support which made my stay in studies in Gwangju more enjoyable. Everybody kept me going, and this thesis would not have been possible without them.