Keyword Selection from Textual Data using Table based K Nearest Neighbor

Taeho Jo
President
Alpha AI Publication
Cheongju, South Korea
tjo018@naver.com

***Abstract*—This article proposes the modified KNN (K Nearest Neighbor) algorithm which receives a table as its input data and is applied to the keyword extraction. The table based algorithms worked successfully in text mining tasks such as text categorization and text clustering in previous works, and the keyword extraction is able to be mapped into the binary classification where each word is classified into keyword or non-keyword. In the proposed system, a text which is given as the input is indexed into a list of words, each word is classified by the proposed KNN version, and the words which are classified into keyword are extracted ad the output. The proposed KNN version is empirically validated as the better approach in deciding whether each word is a keyword or non-keyword in news articles and opinions. In using the table based KNN algorithm, it is easier to trace results from categorizing words.**

***Keywords*-Keyword Extraction, Table Similarity, Table based KNN**

I. Introduction

Keyword extraction refers to the process of selecting essential words reflecting the contents from an article. In this research, the task may be viewed into the binary classification task where each word is classified into a keyword or a non-keyword. Statistical and posting features are considered as features for deciding whether each word is a keyword or not. Under the view, it is assumed that a supervised learning algorithm is applied as the approach. Although the fuzzy keyword extraction where a continuous value between zero and one is assigned to a word as its membership value is available, the scope of this research is restricted to only crisp keyword extraction where each word is classified exclusively to a keyword or a non-keyword.

Encoding words into numerical vectors for using the traditional machine learning algorithms causes some problems. When using texts as features of numerical vectors representing words, many features are required for maintaining the robustness; their dimensions usually reach several hundred in spite of using feature selection schemes [2]. Because almost numerical vectors have zero values dominantly, discrimination among them for computing their distances is very poor [25]. If we set grammatical properties as features to avoid the huge dimensionality, it becomes very difficult to implement the encoding process. Therefore, this research is intended to overcome the above problems by encoding words into tables, instead of numerical vectors.

Let us consider what this research proposes as the solutions to the above problems. In this research, we encode each word into a table of entries of texts including it and its weights of them. We modify the KNN (K Nearest Neighbor) into its table based version where a table is directly given as the input data and the similarity between tables is computed as a normalized value between zero and one. We apply the modified version to the classification task which is mapped from the keyword extraction. As the difference between the word categorization and the keyword extraction, the former is to classify words by topics, whereas the latter is to classify them by their relevancy to the given text.

Let us mention the benefits which are expected from this research. The tables which represent texts may be characterized as more compact and symbolic representations than numerical vectors. There is more discrimination among tables than numerical vectors in computing their similarities because the sparse distribution is completely avoided in encoding texts into tables. We may expect both better performance and more stability from the proposed KNN version as the contribution of defining the normalized similarity measure. However, note that the table size is given as the additional parameter, and it should be optimized between the computation speed and the reliability.

This article consists of the five sections. In Section II, we explore the previous works which are relevant to this research. In Section III, we describe in detail what we propose in this research. In Section IV, we validate empirically the proposed approach by comparing it with the traditional one. In Section V, we mention the significance of this research and the remaining tasks as the conclusion.

II. Previous Works

This section is concerned with the previous works which are relevant to this research. In Section II-A, we explore the previous cases of applying the KNN algorithm to text mining tasks. In Section II-B, we survey the schemes of encoding texts or words into structured data. In Section II-C, we describe the previous machine learning algorithms which receive alternative structured data such as tables and string vectors to numerical vectors. Therefore, in this section, we

provide the history about this research, by surveying the relevant previous works.

A. Applications to Word Classification Tasks

This section is concerned with the previous works on applying the modern KNN version to the keyword extraction and its similar tasks. We mention the topic based word categorization and the index optimization as the tasks which are similar as the keyword extraction. The KNN algorithm was modified into the modern version for solving the problems in encoding words into numerical vectors. In previous works, the trials of applying the KNN version to the keyword extraction are available. This section is intended to survey the previous cases of modifying the KNN algorithm into its modernized version, and applying it to the classification tasks, including the keyword extraction.

Let us present the previous works on applying the modernized KNN algorithm to the topic based word categorization. The KNN version which computes both the feature similarity and the feature value similarity between a novice item and a training example was used for the word categorization [8]. The KNN algorithm which was modified for classifying a string vector directly, was applied to the same task [9]. The KNN algorithm was modified into the version which classified a graph directly as the approach to the word categorization [10]. The modernized KNN algorithm was used for the topic based word categorization where each word is classified into one among the predefined categories.

Let us mention the previous cases of applying the modernized versions of KNN algorithm to the task which is covered in this study. The KNN algorithm which computes the similarity between a training example and a novice one, considering the feature similarities was applied to keyword extraction [11]. The KNN algorithm which is modernized into the version which processes string vectors directly, was adopted for implementing a keyword extraction system [12]. Another type of modernized KNN algorithm which classifies a graph directly was considered as the approach to the keyword extraction [13]. The keyword extraction will be mapped into a binary classification, following the scheme in the above literatures.

Because the text summarization looks similar as the keyword extraction in selecting important ones, let us mention the previous cases of applying the modernized KNN algorithms to the text summarization. The KNN algorithm which considers the feature similarities was proposed as the approach to the text summarization [14]. There was the case of applying the KNN version which processes string vectors directly, instead of numerical vectors, to the text summarization [19]. The version which classifies a graph directly was adopted as the approach to the text summarization [20]. In the above works, the text summarization was mapped into the binary classification of paragraphs.

Let us mention some points which makes this research distinguished from the above literatures. We surveyed the previous cases of using the three kinds of modernized KNN algorithms for the keyword extraction and its related tasks. The word categorization was mentioned with the reason that the task is the source from which the keyword extraction is derived, and the text summarization was mentioned with the reason that each paragraph is classified depending on its importance in the given text. In this research, the KNN algorithm is modernized by making it process tables directly. The keyword extraction is mapped into the binary classification of words, following the schemes in the above literatures, and the proposed KNN algorithm will be applied to the task.

B. Word and Text Encoding

This section is concerned with the previous works on schemes on encoding texts or words. In previous works, some issues in encoding them into numerical vectors were pointed out. The previous works challenged against the issues by encoding them into non-numerical vectors. In this section, we mention the tables, the graphs, and the string vectors, as non-numerical vectors. This section is intended to survey the previous challenges by doing so.

Let us present the previous cases of encoding words or texts into tables. Words were encoded into tables in using the AHC algorithm for the word clustering [15]. Texts were encoded into tables in using the KNN algorithm for the text categorization [16] . Texts were encoded so in using the AHC algorithm for the text clustering [21]. In the above literatures, in using the KNN algorithm and the AHC algorithm, texts or words were encoded into tables.

Let us consider the previous cases of encoding words or texts into string vectors. Words were encoded into string vectors in using the AHC algorithm for clustering words [17]. Texts were encoded into string vectors in using the KNN algorithm for the text categorization [18]. They were also encoded so in using the AHC algorithm for clustering texts [22]. In the literatures, we present the cases of encoding raw data into string vectors in using the machine learning algorithms.

Let us explore the previous works on encoding words or texts into graphs. It was proposed that words were encoded into graphs in using the AHC algorithm for clustering words [7]. It was proposed that texts should be encoded into graphs in using the KNN algorithm for categorizing texts [23]. It was proposed that texts were encoded so, in using the AHC algorithm for clustering texts [24]. In the above literatures, we present the previous cases of encoding raw data into graphs.

We mentioned the three types of structured data, alternative to numerical vectors. We adopt the first type of structured data called tables as word representations. We define the similarity metric between tables and modify the KNN algorithm into the version which processes tables, directly.

We apply the modified KNN algorithm to implementing the keyword extraction system. We empirically validate the modified version compared with the traditional version in extracting keywords from a text.

C. Non-Numerical Vector based Machine Learning Algorithms

This section is concerned with the previous works on non-numerical vector based classification algorithms. In the previous section, we presented the cases of encoding words or texts in using the KNN algorithm and the AHC algorithm. In this section, we mention the string kernel based SVM, the table matching algorithm, and the Neural Text Categorizer, which were developed as the non-numerical vector based classifiers. We use them where texts are encoded into table or string vectors, as the approaches to the text classification. This section is intended to survey the previous works which are involved in applying them to the text classification.

The string kernel based algorithm is the trial of computing the similarity between raw data without encoding them. The string kernel was initially proposed for improving the SVM (Support Vector Machine) performance as the approach to the text categorization by Lodhi et al in 2002 [28]. The k means algorithm was modified based on the string kernel as the approach to the text clustering by Karatzoglou and Feinerer in 2006 [27]. The string kernel based SVM was used for classifying sentences by Kate and Mooney in 2006 [26]. The string kernel which is used in the above literatures is the lexical similarity metric between texts based on characters.

Let us explore the previous works on the table based matching algorithm. It was initially proposed as the approach to the text categorization by Jo and Cho in 2008 [25]. It was applied to the soft text categorization which allows more than one category to be assigned to each text by Jo in 2008 [3]. It was improved into the more robust and stable approach by Jo in 2015 [6]. In using the table based matching algorithm which was covered in the above literatures, texts should be encoded into tables.

Let us mention the Neural Text Categorizer which is the supervised neural network model which is specialized for the text categorization. It was initially proposed as the approach to the text categorization by Jo in 2008 [4]. Its better performance than those of the Naive Bayes and the SVM was confirmed in both the soft categorization and the hard categorization of texts in 2010 [5]. It was applied to the classification of Arabian texts by Abainia et al. in 2015 [1]. It was mentioned as one of dynamic neural networks by Vega and Mendez-Vasquez [29].

We mentioned the three classification algorithms as non-numerical vector based approaches to the text classification. The binary classification of words which is mapped from the keyword extraction is the problem we try to solve in this research. Each word is encoded into a table whose entry consists of a text identifier which includes itself and a weight value which indicates the relation between the text and the word. The KNN algorithm which is modified into the version which classifies tables directly as the approach to the keyword extraction. Its performance is validated in the keyword extractions, compared with the traditional version.

III. Proposed System

This section is concerned with the keyword extraction system where the table based KNN (K Nearest Neighbor) is adopted as the approach, and consists of the four sections. In Section III-A, we describe the process of encoding words into tables as the preprocessing step. In Section III-B, we cover the scheme of computing a similarity between two tables into a normalized value between zero and one. In Section III-C, we mention the proposed KNN where a table is given as the input data instead of a numerical vector. In Section III-D, we explain the system architecture of the keyword extraction where the proposed KNN is adopted..

A. Word Encoding

This section is concerned with the process of encoding words into tables. We mentioned the table based machine learning algorithms as the approaches to text mining tasks, in Section II-B and II-C. Words are encoded into tables with the three steps: corpus indexing, inverted indexing, and term weighting. A table which is a word representation consists of entries, each of which consists of text identifier and weight. This section is intended to describe the three steps which are presented in Figure 1-3.

The process of indexing a corpus into a list of words as the first step of encoding words is illustrated in Figure 1. Texts are partitioned into tokens by white spaces or punctuation marks in the tokenization. Each token is mapped into its root forms by stemming rules which are taken externally in the steaming. In the stop-word removal, grammatical words such as conjunctions or prepositions which are called stop-words, are removed. The list of words which belong to noun, verb, or adjective, is extracted finally from the process.

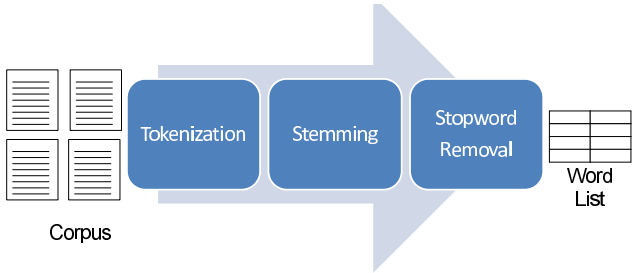

Figure 1. Overall Process of Word Indexing

The inverted index where each word is linked to its related texts is illustrated in Figure 2. The structure where each text is linked to its related words is actually constructed

after indexing the corpus by the process which is shown in Figure 1. It is converted into one which is presented in Figure 2, in this step. Additional information, such as positing information, grammatical one, and statistical one, is assigned to each text in upgrading the inverted index. A list of texts is assigned to each word as the information about it for encoding it into a table.

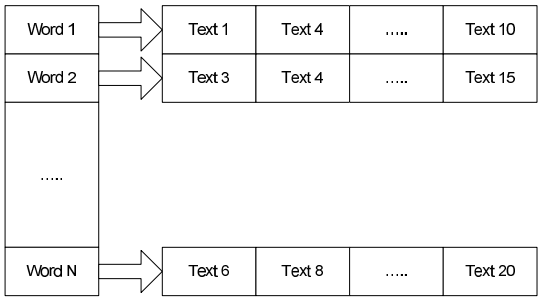

Figure 2. Inverted Index

The process of assigning a weight to each text in the table which represents a word is illustrated in Figure 3. In the previous step, the inverted index which is the basis for doing this step was constructed as shown in Figure 2. The TF-IDF weight is computed by the equation which is presented in Figure 3, for each entry. It is proportional to the frequency of the word in a text, but inversely proportional to the number of documents which include it in the corpus. The corpus is required for computing the TF-IDF weight.

We presented the three steps which are involved in encoding a word into a table in Figure 1-3. If constant weights are assigned to all entries, the table is viewed as an unordered set of text identifiers. Each entry consists of a text identifier and its TF-IDF weight to the word in the table which represents a word. Each entry may be expanded into one with a text identifier and its multiple weights by adopting multiple weight schemes. We need to define the operations on tables for modifying the machine learning algorithms into versions which process them directly.

B. Similarity Metric

This section is concerned with the similarity metric between two tables. In the previous section, we studied the process of encoding words into tables. We need to define the similarity metric between two tables for modifying the KNN algorithm as the approach to the keyword extraction into the version which processes tables directly. We view a table as set of entries each of which consists of a text identifier and its weight and compute the similarity based on the text identifiers which are shared by the two tables. This section is intended to describe the process of computing the similarity between two tables.

Let us mention the function of a table for mapping a table into an item set. A table is expressed as a set of entries as

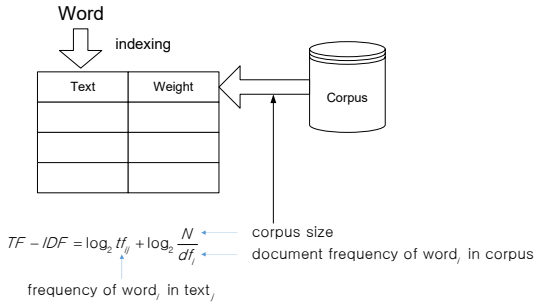

Figure 3. Text Weighting

shown in equation (1),

$$T = \{(text\_id_1, weight_1), (text\_id_2, weight_2), \ldots, (text\_id_{|T|}, weight_{|T|})\} \quad (1)$$

where $text\_id_i$ is a text identifier which include the word and $weight_i$ is its weight in the text identified by $text\_id_i$. The table function is defined for generating a list of text identifiers as expressed in equation (2),

$$F(T) = \{text\_id_1, text\_id_2, \ldots, text\_id_{|T|}\} \quad (2)$$

The elements in the set, $F(T)$, is given text identifiers which include the word which is represented by the table, T. The function will be used for computing the similarity between two tables.

Let us mention the process of computing the similarity between two tables which represent words. The two tables are expressed as two sets of entries in equation (3) and (4),

$$T_1 = \{(text\_id_{11}, weight_{11}), (text\_id_{12}, weight_{12}), \ldots, (text\_id_{1|T|}, weight_{1|T|})\} \quad (3)$$

$$T_2 = \{(text\_id_{21}, weight_{21}), (text\_id_{22}, weight_{22}), \ldots, (text\_id_{2|T|}, weight_{2|T|})\} \quad (4)$$

The two tables are mapped into the sets of text identifiers which are shown in equation (5) and (6), by applying the table function to equation (3) and (4),

$$F(T_1) = \{text\_id_{11}, text\_id_{12}, \ldots, text\_id_{1|T|}\} \quad (5)$$

$$F(T_2) = \{text\_id_{21}, text\_id_{22}, \ldots, text\_id_{2|T|}\} \quad (6)$$

The set of shared text identifiers which is shown in equation (7) is obtained by applying the intersection to equation (5) and (6),

$$F(T_1) \cap F(T_2) = \{stext\_id_1, stext\_id_2, \ldots, stext\_id_k\} \quad (7)$$

The shared table is constructed by taking their weights from the two tables, T_1 and T_2 as shown in equation (8),

$$ST = \{(stext\_id_1, weight_{11}, weight_{21}), \ldots, (stext\_id_k, weight_{1k}, weight_{2k})\} \quad (8)$$

In equation (8), $weight_{1i}$ indicates the weight from the table, T_1, and $weight_{2i}$ indicates the weight from the table, T_2 to the text identifier, $stext\_id_1$.

Let us mention the process of computing the similarity between two tables after extracting the shared entries. The weights of the two tables are given as sums of entry weights, as expressed in equation (9) and (10),

$$W(T_1) = \sum_{i=1}^{|T_1|} weight_{1i} \quad (9)$$

$$W(T_2) = \sum_{i=1}^{|T_2|} weight_{2i} \quad (10)$$

The dual weight sums in the shared table, ST, are defined as equation (11) and (12),

$$W_1(ST) = \sum_{i=1}^{k} sweight_{1i} \quad (11)$$

$$W_2(ST) = \sum_{i=1}^{k} weight_{2i} \quad (12)$$

The similarity between the tables, T_1 and T_2 is computed by equation (13),

$$sim(T_1, T_2) = \frac{W_1(ST) + W_2(ST)}{W(T_1) + W(T_2)} \quad (13)$$

The similarity between tables is always given as normalized value between zero and one.

Above, we mentioned the similarity between two tables as a normalized value between zero and one. If the two tables are identical to each other as shown in equation (14),

$$T_1 = T_2 \quad (14)$$

the similarity between them is 1.0, as shown in equation (15),

$$\begin{aligned} sim(T_1, T_2) &= \frac{W_1(ST) + W_2(ST)}{W(T_1) + W(T_2)} \\ &= \frac{2W_1(ST))}{2W(T_1)} = \frac{2W_1(T_1))}{2W(T_1)} = 1.0 \end{aligned} \quad (15)$$

If the two tables are completely different from each other as shown in equation (16),

$$F(T_1) \cap F(T_2) = \oslash, |ST| = 0 \quad (16)$$

the similarity between them is zero, as shown in equation (17),

$$\begin{aligned} sim(T_1, T_2) &= \frac{W_1(ST) + W_2(ST)}{W(T_1) + W(T_2)} \\ &= \frac{0}{W(T_1) + W(T_2)} = 0.0 \end{aligned} \quad (17)$$

The similarity between two tables is given as a normalized value between zero and one, as shown in equation (18),

$$\begin{aligned} ST \subseteq T_1, ST \subseteq T_2 \\ W_1(ST) + W_2(ST) \leq W(T_1) + W(T_2) \end{aligned} \quad (18)$$

The similarity threshold is set between zero and one in modifying machine learning algorithms using the operation.

C. Proposed Version of KNN

This section is concerned with the proposed version of KNN algorithm which is shown in Figure 4, as the approach to the keyword extraction. We described the process of encoding words into tables in section III-A, and assume that the training examples and a novice one are given as tables. The similarity metric which is described in Section III-B is used for selecting nearest neighbors from training examples. Variants may be derived by considering more selection schemes and more voting schemes, in addition. This section is intended to describe the proposed version of KNN which classifies directly tables and its variants.

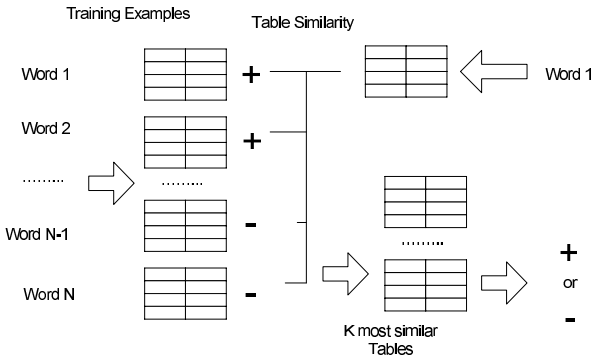

Figure 4. The Proposed Version of KNN

Let us mention the step where the nearest neighbors are selected as the references for classifying items. The training

words and a novice word are mapped into tables as shown in Figure 4, by the process which was covered in Section III-A. The similarities of the novice one with the training ones are computed by the equation (13). The training examples are ranked by their similarities and the k most similar ones are selected as the nearest neighbors. The rank based scheme is adopted in selecting the nearest neighbors in the KNN algorithm.

Let us mention the process of voting the labels of the nearest neighbors for deciding one of a novice item. We notate the set of nearest neighbors of the novice item, T , whose elements are given as tables and their target labels, by equation (19),

$$Ne_k(T) = \{(T_1, y_1), (T_2, y_2), \ldots, (T_k, y_k)\}, \quad y_i \in \{c_1, c_2, \ldots, c_m\} \tag{19}$$

where $c_1, c_2, \ldots, c_m$ are the predefined categories and k is the number of nearest neighbors. The number of the nearest neighbors which are labeled with the category,c_i is notated by $Count(Ne_k(T), c_i)$. The label of the novice item, T, is decided by the majority of categories in the nearest neighbors, as expressed by equation (20),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}}\, Count(Ne_k(T), c_i) \tag{20}$$

The external parameter,k, is usually set as an odd number for avoiding the possibility of largest number of nearest neighbors to more than one category.

Let us mention the weighted voting of labels of nearest neighbors as the alternative scheme to the above. Assuming that the similarity between two tables as a normalized value between zero and one, and we may use the similarities with the nearest neighbors, $sim(T, T_1), sim(T, T_2), \ldots, sim(T, T_k)$ as weights, $w_1, w_2, \ldots, w_k$ by equation (21),

$$w_i = sim(T, T_i) \tag{21}$$

indicates the similarity of a novice table with the ith nearest neighbor. The total weight of nearest neighbors which labeled with the category, c_i by equation (22),

$$Weight(Ne_k(T), c_i) = \sum_{T_j \in c_i}^{k} w_j \tag{22}$$

The label of the novice item, T, is decided by the category which corresponds to the maximum sum of weights as shown in equation (23),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}}\, Weight(Ne_k(T), c_i) \tag{23}$$

When the weights of nearest neighbors are set constantly, equation (23) is same to equation (20), as expressed in equation (24),

$$Weight(Ne_k(T), c_i) = Count(Ne_k(T), c_i) \tag{24}$$

We described the proposed version of the KNN algorithm in this section. In using the proposed KNN algorithm, raw data is encoded into tables, instead of numerical vectors. The similarities of a novice item with the training examples are computed by the similarity metric which is defined in Section III-B. The rank based selection is adopted as the scheme of selecting nearest neighbors among training examples. Because we are interested in the comparison of the traditional version and the proposed version as the ultimate goal, we use the unweighted voting in the experiments which are covered in Section IV.

D. Keyword Extraction System

This section is concerned with the keyword extraction system which adopts the table based KNN algorithm. In Section III-C, we described the proposed version of KNN algorithm as the approach to the keyword extraction. The task is mapped into the binary classification of each word into keyword or non-keyword. In this system, a text is indexed into words and ones which are classified into keywords, are extracted. This section is intended to describe the keyword extraction system with respect to its functions and architecture.

The process of collecting sample words domain by domain for implementing the keyword extraction system, is illustrated in Figure 5. The keyword extraction is interpreted into the binary classification where each word is classified into keyword or non-keyword in this study. The topic based word classification belongs to the domain independent classification where same word is classified identically with regardless of domain, whereas the keyword extraction belong to the domain dependent classification where same word may be classified differently depending on the domain. The words which are labeled keyword or non-keyword are collected domain by domain. The domain of the input text should be presented before executing the keyword extraction.

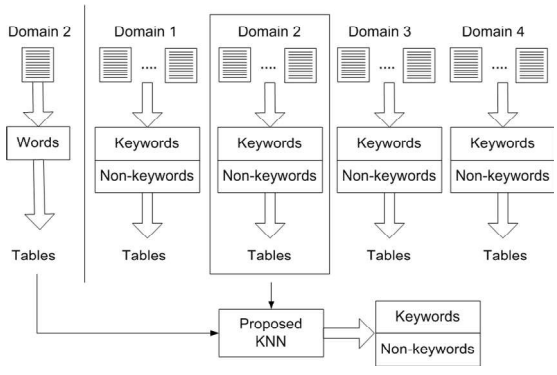

Figure 5. Sample Words

The entire architecture of the proposed keyword extraction system is illustrated in Figure 6. A text is given as the input, and words are extracted from it in the indexing module.

The sample words in the keyword group and the non-keyword group and ones which are indexed from the text are mapped into tables in the encoding module. The words which indexed from the text are classified into one of the two categories in the similarity computation module and the voting module. The system consists of the four modules: the indexing module, the encoding module, the similarity computation module, and the voting module.

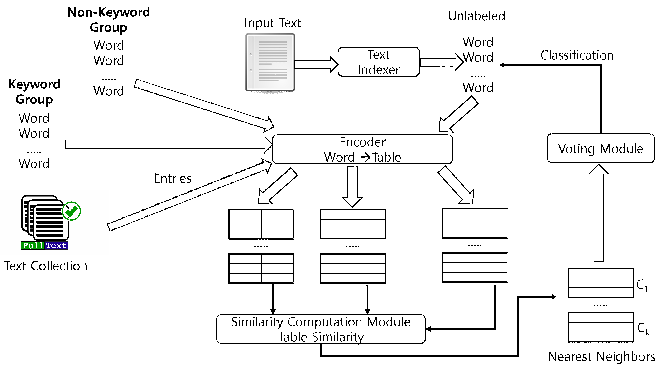

Figure 6. Proposed System Architecture

The execution process of the proposed system is illustrated as the block diagram in Figure 7. The sample words which are labeled with keyword or non-keyword are collected from each domain, and encoded into tables. The input text is indexed into a list of words and they are also encoded into tables. The nearest neighbors are selected by computing its similarities with the sample words and its label is decided by voting ones of its nearest neighbors for each word. The words which are classified into keyword are extracted as the final output.

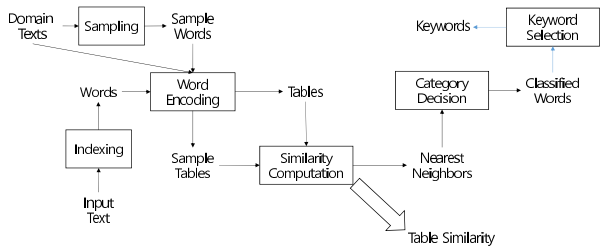

Figure 7. Execution Process of Proposed System

Let us make some remarks on the proposed system which is illustrated in Figure 6 as the architecture. The keyword extraction is defined as the binary classification where each word is classified into keyword or non-keyword. Each word is encoded into a table instead of a numerical vector and a table is classified directly. Among words included in the input text, ones which are classified into keyword are selected as keywords of the input text. We need to add the text classification module, in order to predict the domain of the input text.

IV. Experiments

This section is concerned with the empirical experiments for validating the proposed version of KNN, and consists of the five sections. In Section IV-A, we present the results from applying the proposed version of KNN to the keyword extraction on the collection, NewsPage.com. In Section IV-B, we show the results from applying it for classifying words into keyword or not, from the collection, Opinosis. In Section IV-C and IV-D, we mention the results from comparing the two versions of KNN with each other in the task of keyword extraction from 20NewsGroups.

A. NewsPage.com

This section is concerned with the experiments for validating the better performance of the proposed version on the collection: NewsPage.com. We interpret the keyword extraction into the binary classification where each word is classified into keyword or non-keyword, and gather words which are labeled with one of the two categories, from the collection, topic by topic. Each word is allowed to be classified into one of the two labels, exclusively. We fix the input size as 50 dimensions of numerical vectors and use the accuracy as the evaluation measure. Therefore, this section is intended to observe the performance of the both versions of KNN in the four different domains.

In Table I, we specify NewsPage.com which is used as the source for extracting the classified words, in this set of experiments. The text collection was used for evaluating approaches to text categorization, in previous works [6]. In each topic, we extracted 125 words labeled with keyword, and 125 words labeled with non-keyword. The set of 250 words in each topic is partitioned into the 200 words as training ones and the 50 words as the test ones, keeping the complete balanced distribution over the two labels, as shown in Table I. In building the test collection of words, we decide whether each word is a keyword or not, depending on its frequency concentrated in the given category combining with the subjectivity, in scanning articles.

Table I
The Number of Texts and Words in NewsPage.com

Category	#Texts	#Training Words	#Test Words
Business	500	200 (100+100)	50 (25+25)
Health	500	200 (100+100)	50 (25+25)
Internet	500	200 (100+100)	50 (25+25)
Sports	500	200 (100+100)	50 (25+25)

Let us mention the experimental process of validating empirically the proposed approach to the task of keyword extraction. We collect sample words which are labeled with keyword or non-keyword in each of the four domains: Business, Sports, Internet, and Health, depending on subjectivities and concentrated frequencies of words, and encode them into numerical vectors and tables. In each domain, for each of the 50 test examples, the KNN computes its similarities with the 200 training examples, and select the three most similar training examples as its nearest neighbors. Independently, we perform the four experiments each of which classifies each word into keyword or non-keyword by the two versions of KNN algorithm. For evaluating the both versions of KNN in the classification which is mapped from the keyword extraction, we compute the classification accuracy by dividing the number of correctly classified test examples by the number of test examples.

In Figure 8, we illustrate the experimental results from decoding whether each word is a keyword, or not, using the both versions of KNN algorithm. The y axis indicates the accuracy which is the rate of the correctly classified examples in the test set. Each group in the x-axis indicates the domain within which the keyword extraction which is viewed into a binary classification is performed, independently. In each group, the gray bar and the black bar indicate the performance of the traditional version and the proposed version of KNN algorithm, respectively. The most right group in Figure 8 indicates the average over accuracies over the left four groups, and set the input size which is the dimension of numerical vectors as 50.

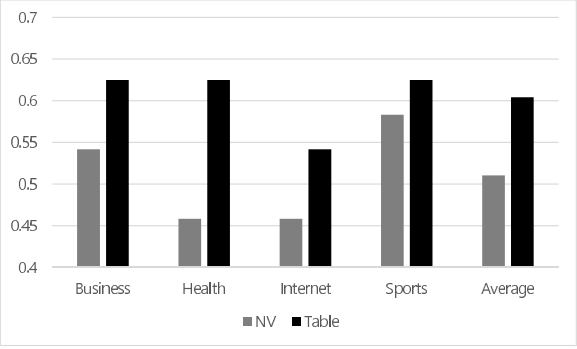

Figure 8. Results from Recognizing Keywords in Text Collection: NewsPage.com

Let us make the discussions on the results from doing the keyword extraction using the both versions of KNN algorithm, as shown in Figure 8. The accuracy which is the performance measure of the classified task is in the range between 0.45 and 0.62. The proposed version of the KNN algorithm works better in the three domains: Health, Internet and Business. It is lost slightly in the domain, Sports. However, from this set of experiments, we conclude the proposed version works better than the traditional one, in averaging over the four cases.

B. Opinopsis

This section is concerned with the set of experiments for validating the better performance of the proposed version on the collection, Opinosis. We view the keyword extraction into the binary classification where each word is classified into keyword or non-keyword, and collect words which are classified by their frequencies and subjectivities from Opniopsis. In each topic, each word is exclusively classified into one of keyword and non-keyword. We fix the input size to 50, and use the accuracy as the evaluation measure. In this section, we observe the performances of the both versions of KNN algorithm in the three different domains.

In Table II, we illustrate the text collection, Opinosis, which is used as the source for extracting the classified words in this set of experiments. The collection was used in previous works, for evaluating the approaches to text categorization. We extract the 250 words from each topic; the half of them is labeled with 'keyword'. In each topic, the 250 words is partitioned into the 200 words as the training set and the 50 words as the test set, keeping the completely balanced distribution over the two labels, as shown in Table II. In building the collection of labeled words, we label them into 'keyword' or 'non-keyword' by combining their frequencies which are concentrated in their own category with the subjectivity in scanning individual articles.

Table II
THE NUMBER OF TEXTS AND WORDS IN OPINIOPSIS

Category	#Texts	#Training Words	#Test Words
Car	23	200 (100+100)	50 (25+25)
Electronic	16	200 (100+100)	50 (25+25)
Hotel	12	200 (100+100)	50 (25+25)

We perform this set of experiments by the process which is described in Section IV-A. We collect sample words which are labeled with 'keyword' and 'non-keyword' in each of the three domains: 'Car', 'Electronics', and 'Hotel', and we encode them into 50 sized numerical vectors and tables. For each test example, the both versions of KNN computes its similarities with the 200 training examples and select the three most similar training examples as its nearest neighbors. Each test example is classified into 'keyword' or 'non-keyword' by the two versions of KNN algorithm; we performed the three independent experiments as many as the domains. The classification accuracy is computed by the number of correctly classified test examples by the number of the test examples for evaluating the both versions of KNN algorithm.

In Figure 9, we illustrate the experimental results from deciding whether each word is a keyword, or not. The y-axis indicates the value of accuracy and the x-axis indicate the group of two versions by a domain of Opinopsis. In each group, the gray bar and the black indicate the achievements

of the traditional version and the proposed version of KNN algorithm. In Figure 9, the most right group indicates the averages over results of the left three groups. Therefore, Figure 9 show the results from classifying each group into one of 'keyword' and 'non-keyword', by both versions of KNN algorithm.

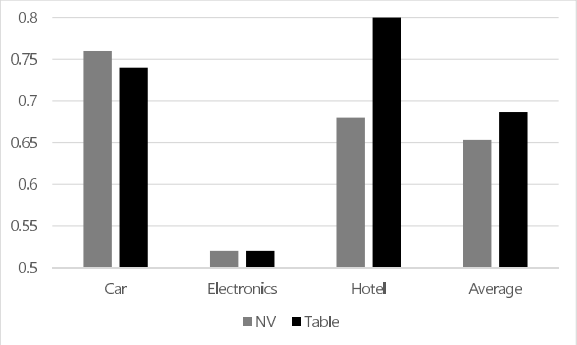

Figure 9. Results from Recognizing Keywords in Text Collection: Opiniopsis

We discuss the results from doing the keyword extraction which is mapped into a binary classification, using the both versions of KNN algorithm, on Opinosis, shown in Figure IV-A. The accuracy values of the both versions range between 0.5 and 0.8. The proposed version works outstandingly better than the traditional one in one of the three domains: Hotel. It is comparable with the traditional version in the others: Car and Electronics. From this set of experiments, we conclude the proposed version works slightly better than the traditional version in averaging the three cases.

C. 20NewsGroups I: General Version

This collection is concerned with one more set of experiments for validating the better performance of the proposed version on text collection: 20NewsGroups I. We gather words which are labeled with 'keyword' or 'non-keyword' from each broad category of 20NewsGroups, under the view of the keyword extraction into a binary classification. The task in this set of experiments is to classify each word exclusively into one of the two categories in each topic which is called domain. We fix the input size to 50 in encoding words, and use the accuracy as the evaluation measure. Therefore, in this section, we observe the performances of the both versions in the four different domains.

In Table III, we specify the general version of 20NewsGroups which is used for evaluating the two versions of KNN algorithm. In 20NewsGroup, the hierarchical classification system is defined with the two levels; in the first level, the six categories, alt, comp, rec, sci, talk, misc, and soc, are defined, and among them, the four categories are selected, as shown in Table III. In each category, we select 1000 texts at random and extract 250 words from them. Among the 250 words, one half of them is labeled with 'keyword', and the other half is labeled with 'non-keyword'. As shown in Table III, the 250 words is partitioned into the 200 words in the training set, and the 50 words in the test set, keeping the complete balance over them. In the process of gathering the classified words, each of them is labeled manually into one of the two categories by scanning individual texts.

Table III
THE NUMBER OF TEXTS AND WORDS IN 20NEWSGROUPS I

Category	#Texts	#Training Words	#Test Words
Comp	1000	200 (100+100)	50 (25+25)
Rec	1000	200 (100+100)	50 (25+25)
Sci	1000	200 (100+100)	50 (25+25)
Talk	1000	200 (100+100)	50 (25+25)

The experimental process is identical is that in the previous sets of experiments. We collect the words by labeling manually them with 'keyword' or 'non-keyword' by scanning individual texts in each of the four domains, comp, rec, sci, and talk, and encode them into numerical vectors and tables with the input size fixed to 50. For each test example, we compute its similarities with the 200 training examples, and select the three similar ones as its nearest neighbors. The versions of KNN algorithm classify each of the 50 test examples into one of the two categories by voting the labels of its nearest neighbors. Therefore, we perform the four independent set of experiments as many as domains, in each of which the two versions are compared with each other in the binary classification task.

In Figure 10, we illustrate the experimental results from deciding whether each word is a keyword or not on the broad version of 20NewsGroups. Figure 10 has the identical frame of presenting the results to those of Figure 8 and 9. In each group, the gray bar and the black bar indicates the achievements of the traditional version and the proposed version of KNN algorithm, respectively. Each group in the x axis indicates the domain within which each word is judged as a keyword or a non-keyword. This set of experiments consists of the four binary classifications in each of which each word is classified into one of the two categories.

Let us discuss the results from doing the keyword extraction using the both versions of KNN algorithm, on the broad version of 20NewsGroups. The accuracies of the both versions of KNN algorithm range between 0.47 and 0.64. The proposed version shows the better performance in the two domains, comp and talk. However, it shows its competitive performances in the others. From this set of experiments, the proposed version keeps its better performance, in averaging its four achievements, in spite of its some leaded ones.

D. 20NewsGroups II: Specific Version

This section is concerned with one more set of experiments where the better performance of the proposed version is validated on another version of 20NewsGroups.

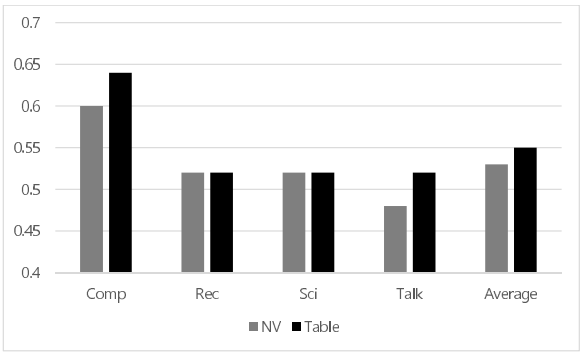

Figure 10. Results from Recognizing Keywords in Text Collection: 20NewsGroup I

We gather the words which are labeled with 'keyword' or 'non-keyword'. We map the keyword extraction into a binary classification, and carry out the independent four binary classification tasks as many as topics, in this set of experiments. We fix the input size in representing words to 50, and use the accuracy as the evaluation metric. Therefore, in this section, we observe the performances of the both versions of the KNN with the four different domains.

In Table IV, we specify the second version of 20NewsGroups which is used in this set of experiments. Within the general category, sci, the four categories, electro, medicine, script, and space, are predefined. In each specific category as a domain, we build the collection of labeled words by extracting 250 important words from approximately 1000 texts. We label manually the words with 'keyword' or 'non-keyword', maintaining the complete balance. In each domain, the set of 250 words is partitioned with the training set of 200 words and the test set of 50 words, as shown in Table IV.

Table IV
THE NUMBER OF TEXTS AND WORDS IN 20NEWSGROUPS II

Category	#Texts	#Training Words	#Test Words
Electro	1000	200 (100+100)	50 (25+25)
Medicine	1000	200 (100+100)	50 (25+25)
Script	1000	200 (100+100)	50 (25+25)
Space	1000	200 (100+100)	50 (25+25)

The process of doing this set of experiments is same to that in the previous sets of experiments. We collect the sample words which are labeled with 'keyword' or 'non-keyword', in each of the four domains: 'electro', 'medicine', 'script', and 'space, and encode them, fixing the in input size to 50. We use the two versions of KNN algorithm for their comparisons. Each example is classified into one of the two categories, by the both versions. We use the classification accuracy as the evaluation metric.

We present the experimental results from classifying the words using the both versions of KNN algorithm on the specific version of 20NewsGroups. The frame of illustrating the classification results is identical to the previous ones. In each group, the gray bar and the black bar stand for the achievements of the traditional version and the proposed version, respectively. The y-axis in Figure 11, indicates the classification accuracy which is used as the performance metric. In this set of experiments, we execute the four independent classification tasks which correspond to their own domains, where each word is classified into 'keyword' or 'non-keyword'.

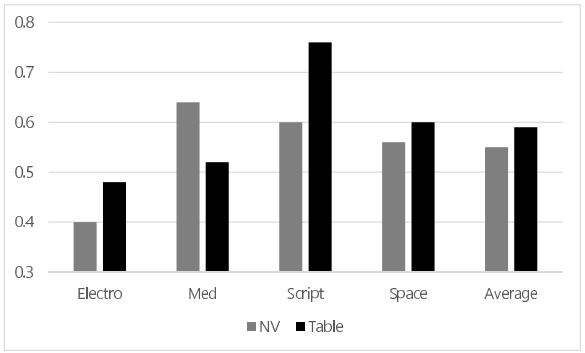

Figure 11. Results from Recognizing Keywords in Text Collection: 20NewsGroup II

Let us discuss on the results from doing the keyword extraction on the specific version of 20NewsGroups, as shown in Figure 11. The accuracies of both versions of KNN algorithm range between 0.40 and 0.76. The proposed version shows its better results in three of the four domains. However, it is leaded by the traditional version, in the domain, 'medicine'. From this set of experiments, it is concluded that the proposed version is slightly better by averaging over the accuracies of the four domains.

V. CONLUSION

Let us discuss the entire results from extracting keywords using the two versions of KNN algorithm. The both versions are compared with each other in the task of word classification which is mapped from the keyword extraction, in these sets of experiments. The proposed version shows its better results in all of the four collections. The accuracies of the traditional version range between 0.46 and 0.76 and those of the proposed version range between 0.49 and 0.80. From the four sets of experiments, we conclude the proposed version improved the keyword extraction performance as the contribution of this research.

Let us consider the remaining tasks for doing the further research. In one of specific domains such as engineering, science, and medicine, we need to validate and customize the proposed research in the keyword extraction, by transforming it into a classification task. Because various schemes of weighting words are available, more than one weight may be assigned to each word, so it need to be considered in computing the similarity between tables. Other machine

learning algorithms may be modified as well as KNN into their table based versions. By adopting the proposed approach, we implement the keyword extraction system as a module of other programs or an independent program.

REFERENCES

[1] K. Abainia, S. Ouamour, and H. Sayoud. "Neural Text Categorizer for topic identification of noisy Arabic Texts", 1-8, Proceedings of 12th IEEE Conference on Computer Systems and Applications, 2015.

[2] L.D. Baker and A. K. McCallum , "Distributional clustering of words for text classification", pp96-103 Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, 1998.

[3] T. Jo, "Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578", 875-882, Journal of Korea Multimedia Society, Vol 11, No 6, 2008.

[4] T. Jo, "Neural Text Categorizer for Exclusive Text Categorization", 77-86, Journal of Information Processing Systems, Vol 4, No 2, 2008.

[5] T. Jo, "NTC (Neural Text Categorizer): Neural Network for Text Categorization", 83-96, International Journal of Information Studies, Vol 2, No 2, 2010.

[6] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", 839-849, Soft Computing, Vol 19, No 4, 2015.

[7] T. Jo, "Encoding Words into Graphs for Clustering Word by AHC Algorithm", 90-95, The Proceedings of 12th International Conference on Multimedia Information Technology and Applications, 2016.

[8] T. Jo, "Modification of K Nearest Neighbor into String Vector based Version for Classifying Words in Current Affairs", 72-75, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.

[9] T. Jo, "K Nearest Neighbor specialized for Word Categorization in Current Affairs by Graph based Version", 64-65, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.

[10] T. Jo, "Extracting Keywords from News Articles using Feature Similarity based K Nearest Neighbor", 68-71, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.

[11] T. Jo, "Modifying K Nearest Neighbor into String Vector based Version for Extracting Keywords from News Articles", 43-46, The Proceedings of International Conference on Applied Cognitive Computing, 2018.

[12] T. Jo, "Graph based K Nearest Neighbors for Keyword Extraction in Current Affair Domain", 47-48, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.

[13] T. Jo, "Summarizing News Articles by Feature Similarity based Version of K Nearest Neighbor", 51-52, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.

[14] T. Jo, "Semantic Word Categorization using Feature Similarity based K Nearest Neighbor", 67-78, Journal of Multimedia Information Systems, 2018.

[15] T. Jo, "Using Table based AHC Algorithm for clustering Words in Domain on Current Affairs", 1222-1225, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[16] T. Jo, "Modification into Table based K Nearest Neighbor for News Article Classification", 49-50, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.

[17] T. Jo, "String Vector based AHC Algorithm for Word Clustering from News Articles", 83-86, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.

[18] T. Jo, "Improving K Nearest Neighbor into String Vector Version for Text Categorization", 1091-1097, ICACT Transaction on Communication Technology, Vol 7, No 1, 2018.

[19] T. Jo, "String Vector based K Nearest Neighbor for News Article Summarization", 146-149, The Proceedings of 21st International Conference on Artificial Intelligence, 2019.

[20] T. Jo, "Validation of Graph based K Nearest Neighbor for Summarizing News Articles", 5-8, The Proceedings of International Conference on Green and Human Information Technology Part II, 2019.

[21] T. Jo, "Applying Table based AHC Algorithm to News Article Clustering", 8-11, The Proceedings of International Conference on Green and Human Information Technology, Part I, 2019.

[22] T. Jo, "Introduction of String Vectors to AHC Algorithm for Clustering News Articles", 150-153, The Proceedings of 21st International Conference on Artificial Intelligence, 2019.

[23] T. Jo, "Graph based Version of K Nearest Neighbor for classifying News Articles", 4-7, The Proceedings of International Conference on Green and Human Information Technology Part I, 2019.

[24] T. Jo, "Graph based Version for Clustering Texts in Current Affair Domain", 171-174, The Proceedings of 15st International Conference on Data Science, 2019.

[25] T. Jo and D. Cho, "Index Based Approach for Text Categorization", 127-132, International Journal of Mathematics and Computers in Simulation, Vol 2, No 1, 2007.

[26] R. J. Kate and R. J. Mooney, "Using String Kernels for Learning Semantic Parsers", 913-920, Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, 2006.

[27] A. Karatzoglou and I. Feinerer, "Text Clustering with String Kernels in R", 91-98, Advances in Data Analysis, 2006.

[28] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text Classification with String Kernels”, 419-444, Journal of Machine Learning Research, Vol 2, No 2, 2002.

[29] L. Vega and A. Mendez-Vazquez, “Dynamic Neural Networks for Text Classification”, 6-11, The Proceedings of International Conference on Computational Intelligence and Applications, 2016.