Application of Table based K Nearest Neighbor for Index Optimization

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 index optimization. The motivations of this research are the successful results from applying the table based algorithms to the text categorizations in previous works and the index optimization is able to be viewed into a classification task where each word is classified into expansion, inclusion, and removal. In the proposed system, each word in the given text is classified into one of the three categories by the proposed KNN algorithm, associates words are added to ones which are classified into expansion, and ones which are classified into inclusion are kept by themselves without adding any word. The proposed KNN version is empirically validated as the better approach in deciding the importance level of words in news articles and opinions. In using the table based KNN algorithm, it is easier to trace results from categorizing words.**

***Keywords*-Index Optimization, Table Similarity, Table based KNN**

I. Introduction

Index optimization refers to the process of deciding each word into inclusion, or removal, in order to generate its optimized index for maximizing the information retrieval performance. In this research, in addition to the inclusion and the removal, we consider the expansion which adds more semantically similar words from the given corpus to the essential words. The index optimization is interpreted into the classification task where each word is classified into the three categories: expansion, inclusion, and removal. Under the interpretation, it is assumed that unsupervised learning algorithms will be used as the approach to the task. The scope of this research is restricted to only crisp index optimization which does not allow each word to be overlapped over more than one category.

In encoding words into numerical vectors, some problems are inevitable. If we use texts as features of numerical vectors representing words, many features are required for keeping enough robustness of the system [2]. The discrimination among numerical vectors representing them is very poor because of the sparse distribution in each of them [25]. When using grammatical properties as features, instead of texts, it becomes very difficult and complicated to implement the encoding process. Therefore, this research attempts to solve the above problems by encoding words into tables, instead of numerical vectors.

Let us propose some agenda in this research, in order to solve the above problems. In this research, we propose the index optimization as an instance of classification task for improving the cohesion of indexed words. We encode each word into a table which consists of entries of texts including it and its weights in them. We define the similarity measure between tables which is always given as a normalized value, and using it, modify the KNN (K Nearest Neighbor) where a table is directly given as the input data. Hence, we apply the modified KNN to the classification task which is transformed from the index optimization.

This research may be expected to provide some benefits for developing the text segmentation system. The table which represents a text may be regarded as the more symbolic and compact text representation; it needs much less than one hundred elements for maintaining the system robustness. We may expect much more discrimination among tables than among numerical vectors, because the sparse distribution does not happen in encoding texts into tables. We expect both better performance and more stability from the proposed KNN version because the similarity measure between tables is always given as a normalized value between zero and one. However, note that the table size is given as the additional parameter, so it should be optimized between the computation speed and the system reliability.

This article is organized into the five sections. In Section II, we survey the relevant previous works. 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 general discussion on the empirical validations and remaining tasks for doing the further research.

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 cases of applying the modern KNN algorithm to the index optimization and its similar tasks. The topic based word categorization is the typical semantic word classification, and the keyword extraction and the index optimization are derived from it. The KNN algorithm is modified for improving the classification performance by solving the problems in encoding words into numerical vectors. We present the better performance of the modern KNN algorithm in the three tasks than the traditional one. This section is intended to explore the previous works on the modernized KNN algorithm and its applications to the tasks.

Let us mention applying the modernized KNN algorithm to the topic based word categorization as the source from which the index optimization is derived. The KNN algorithm which adopted the similarity metric between numerical vectors considering the similarities among features was proposed as the approach to the topic based word categorization [9]. The KNN algorithm which classified a string vector directly, was adopted as the approach to the task [10]. The KNN algorithm was modified into the version which classified a graph [11]. The task which is mentioned in the previous works, mentioned above, is to classify each word into one among predefined topics.

The keyword extraction is derived from the word categorization as its special case. The version of KNN algorithm which was modified by replacing cosine similarity metric by one considering the feature similarities was applied to keyword extraction [12]. The KNN algorithm was modernized into the version which processes string vectors directly, as the approach to the keyword extraction [13]. The KNN version which classifies a graph was used for extracting keywords from text [14]. In the above literatures, the keyword extraction is viewed into the binary classification where each word is classified into keyword or non-keyword.

Let us mention the previous cases of applying the modernized KNN algorithms to the task which covered in this study. The KNN algorithm which considers the feature similarities, in computing the similarity between two numerical vectors, was applied to the index optimization [15]. The modernized KNN version which receives a string vector as an input data, instead of numerical vector, was used for the index optimization [16]. The KNN algorithm which is modernized into the version which processes graphs directly, was proposed as the approach to the index optimization [7]. In the above literatures, the index optimization was interpreted as the classification of each word into expandion, inclusion, or removal.

Let us mention some points which distinguish this research from the ones which were surveyed above. We presented the cases of applying the three kinds of modernized KNN algorithms to the index optimization and its related tasks. We mentioned the word categorization as the source from which the index optimization is derived as its specific instance and the keyword extraction as another specific instance which is derived from the word categorization, together with the index optimization which is covered in this study. The modernized version which is proposed in this research classifies a table which represent a word, directly. The index optimization is mapped into the task which classifies a word into one of the three categories, following style in the previous works, and the proposed version is applied to the task, in this study.

B. Word and Text Encoding

This section is concerned with the previous works on encoding texts or words into the replacements of numerical vectors. The problems in encoding them into numerical vectors were already mentioned in previous works. They tried to solve the problems by encoding words and texts into other forms which replace the numerical vectors. In this section, we mention the tables, the string vectors, and the graphs as the replacements of numerical vectors. This section is intended to explore the previous works on encoding texts or words into replacements in other tasks.

Let us explore the previous works on encoding texts or words into tables. Words were encoded into tables in using the AHC algorithm for the word clustering [17]. Texts were encoded into tables in using the KNN algorithm for the text categorization [18]. Texts were encoded so in using the AHC algorithm for the text clustering [21]. In the above literatures, texts or words were encoded into tables in using the KNN algorithm and the AHC algorithm.

Let us survey 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 [19]. Texts were encoded into string vectors, in using the KNN algorithm for categorizing texts [20]. In using the AHC algorithm for clustering texts, texts were also encoded so [22]. We present the previous cases of encoding raw data into string vectors.

Let us mention the previous works on encoding words or texts into graphs. It was proposed that words should be encoded into graphs, in using the AHC algorithm for clustering words, semantically [8]. It was proposed that texts should be encoded into graphs, in using the KNN algorithm for categorizing texts [23]. It was proposed that texts should be encoded so, in using the AHC algorithm for clustering texts [24]. In the above literatures, we present the previous cases of mapping raw data into graphs in order to modernize the machine learning algorithm.

We mentioned the three types of structured data as word representations or text representations, in the previous works. In this research, we adopt the first type of structured data, called tables. We define the similarity metric between tables and modify the KNN algorithm into the version which processes tables directly. We use the modified version of KNN algorithm for implementing the index optimization system. The modified version is validated in the classification task which is mapped from the index optimization, comparing it with the traditional version.

C. Non-Numerical Vector based Machine Learning Algorithms

This section is concerned with the previous works on the non-numerical vector based machine learning algorithms. In the previous section, we presented the cases of encoding words or texts into tables, string vectors, or graphs. In this section, as the approaches to the text categorization where texts are encoded into non-numerical vectors, we mention the three supervised learning algorithms: the string kernel based Support Vector Machine, the table matching algorithm, and the Neural Text Categorizer. The index optimization is mapped into the classification task, and it belongs to the classification, together with the text classification. This section is intended to survey the previous works on the non-numerical vector based machine learning algorithms which are used for categorizing texts.

Let us mention the string kernel as the kernel function of two raw texts, instead of two numerical vectors. The string kernel was initially proposed for modifying the SVM (Support Vector Machine) as the approach to the text categorization by Lodhi et al. in 2002 [28]. It was utilized for modifying the k means algorithm as the approach to the text clustering by Karatzoglou and Feinerer in 2006 [27]. The string kernel based SVM was applied to the sentence classification by Kate and Mooney in 2006 [26]. The string kernel which was proposed and used in the above literatures is the similarity metric between two raw texts, depending on characters.

Let us explore the previous works on the table based matching algorithm as a non-numerical vector based classification 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 where each text is allowed to be classified into more than one category [3]. It was improved into the more robust and stable version by Jo in 2015 [6]. Texts should be encoded into tables, instead of numerical vectors, in using the table based matching algorithm which are presented in the above literatures.

Let us mention the Neural Text Categorizer as a non-numerical vector based neural networks. It was initially proposed as the approach to the text categorization by Jo in 2008 [4]. It was empirically validated into its better performance than those of the Naive Bayes and the SVM in both soft and hard text categorization tasks in 2010 [5]. It was applied to the classification of Arabian texts by Abainia et al. in 2015 [1]. It was mentioned as an innovative approach in the research paper about the dynamic neural networks for the text categorization by Vega and Mendez-Vasquez in 2016 [29].

We mentioned the three non-numerical vector based classification algorithms as the approaches to the text categorization. The development of the non-numerical vector based algorithms is intended to solve the problems in encoding texts into numerical vectors by representing them into other structured forms. In this research, words are encoded into tables, following the cases in using the table based matching algorithm. The KNN algorithm is modified into the version which deals with tables directly as the approach to the index optimization. The task is interpreted into the classification of each word based on its importance degree into expansion, inclusion, and removal.

III. Proposed System

This article is concerned with the index optimization 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 3, 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 architecture of the index optimization system where the proposed KNN is adopted.

A. Word Encoding

This section is concerned with the process of mapping words into tables. In Section II-B and II-C, we presented the previous cases of encoding raw data into tables. In this study, words are encoded into tables with the three steps: corpus indexing, inverted indexing, and term weighting. A table as a word representation consists of entries each of which is composed of a text identifier and a weight. This section is intended to describe the three steps which are presented in Figure 1-3.

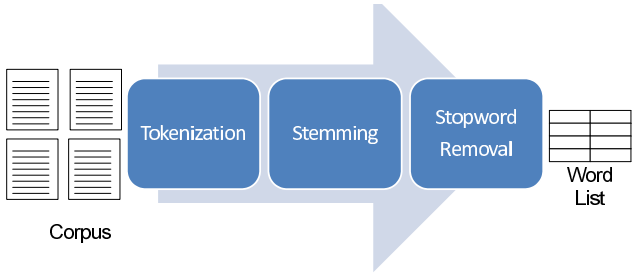

Figure 1. Overall Process of Word Indexing

Figure 1 illustrates the three phases of indexing a corpus into a list of words. The tokenization is the partition of texts in the corpus into tokens by white spaces or punctuation marks. The steaming is the process of mapping each token into its root form by the stemming rules. The stop-word removal is the process of excluding the grammatical words such as prepositions or conjunctions from the list for more efficiency. The words which belong to noun, verb, or adjective remain from the three steps which are shown in Figure 1.

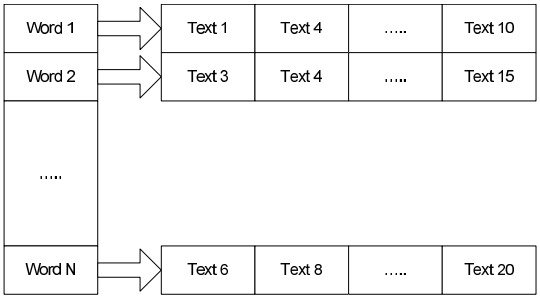

Figure 2. Inverted Index

The index structure where each word is linked to its related texts is illustrated in Figure 2. In the previous step which is presented in Figure 1, the index structure where each text is linked to its related words is constructed. In this step, the text based index structure is converted into the word based index structure. The fact that the axis is given on words instead of texts becomes the reason of calling it inverted index. A list of texts which is linked to each word becomes the information for encoding it into a table.

The process of weighting text identifiers in the table which represents a word is illustrated in Figure 3. In the previous step, the inverted index which is presented in Figure 2, was constructed as the basis. The weight which is the relationship between a word and a text is computed by the equation which is presented in Figure 3. The TF-IDF weight which is assigned to each text identifier is proportional to the word frequency in the text, but inversely proportional to the number of texts in the corpus which includes the word, called document frequency. The corpus is required for computing the TF-IDF weight, as presented in Figure 3.

A word is encoded into a table through the three steps which are presented in Figure 1-3. A table is defined as a set of entries, and each entry consists of more than one value. The entry consists of a text identifier and its TF-IDF weight in the table which represents a word. If multiple weight schemes are adopted, each entry is expanded into one which consists of a text identifiers and its multiple weights. We need to define the operations on tables for modifying machine learning algorithms into the versions which process them directly.

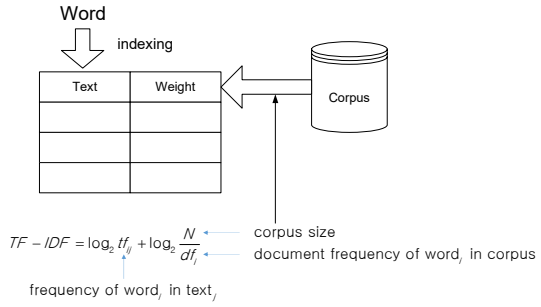

Figure 3. Text Weighting

B. Similarity Metric

This section is concerned with the similarity metric between two tables. In the previous section, we mentioned the process of mapping words into tables. We need to define the similarity between tables for modifying the KNN algorithm as the approach to the index optimization. We view a table as a set of entries each of which consists of a text identifier and a weight and compute the similarity between them based on text identifiers shared by them. This section is intended to describe the process of computing the similarity between 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 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),

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

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

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),

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

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

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

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

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

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

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),

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

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

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

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

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

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

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

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 (19),

$$T_1 = T_2 \tag{14}$$

the similarity between them is 1.0, as shown in equation (20),

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

If the two tables are completely different from each other as shown in equation (21),

$$F(T_1) \cap F(T_2) = \oslash, |ST| = 0 \tag{16}$$

the similarity between them is zero, as shown in equation (22),

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

The similarity between two tables is given as a normalized value between zero and one, as shown in equation (23),

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

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

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 (19),

$$T_1 = T_2 \tag{19}$$

the similarity between them is 1.0, as shown in equation (20),

$$\begin{aligned} sim(T_1, T_2) &= \frac{W_1(ST) + W_2(ST)}{W(T_1) + W(T_2)} \\ &= \frac{2W_1(ST))}{2W(T_1)} = \frac{2W_1(T_1))}{2W(T_1)} = 1.0 \end{aligned} \tag{20}$$

If the two tables are completely different from each other as shown in equation (21),

$$F(T_1) \cap F(T_2) = \oslash, |ST| = 0 \tag{21}$$

the similarity between them is zero, as shown in equation (22),

$$sim(T_1, T_2) = \frac{W_1(ST) + W_2(ST)}{W(T_1) + W(T_2)} = \frac{0}{W(T_1) + W(T_2)} = 0.0 \quad (22)$$

The similarity between two tables is given as a normalized value between zero and one, as shown in equation (23),

$$ST \subseteq T_1, ST \subseteq T_2 \quad W_1(ST) + W_2(ST) \leq W(T_1) + W(T_2) \quad (23)$$

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 illustrated in Figure 4 as the approach to the index optimization. We describe 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 the training examples. In addition, variants may be derived by considering more selection schemes and voting ones. This section is intended to describe the proposed version of KNN algorithm which classifies tables directly and its variants.

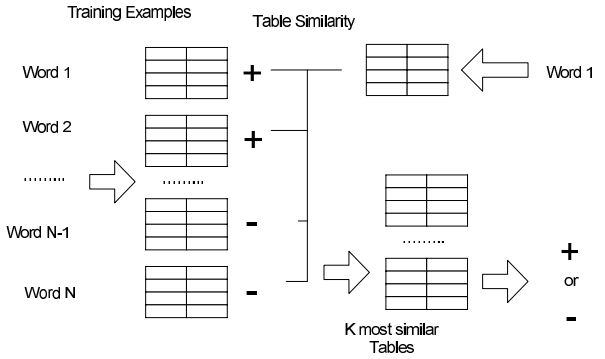

Figure 4. The Proposed Version of KNN

Let us mention the process of selecting nearest neighbors as the references for classifying a data item from the training examples. The training words and a novice word are encoded into tables by the process which was described in Section III-A. The similarities of a novice item with the training ones are computed by 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 (24),

$$Ne_k(T) = \{(T_1, y_1), (T_2, y_2), \ldots, (T_k, y_k)\}, \quad y_i \in \{c_1, c_2, \ldots, c_m\} \quad (24)$$

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 (25),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Count(Ne_k(T), c_i) \quad (25)$$

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 (26),

$$w_i = sim(T, T_i) \quad (26)$$

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 (27),

$$Weight(Ne_k(T), c_i) = \sum_{T_j \in c_i}^{k} w_j \quad (27)$$

The label of the novice item, T, is decided by the category which corresponds to the maximum sum of weights as shown in equation (28),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Weight(Ne_k(T), c_i) \quad (28)$$

When the weights of nearest neighbors are set constantly, equation (28) is same to equation (25), as expressed in equation (29),

$$Weight(Ne_k(T), c_i) = Count(Ne_k(T), c_i) \quad (29)$$

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. Index Optimization System

This section is concerned with the index optimization 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 index optimization. It is viewed into the classification of each word into one of the three categories: expansion, inclusion, and removal. The scope is restricted to indexing of a text into words, encoding words into tables, and classifying each table into one of the three categories. This section is intended to describe the index optimization system with respect to its functions and its architecture.

The sample words which are labeled with one of the three categories are illustrated in Figure 5. The index optimization is viewed as the classification where each word is classified into one of expansion, inclusion, and removal. The topic based word classification belongs to the domain independent task where a word is classified identically with regardless of domain, whereas the index optimization belongs to the domain dependent task where it may be classified differently depending on the domain. Domain by domain, sample words which are labeled with one of the three categories are collected as shown in Figure 5. Before executing the index optimization in the system, the input text domain should be presented.

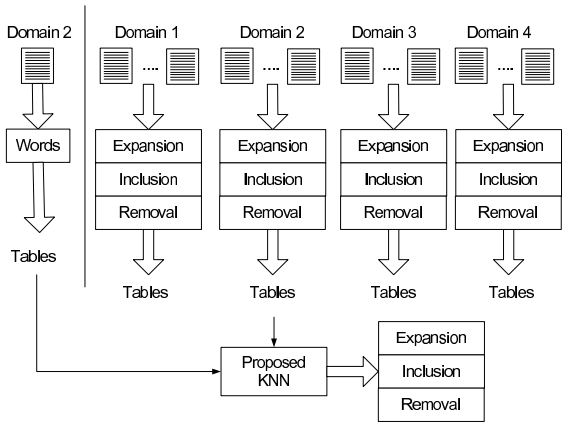

Figure 5. Sample Words

The entire architecture of the proposed index optimization system is illustrated in Figure 6. A text is given as the input and the words are extracted from it in the indexing module. The sample words in the three groups, the expansion group, the inclusion one, and the removal one, and ones indexed from the text are mapped into tables in the encoding module. The words are classified into one of the three categories, in the similarity computation module and the voting module. The words which are classified into removal will be discarded in the system.

The execution process of the proposed system is illustrated as a block diagram in Figure 7. The sample words

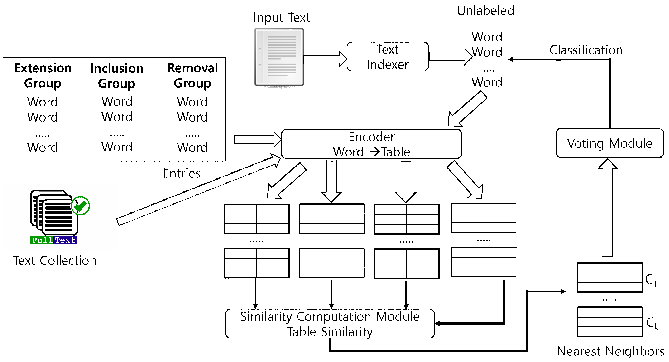

Figure 6. Proposed System Architecture

which are labeled with one of the three categories 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 the similarity computation and the label of decided by ones of its nearest neighbors. The words which are classified into removal are removed from the system.

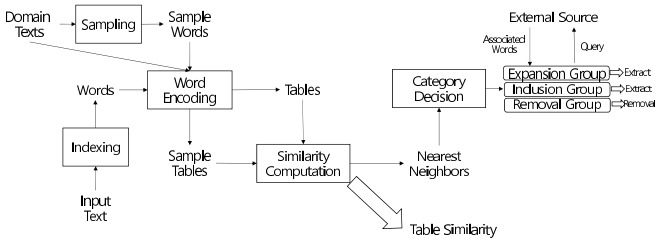

Figure 7. Execution Process of Proposed System

Let us make some remarks on the proposed system which is illustrated in Figure 6, as the architecture. The index optimization is defined as the classification task where each word is classified into expansion, inclusion, or removal. Each word is encoded into a table instead of a numerical vector, and a table is classified directly. Ones which are classified into removal are excluded from indexing the input ext. We need to add module which retrieve more words which are relevant to ones which are labeled with expansion for implementing the index expansion.

IV. Experiments

This section is concerned with the empirical experiments for validating the proposed version of KNN, and consists

of the four sections. In Section IV-A, we present the results from applying the proposed version of KNN to the index optimization on the collection, NewsPage.com. In Section IV-B and IV-C, we mention the results from comparing the two versions of KNN with each other in the task of index optimization 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 index optimization into the trinary classification where each word is classified into expansion, inclusion, and removal, and gather words which are labeled with one of the three categories, from the collection, topic by topic. Each word is allowed to be classified into one of the three labels, exclusively. We fix the input size as 50 dimensions of numerical vectors and 50 entries of tables, 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, NewsPage.com, was used in previous works for evaluating approaches to text categorization [6]. In each topic, 375 words are extracted: 125 words labeled with expansion, 125 words labeled with inclusion, and 125 words labeled with removal. In each category, the set of 375 words is portioned into the 300 words as training examples and the 75 words as the test example, keeping the balanced distributions over the three labels. We decide target labels of words by their frequencies concentrated in the given category, combined with the subjectivity in scanning texts.

Table I
THE NUMBER OF TEXTS AND WORDS IN NEWSPAGE.COM

Category	#Texts	#Training Words	#Test Words
Business	500	300(100+100+100)	75(25+25+25)
Health	500	300(100+100+100)	75(25+25+25)
Internet	500	300(100+100+100)	75(25+25+25)
Sports	500	300(100+100+100)	75(25+25+25)

Let us mention the experimental process of validating empirically the proposed approach to the task of index optimization. We collect sample words which are labeled with expansion, inclusion, or removal, 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 75 test examples, the KNN computes its similarities with the 300 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 one of the three labels by the two versions of KNN algorithm. For evaluating the both versions of KNN in the classification which is mapped from the index optimization, 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 classifying the words into one of the three categories as the process of index optimization, using the both versions of KNN algorithm. The y-axis indicate the accuracy which is the rate of the correctly classified words in the test set. In the x-axis, each group indicates the domain within which the index optimization which is viewed as the classification task is performed, independently. In each group, the gray bar and the black bar indicate the achievements of the traditional version and the proposed version, respectively. In the x-axis, the most right group indicates the average over the accuracies of the left four groups, and the input size which is the dimensional of numerical vectors is fixed to 50.

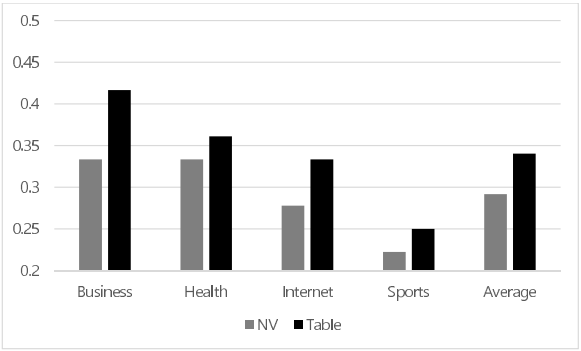

Figure 8. Results from Index Optimization in Text Collection: NewsPage.com

Let us make the discussions on the results from doing the index optimization, using the both versions of KNN algorithm, as shown in Figure 8. The accuracy which is the performance measure of this classification task is in the range between 0.33 and 0.42. The proposed version of KNN algorithm works better in the all domains. The performance difference between the two versions is outstanding in the two domains, Business and Internet. From this set of experiments, we conclude that the proposed version works better than the traditional one, in averaging over the four cases.

B. 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 ‘expansion’, ‘inclusion’ or ‘removal’ from each broad category of 20NewsGroups, under the view of the index optimization into a binary classification. The task in this set of experiments is to classify each word exclusively into one of the three 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 II, 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 II. In each category, we select 1000 texts at random and extract 375 words from them. Among the 375 words, one third of them is labeled with 'expansion', the second third is labeled with 'inclusion', and the other third is labeled with 'removal'. As shown in Table II, the 375 words is partitioned into the 300 words in the training set, and the 75 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 three categories by scanning individual texts.

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

Category	#Texts	#Training Words	#Test Words
Comp	1000	300(100+100+100)	75(25+25+25)
Rec	1000	300(100+100+100)	75(25+25+25)
Sci	1000	300(100+100+100)	75(25+25+25)
Talk	1000	300(100+100+100)	75(25+25+25)

The experimental process is identical is that in the previous sets of experiments. We collect the words by labeling manually them with 'expansion', 'inclusion', and 'removal', 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 300 training examples, and select the three similar ones as its nearest neighbors. The versions of KNN algorithm classify each of the 75 test examples into one of the three 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 9, we illustrate the experimental results from deciding the importance degree of each word for maximize the information retrieval performance, on the broad version of 20NewsGroups. Figure 9 has the identical frame of presenting the results to those of Figure 8. 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 one of the three importance degree. This set of experiments consists of the four binary classifications in each of which each word is classified into one of the three categories as the index optimization.

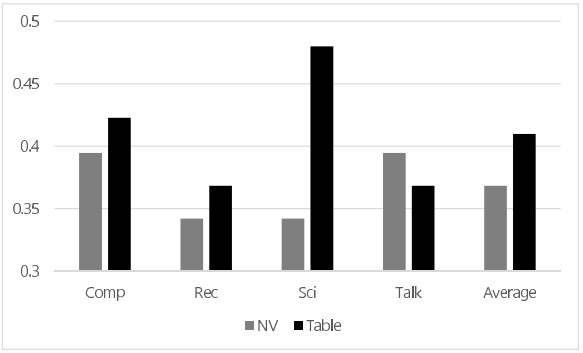

Figure 9. Results from Index Optimization in Text Collection: 20NewsGroup I

Let us discuss the results from doing the index optimization using the both versions of KNN algorithm, on the broad version of 20NewsGroups. The accuracies of the both versions of KNN algorithm range between 0.34 and 0.47. The proposed version shows the better performance in the three of the four domains. However, it shows its competitive performances in the other. From this set of experiments, the proposed version wins over the traditional one, in averaging its four achievements.

C. 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. We gather the words which are labeled with 'expansion', 'inclusion', or 'removal'. We map the index optimization 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 III, 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 375 important words from approximately 1000 texts. We label manually the words with 'expansion', 'inclusion' or 'removal', maintaining the complete balance. In each domain, the set of 375 words is partitioned with the training set of 300 words and the test set of 75 words, as shown in Table III.

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 'expansion', 'inclusion', or 'removal', in each of the four domains: 'electro', 'medicine', 'script', and 'space, and encode them, fixing the in input size

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

Category	#Texts	#Training Words	#Test Words
Electro	1000	300(100+100+100)	75(25+25+25)
Medicine	1000	300(100+100+100)	75(25+25+25)
Script	1000	300(100+100+100)	75(25+25+25)
Space	1000	300(100+100+100)	75(25+25+25)

to 50. We use the two versions of KNN algorithm for their comparisons. Each example is classified into one of the three 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 10, 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 'expansion', 'inclusion', or 'removal'.

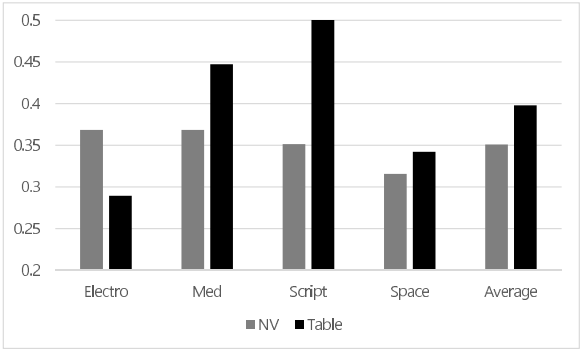

Figure 10. Results from Index Optimization in Text Collection: 20NewsGroup II

Let us discuss on the results from doing the index optimization on the specific version of 20NewsGroups, as shown in Figure 10. The accuracies of both versions of KNN algorithm range between 0.2 and 0.5. The proposed version shows its better results in three of the four domains. However, it is leaded in the domain, 'electro'. In spite of that, from this set of experiments, it is concluded that the proposed version wins over the traditional one, according to the average over the four accuracies.

V. CONCLUSION

Let us discuss the entire results from performing the index optimization 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 index optimization, in these sets of experiments. The proposed version shows its better results in all of the three collections and its matching ones in the others. The accuracies of the traditional version range between 0.21 and 0.39 and those of the proposed version range between 0.20 and 0.50. From the three sets of experiments, we conclude the proposed version improved the index optimization 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 index optimization, 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 index optimization 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, "Graph based KNN for Optimizing Index of News Articles", 53-62, Journal of Multimedia Information System, Vol 3, No 3, 2016.

[8] 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.

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] T. Jo, “Index Optimization in News Articles using Feature Similarity based K Nearest Neighbor”, 106-109, The Proceedings of 17th Int’l Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government, 2018.

[16] T. Jo, “String Vector based Version of K Nearest Neighbor for Index Optimization in Current Affairs”, 47-50, The Proceedings of International Conference on Applied Cognitive Computing, 2018.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[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.