Extracting Keywords from Text by Feature Similarity 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 considers the feature similarity and is applied to the keyword extraction. The texts which are given as features for encoding words into numerical vectors are semantic related entities, rather than independent ones, and the keyword extraction is able to be viewed into a 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. The significance of this research is to improve the classification performance by utilizing the feature similarities.**

I. Introduction

The keyword extraction refers to the process of extracting the essential words which reflect the entire content from an article. In this research, the keyword extraction is viewed into a binary classification where each word is classified into one of the two categories: 'keyword', or 'non-keyword'. We prepare sample words which are labeled with one of the two categories and encode them into their structured forms. By learning the sample words, we built the classification capacity, encode novice words into the structured forms, and classify them into one of the two categories. In this research, we use a supervised learning algorithm for the task which is viewed into the classification task.

Let us mention some problems which this research tries to solve. When discovering the dependencies among features of encoding texts or words into numerical vectors, the Bayesian networks was proposed as the approaches to the text mining tasks, but the complicated analysis is required for using them [29]. The assumption that features are independent of each other causes the requirement of many features for the robustness in encoding so. Since each feature has the very weak coverage, we cannot avoid the sparse distribution of each numerical vector where zero values are dominant with more than 95% [1]. Therefore, this research is intended to solve the problems by considering the feature similarity as well as the feature value similarity.

Let us mention what we propose in this research as its idea. In this research, we consider the both similarity measures, feature similarity and feature value similarity, for computing the similarity between numerical vectors. The keyword extraction is viewed into the binary classification where a supervised learning algorithm is applicable. The KNN (K Nearest Neighbor) is modified into the version which accommodates the both similarity measures and applied to the keyword extraction task. Therefore, the goal of this research is to improve the keyword extraction performance by solving the above problems.

We mention what we expect from this research as the benefits. Considering the both similarities which are covered in this research opens the potential way of reducing the dimensionality of numerical vectors for encoding texts. Computing the similarity between two texts by the two measures reflects more semantic similarity between words. It is expected to improve the discriminations among even sparse vectors by using the both kinds of similarities. Therefore, this research pursues the benefits for implementing the keyword extraction systems.

Let us mention the organization of this research. 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 previous cases of modernizing the KNN algorithm by considering the similarities among features. In Section II-C, we present the literatures which are concerned with the scheme of computing the similarity between texts. Therefore, in this section, we provide the history about this research, by surveying the relevant previous works.

A. Using KNN Algorithm to Text Mining Tasks

This section is concerned with the previous cases of applying the KNN algorithm to text mining tasks. Classifying texts or words belong to a text mining task, and

the KNN algorithm is adopted as the approach to the task in this research. The KNN algorithm belongs to the lazy learning algorithm which does not learn training examples in advance. The fact that the KNN algorithm is popularly used in classification tasks in any domain, as well as text categorization is the reason of adopting and modifying the KNN algorithm. In this section, we survey cases of applying the KNN algorithm to the word categorization, text categorization, and spam mail filtering.

Let us survey the application of the modernized KNN algorithms to the word categorization, before mentioning their applications to the keyword extraction. The KNN algorithm was modified into more suitable version which receives a table as its input data, as the approach to the word categorization [12]. The modernized KNN algorithm which classifies a string vector directly, instead of a numerical vector, was proposed, as the approach to the topic based word categorization [13]. The KNN version which classifies a graph directly was applied to the word categorization [14]. In the above literatures, the modernized versions of KNN algorithm were applied to the word categorization.

Let us survey the cases of using the modernized KNN algorithms for the keyword extraction which is covered in this study. The KNN algorithm which processes tables directly was applied to the keyword extraction [15]. The KNN algorithm which was modernized into the version which processes string vectors directly, instead of numerical vectors is used for extracting keywords from a text [16]. The results from applying the KNN algorithm which is modified into one which processes graphs directly were successful [17]. In the above literatures, the keyword extraction is mapped into the binary classification in applying the modernized KNN algorithm.

Let us mention the text summarization is similar as the keyword extraction, in selecting important paragraphs as the summary. The KNN algorithm which classifies a table directly was applied to the text summarization [22]. The modernized KNN algorithm which deals with the string vectors, instead of numerical vectors, was proposed as the approach to the text summarization [23]. Another KNN version which processes graphs directly was adopted for implementing a text summarization system [24]. The keyword extraction is regarded as the selection of important words, whereas the text summarization is regarded as the selection of important paragraphs.

Let us mention some points which are distinguished from the above literatures. We explored the previous cases of using the KNN algorithms which were modernized in the different directions for the keyword extraction and its related ones. The three kinds of modernized versions were mentioned as the approach to the tasks including the keyword extraction. The modernized version of the KNN algorithm, which is proposed as the keyword extraction is one which considers the similarities among features for computing the similarity between numerical vectors. In the above literatures, the keyword extraction was mapped into the binary classification of words, and the proposed version of KNN algorithm will be to the mapped task as the empirical validations.

B. Feature Similarity

This section is concerned with the previous cases of modernizing the KNN algorithm by considering the similarities among features. The sparse distribution in each numerical vector is the most outstanding issue in encoding texts or words into numerical vectors. In previous works, results in using the modernized version of KNN algorithm to text mining tasks were successful. We mention the three tasks where the performance of the modernized version is better than the traditional one: word categorization, word clustering, and index optimization. This section is intended to survey the previous works on the successful results of using the modernized KNN algorithm to the three tasks.

Let us review some works on the feature similarities in using the KNN algorithm for the word categorization. In 2015, it was initially proposed that the KNN algorithm should use the similarity between numerical vectors, considering the feature similarity [5]. In 2018, the successful results in using the modernized version of KNN algorithm was discovered [18]. The empirical validations in using the modernized KNN algorithm for the word categorization were completed through multiple test sets with different domains [19]. In the above literatures, the modernized KNN algorithm was proposed, and its better performance in the word categorization was confirmed.

Let us review some works on the feature similarities in using the AHC algorithm for clustering words. In 2015, it was initially proposed that the AHC algorithm should use the similarities among features as the approach to the data clustering [6]. In 2018, it was discovered that results from using the above version of AHC algorithm in the data clustering was successful [20]. In 2018, the better performance of the above version of AHC algorithm in clustering texts in various domains was confirmed [21]. The above literatures proposed the modernized version of AHC algorithm and validated its performance through experiments.

Let us review some works on the feature similarities in using the KNN algorithms for the tasks which are derived from the word categorization. In 2015, it was proposed as a position paper that the KNN algorithm should consider the feature similarities in using it for the keyword extraction [7]. The KNN version was also proposed in using it for the index optimization [8]. The better performance of the KNN version which considers the feature similarities was confirmed through the empirical experiments on the index optimization [11]. In the literatures, the KNN version was validated as the better approach to the index optimization.

We mentioned the three tasks where it is effective to consider the feature similarities in the previous works. We try to consider the feature similarities in the keyword extraction, and view it into a binary classification, in this research. We define the similarity metric between numerical vectors, considering the feature similarities, and modify the KNN algorithm, based on it. We apply the modified version of KNN algorithm to the binary classification which is mapped from the keyword extraction. We empirically validate the modified version by comparing it with the traditional version in extracting keywords from a text.

C. Text Similarity

This section is concerned with the previous works on the non-numerical vector based classification algorithm and the semantic operation on strings. In previous section, we presented the cases of using the feature similarities for computing the similarity between a novice example and a training example in using the KNN algorithm. In this section, we present the string kernel based Support Vector Machine and the table based matching algorithm as the previous cases of computing lexical similarity between texts, and mention the previous works on the semantic operation between strings. The reason of surveying the works on the text similarity is that texts are given as features for encoding words into numerical vectors. This section is intended to explore the previous works on the text similarity and the semantic similarity between two strings.

Let us consider the string kernel as the lexical similarity between two raw texts. 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]. It was utilized for improving the k means algorithm performance as well as the SVM one, 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 mentioned in the above literatures is the similarity metric between raw texts based on characters rather than meanings.

There were trials of computing similarity between texts in using the table based matching algorithms. It was initially proposed as the approach to the text categorization by Jo and Cho in 2008 [25]. It was used for the soft categorization of texts which allows to assign more than one category to each text [2]. It was upgraded into the more robust and stable version by Jo in 2015 [9]. In the table based matching algorithm which is covered in the above literatures, the similarity between two texts is computed by encoding them into tables.

Let us mention the semantic operations on strings which are necessary for computing the feature similarities. The semantic operations on strings were initially proposed by Jo in 2012 [3]. They were simulated in the collection of news articles called Reuter 21278, in 2013 [4]. They were simulated in the collection of news articles in the current affair domain and the collection of medical documents called OSHUMED in the medical domain in 2015 [10]. What are proposed in the above literatures is the basis for computing the feature similarities.

We surveyed the previous works on the similarity between two texts and the semantic operation on the strings. In the above literatures, the lexical similarity between texts which are based on characters was proposed in the works on the string kernel, the similarity between tables representing texts is defined in the table based matching algorithm, and the empirical simulations of semantic operations on strings were executed. In this research, we use texts as features in encoding words into numerical vectors, and consider the similarities among texts for computing the semantic similarity between words. The KNN algorithm was modified into version which computes the similarity between items, considering similarities among their features, as approach to the keyword extraction. Its performance will be validated in the classification tasks which are mapped from keyword extractions, compared with the traditional KNN algorithm.

III. Proposed Approach

This section is concerned with modifying the KNN (K Nearest Neighbor) algorithm into the version which considers the similarities among features as well as feature values and its application to the keyword extraction, and it consists of the four sections. In Section III-A, we describe the process of encoding words into numerical vectors. In Section III-B, we do formally the proposed scheme of computing the similarity between two numerical vectors. In Section III-C, we mention the proposed version of KNN algorithm which considers the similarity among features. In Section III-D, we explain the architecture of the keyword extraction system where the proposed KNN is adopted.

A. Word Encoding

This section is concerned with the process of encoding words into numerical vectors. Texts in a corpus becomes the features in encoding words so. The three steps which are presented in Figure 1-3, are involved in encoding words into numerical vectors. In this research, we consider the similarities among texts which were mentioned in Section II-C, for computing the similarity between numerical vectors. This section is intended to describe the three steps: feature extraction, feature selection, and feature value assignment.

The process of extracting features for encoding words into numerical vectors is illustrated in Figure 1. The N words are initially given as encoding targets. For each word, texts are extracted which include itself, and union the sets of texts into a single list of ones. The list of texts from the process which is presented in Figure 1, is given as the feature candidates.

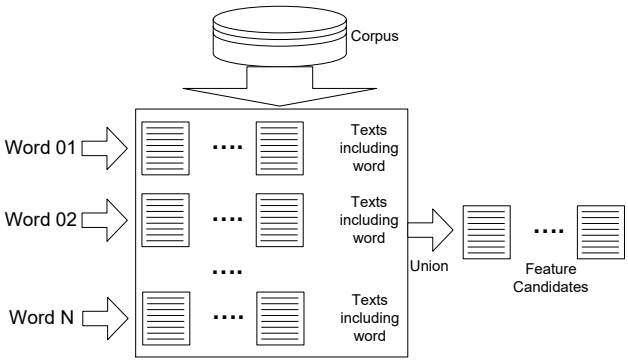

Figure 1. Feature Extraction

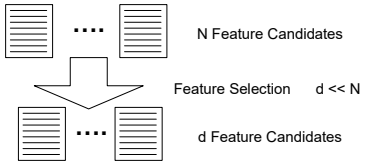

Figure 2. Feature Selection

All texts in the corpus may be set as features, if the corpus is very small.

The process of selecting some among feature candidates is illustrated in Figure 2. The feature candidates are extracted in the process which is illustrated in Figure 1. The relationship between a text and a word such as the frequency, the status of presence or absence, and the TF-IDF (Term Frequency and Inverse Document Frequency) weight, are defined as the selection criteria. Texts are selected by their coverage to words, the total weights of words, or the total frequencies of words. We use the selected texts as features for encoding words into numerical vectors.

The schemes of assigning values to features are illustrated in Figure 3. In the first scheme, a binary values which indicates whether a word is present or absent in the text is assigned to each feature. In the second scheme, the word frequency is assigned to the feature corresponding to the text. In the last scheme, the TF-IDF weight which is computed by the equation in Figure 3 may be assigned. In this research, we adopt the third scheme for assigning numerical values to features in encoding a word into a numerical vector.

We mentioned above the three steps of encoding a word into a numerical vector which are presented in Figure 1-3. In the numerical vector which represents a word, the features are given as texts and the feature values are relationship values between texts and the word. The three hundreds features are used for representing a text into a numerical vectors in [28] on the text categorization. The huge dimensionality and the sparse distribution are main issues in encoding texts and words into numerical vectors. This research is intended to solve the above issues using the similarities among texts which are given as the features.

B. Similarity Metric

This section is concerned with the similarity metric which considers the feature similarities between two numerical vectors. In the previous section, we studied the process of converting words into numerical vectors. We need to define the innovative similarity metric between two numerical vectors, in order to solve the poor discriminations among numerical vectors from their sparse distributions. We mention the feature value similarity which is the similarity of elements in two vectors and the feature similarity which

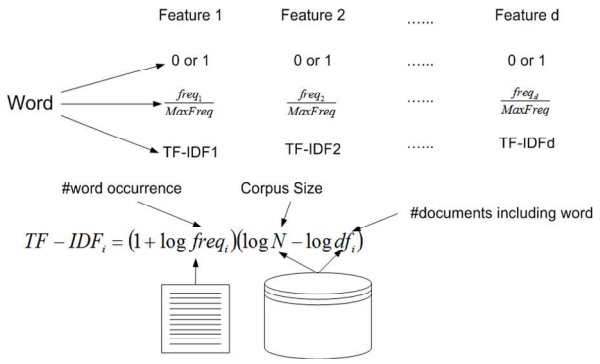

Figure 3. Feature Value Assignment

is one among features. This section is intended to describe the proposed similarity metric between two vectors.

The frame of computing the proposed similarity metric between two numerical vectors is illustrated in Figure 4. $f_1, f_2, \ldots, f_d$ are features of numerical vectors, and $x_1, x_2, \ldots, x_d$ and $y_1, y_2, \ldots, y_d$ are feature values of the two vectors, $\mathbf{x}$ and $\mathbf{y}$. The feature similarity is one between two features, f_i and f_j and the feature value similarity is one between two feature values in the two vectors, x_i and y_i. The similarity metric between two numerical vectors is defined by combining the two kinds of similarities with each other. The value of the proposed similarity metric is always given as a normalized value between zero and one.

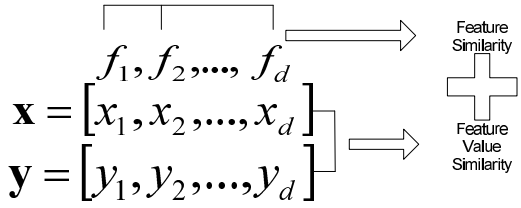

Figure 4. The Combination of Feature and Feature Value Similarity

Let us mention the process of computing the similarity between features which are given as texts. The two features are notated by f_i and f_j, and the similarity between them is notated by $sim(f_i, f_j)$. The two features, f_i and f_j are expressed as the two sets of words, F_i and F_j, and $|F_i|$ and $|F_j|$ become the cardinalities of the two sets. The similarity between the two features is computed by equation (1),

$$sim(f_i, f_j) = \frac{2|F_i \cap F_j|}{|F_i| + |F_j|} \tag{1}$$

and the similarity is always given as a normalized value between zero and one. The similarity between the features is called feature similarity.

Let us mention the process of computing the proposed similarity metric between the two vectors, considering the features. The two numerical vectors, $\mathbf{x}_1$ and $\mathbf{x}_2$ are expressed as equation (2) and (3),

$$\mathbf{x}_1 = [x_{11}\ x_{12}\ \ldots\ x_{1d}] \tag{2}$$

$$\mathbf{x}_2 = [x_{21}\ x_{22}\ \ldots\ x_{2d}] \tag{3}$$

The feature similarity between the two features, f_i and f_j is notated as equation (4),

$$s_{ij} = sim(f_i, f_j) \tag{4}$$

The proposed similarity metric between two vectors is defined by equation (5),

$$sim(\mathbf{x}_1, \mathbf{x}_2) = \frac{\sum_{i=1}^{d} \sum_{j=1}^{d} s_{ij} x_{1i} x_{2j}}{d \cdot \|\mathbf{x}_1\| \cdot \|\mathbf{x}_2\|} \tag{5}$$

Equation (5) reflects the assignment of feature similarities as weights to the product of elements in different features. The similarity between two vectors which is computed by equation (5), is always given as a normalized value between zero and one.

We mentioned the proposed similarity metric between two numerical vectors as a normalized value between zero and one. The similarity between two vectors which is computed by equation (5) is given as 1.0, if the conditions are expressed as equation (6),

$$\begin{aligned} \forall_{i,j} s_{ij} = 1.0, \mathbf{x}_1 = \mathbf{x}_2 \\ x_{11} = x_{21}, x_{12} = x_{22}, \ldots, x_{1d} = x_{2d} \end{aligned} \tag{6}$$

The similarity between two vectors is given as 0.0, if the conditions are expressed as equation (7),

$$\forall_{i,j} s_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

If the feature similarities are given as equation (7), and the two vectors are orthogonal to each other, $\mathbf{x}_1 \cdot \mathbf{x}_2 = 0$ the similarity is given as zero, as expressed in equation (8),

$$\begin{aligned} sim(\mathbf{x}_1, \mathbf{x}_2) &= \frac{\sum_{i=1}^{d} \sum_{j=1}^{d} s_{ij} x_{1i} x_{2j}}{d \cdot \|\mathbf{x}_1\| \cdot \|\mathbf{x}_2\|} \\ &= \frac{\mathbf{x}_1 \cdot \mathbf{x}_2}{d \cdot \|\mathbf{x}_1\| \cdot \|\mathbf{x}_2\|} = \frac{0}{d \cdot \|\mathbf{x}_1\| \cdot \|\mathbf{x}_2\|} = 0 \end{aligned} \tag{8}$$

In the cosine similarity between two vectors, as the traditional metric, the feature similarities are given as equation (7).

C. Proposed Version of KNN

This section is concerned with the proposed version of the KNN algorithm which is shown in Figure 5, as the approach to the keyword extraction. We describe the process of encoding words into numerical vectors in Section III-A, and assume that training examples and a novice item are given as numerical vectors. The similarity metric between numerical vectors which was covered in Section III-B is used for selecting nearest neighbors from the training examples. Variants may be derived from it by considering various selection schemes and voting schemes. This section is intended to describe the proposed version of KNN algorithm and its variants.

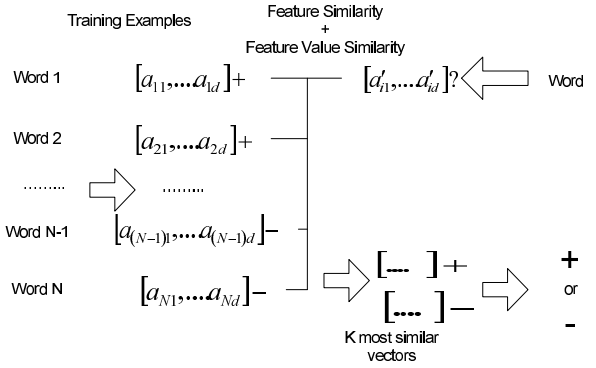

Figure 5. The Proposed Version of KNN

Let us mention the process of selecting nearest neighbors as the references for classifying them. We notate the training examples and a novice item which are mapped into numerical vectors by the process which is described in Section 3.1 by $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ The similarities of the novice item, $\mathbf{x}$, with the training examples,$sim(\mathbf{x}_1, \mathbf{x}), sim(\mathbf{x}_2, \mathbf{x}), \ldots, sim(\mathbf{x}_N, \mathbf{x})$, are computed. 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, $\mathbf{x}$ whose elements are given as numerical vectors and their target labels, by equation (9),

$$Ne_k(\mathbf{x}) = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_k, y_k)\}, \quad y_i \in \{c_1, c_2, \ldots, c_m\} \tag{9}$$

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 $Count(Ne_k(\mathbf{x}), c_i)$. The label of the novice item, $\mathbf{x}$, is decided by the majority of categories in the nearest neighbors, as expressed by equation (10),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Count(Ne_k(\mathbf{x}), c_i) \tag{10}$$

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, $sim(\mathbf{x}, \mathbf{x}_1), sim(\mathbf{x}, \mathbf{x}_2), \ldots, sim(\mathbf{x}, \mathbf{x}_k)$ as weights, $w_1, w_2, \ldots, w_k$ by equation (11),

$$w_i = sim(\mathbf{x}, \mathbf{x}_i) \tag{11}$$

indicates the similarity of a novice item with the ith nearest neighbor. The total weight of nearest neighbors which labeled with the category, c_i by equation (12),

$$Weight(Ne_k(\mathbf{x}), c_i) = \sum_{\mathbf{x}_j \in c_i}^{k} w_j \tag{12}$$

The label of the novice item, $\mathbf{x}$, is decided by the category which corresponds to the maximum sum of weights as shown in equation (13),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Weight(Ne_k(\mathbf{x}), c_i) \tag{13}$$

When the weights of nearest neighbors are set constantly, equation (13) is same to equation (10), as expressed in equation (14),

$$Weight(Ne_k(\mathbf{x}), c_i) = Count(Ne_k(\mathbf{x}), c_i) \tag{14}$$

We described the proposed version of the KNN algorithm as the approach to the keyword extraction. It is assumed that the relationships among features are available in using the proposed version for the classification task. The similarity metric which was defined Section III-B is used for computing similarities of a novice item with training ones. We adopted the rank based selection for selecting the nearest neighbors from the training examples. We used the unweighted voting for decoding the label of a novice item in experiments which are covered in Section IV.

D. Keyword Extraction System

This section is concerned with the keyword extraction system which adopts the KNN algorithm considering the feature similarities. In Section III-C, we described the proposed version of KNN algorithm as the approach to the keyword extraction. The task is viewed as the binary classification where each word is classified into keyword or non-keyword. The proposed version of the KNN algorithm

is applied to the binary classification which is mapped from the keyword extraction. This section is intended to describe the keyword extraction system with respect to its functions and its architecture.

The process of collecting sample words for implementing the keyword extraction system is illustrated in Figure 6. The keyword extraction is viewed as the binary classification where each word is classified into keyword or non-keyword. The topic based word classification belongs to the domain independent classification, whereas the keyword extraction does to the domain dependent classification as shown in Figure 6 where even same word may be classified differently depending on the domain. The words which are labeled with keyword or non-keyword are collected domain by domain. It is necessary to tag the input text with its own domain for executing the keyword extraction.

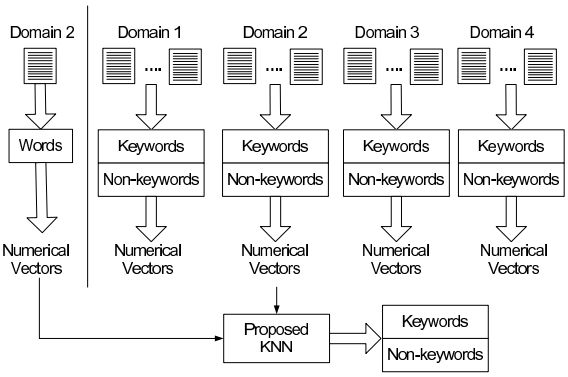

Figure 6. Sample Words

The entire architecture of the proposed keyword extraction system is illustrated in Figure 7. 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 numerical vectors 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.

The execution process of the proposed system is illustrated as the block diagram in Figure 8. The sample words which are labeled with keyword or non-keyword are collected from each domain, and encoded into numerical vectors. The input text is indexed into a list of words and they are also encoded into numerical vectors. The nearest neighbors are selected by the similarity computation 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.

Let us make some remarks on the proposed system which is illustrated in Figure 7 as the architecture. We

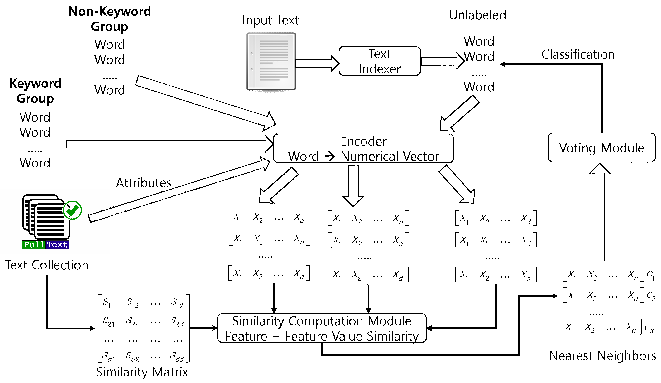

Figure 7. Proposed System Architecture

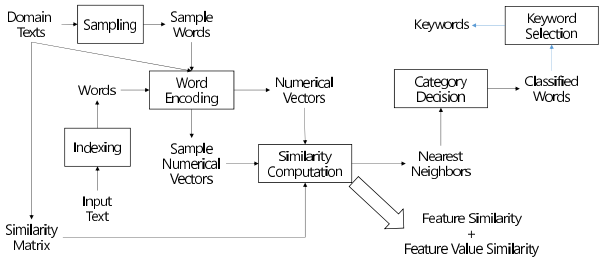

Figure 8. Execution Process of Proposed System

proposed the similarity metric which considers both the feature similarities and the feature value ones. It eliminates the poor discrimination among numerical vectors which is caused by the sparse distribution in each of them. The classification performance is improved by what is proposed in this research, as shown in Section. In the next research, we present the graphical user interface and the source codes which are necessary for implementing the system as a complete one.

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 [5]. 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. 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 9, 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 9 indicates the average over accuracies over the left four groups, and set the input size which is the dimension of numerical vectors as 50.

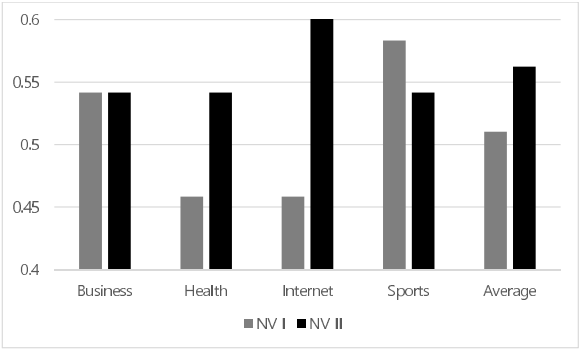

Figure 9. Results from Recognizing Keywords in Text Collection: NewsPage.com

Let us make the discussions on the results from doing the keyword extraction using the both versions of KNN algorithm, as shown in Figure 9. The accuracy which is the performance measure of the classified task is in the range between 0.45 and 0.6. The proposed version of the KNN algorithm works better in the two domains: Health and Internet. It matches with the traditional version in the domain, Business, but is lost 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. 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 10, 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 10, the most right group indicates the averages over results of the left three groups. Therefore, Figure 10 show the results from classifying each group into one of 'keyword' and 'non-keyword', by both versions of KNN algorithm.

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 10. The accuracy values of the both versions range between 0.45 and 0.6. The proposed version works better than the traditional one in one of the three domains: Hotel. It is comparable with the traditional version in the domain, Electronics, and it is leaded in domain, Car. From this set of experiments, we conclude the proposed version works competitively with the traditional version in averaging the three cases.

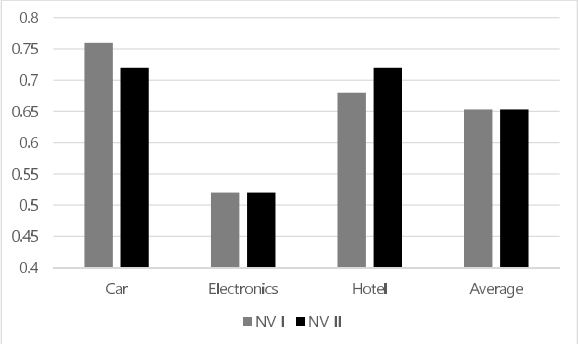

Figure 10. Results from Recognizing Keywords in Text Collection: Opiniopsis

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 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 11, we illustrate the experimental results from deciding whether each word is a keyword or not on the broad version of 20NewsGroups. Figure 11 has the identical frame of presenting the results to those of Figure 9 and 10. 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.

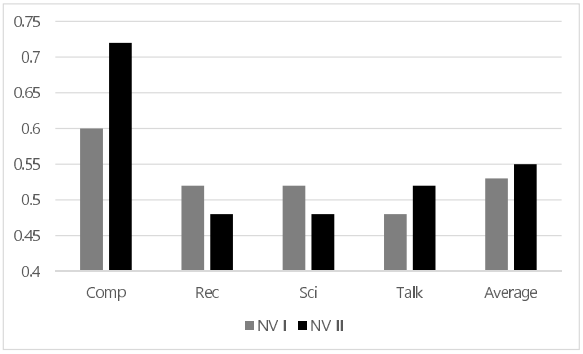

Figure 11. Results from Recognizing Keywords in Text Collection: 20NewsGroup I

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 0.72. The proposed version shows the better performance in the two domains, comp and talk. However, it shows its slightly less performances in the others. From this set of experiments, the proposed version keeps its better performance, in averaging its four achievements.

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. 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 12, 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’.

Let us discuss on the results from doing the keyword extraction on the specific version of 20NewsGroups, as shown in Figure 12. The accuracies of both versions of KNN algorithm range between 0.40 and 0.65. The proposed version shows its better results in the domain, ‘electro’. It shows its comparable one in the two domains, ‘script’ and ‘space’, and its less one in the domain, ‘medicine’. From this set of experiments, it is concluded that the both versions are

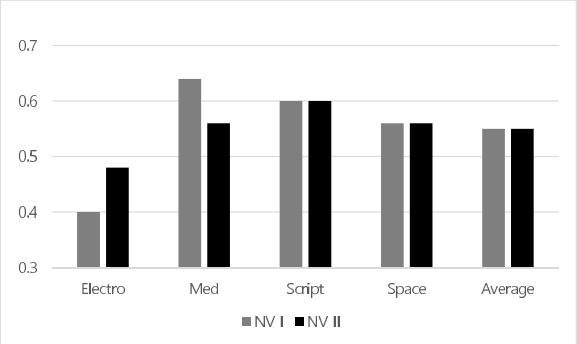

Figure 12. Results from Recognizing Keywords in Text Collection: 20NewsGroup II

comparable with each other.

V. Conclusion

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 two of the four collections and its matching ones in the others. The accuracies of the traditional version range between 0.4 and 0.76 and those of the proposed version range between 0.49 and 0.72. From the four sets of experiments, we conclude the proposed version improved the keyword extraction performance as the contribution of this research.

Let us mention the remaining tasks for doing the further research. We need to validate the proposed approach in specific domains such as medicine, engineering, and economics, as well as in generic domains such as ones of news articles. We may consider the computation of similarities among some main features rather than among all features for reducing the computation time. We try to modify other machine learning algorithms such as Naive Bayes, Perceptrons, and SVM (Support Vector Machine) based on both kinds of similarities. By adopting the proposed approach, we may implement the keyword extraction system as a real program.

References

[1] T. Jo, "The Implementation of Dynamic Document Organization using Text Categorization and Text Clustering" PhD Dissertation of University of Ottawa, 2006.

[2] 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.

[3] T. Jo, "Semantic Numerical Operations on Strings", 55-60, The Proceedings of Fourth International Conference on Future Computational Technologies and Applications, 2012.

[4] T. Jo, "Simulation of Numerical Semantic Operations on Words in Reuter21578", 19-28, The Proceedings of 14th International Symposium on Advanced Intelligent Systems, 2013.

[5] T. Jo, "KNN based Word Categorization considering Feature Similarities", 343-346, The Proceedings of 17th International Conference on Artificial Intelligence, 2015.

[6] T. Jo, "AHC based Clustering considering Feature Similarities", 67-70, The Proceedings of 11th International Conference on Data Mining, 2015.

[7] T. Jo, "Keyword Extraction by KNN considering Feature Similarities", 64-68, The Proceedings of The 2nd International Conference on Advances in Big Data Analysis, 2015.

[8] T. Jo, "Index Optimization with KNN considering Similarities among Features", 120-124, The Proceedings of 14th International Conference on Advances in Information and Knowledge Engineering, 2015.

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

[10] T. Jo, "Simulation of Numerical Semantic Operations on String in Text Collection", 45585-45591, International Journal of Applied Engineering Research, Vol 10, No 24, 2015.

[11] T, Jo, "Graph based KNN for Optimizing Index of News Articles", 53-62, Journal of Multimedia Information System, Vol 3, No 3, 2016.

[12] T, Jo, "Table based K Nearest Neighbor for Word Categorization in News Articles", 1214-1217, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[13] 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.

[14] 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.

[15] T. Jo, "Keyword Extraction in News Articles using Table based K Nearest Neighbors", 1230-1233, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[16] 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.

[17] 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.

[18] T. Jo, "Word Classification in Domain on Current Affairs by Feature Similarity based K Nearest Neighbor", 348-351, The Proceedings of International Conference on Artificial Intelligence, 2018.

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

[20] T. Jo, “Modification of AHC algorithm for Clustering Words into Feature Similarity based Version”, 359-362, The Proceedings of International Conference on Artificial Intelligence, 2018.

[21] T, Jo, “Clustering Texts using Feature Similarity based AHC Algorithm”, 5993-6003, Journal of Intelligent and Fuzzy Systems, Vol 35, 2018.

[22] T. Jo, “Automatic Summarization System in Current Affair Domain by Table based K Nearest Neighbor”, 115-121, The Proceedings of 2nd International Conference on Advanced Engineering and ICT-Convergence, 2019.

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

[24] 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.

[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”, pp913-920, 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”, pp91-98, 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] T. Mitchell, Machine Learning, McGraw-Hill, 1997.