Application of String Vector based K Nearest Neighbor to Semantic Word Classification

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 string vector as its input data and is applied to the word categorization. The results from applying the string vector based algorithms to the text categorizations were successful in previous works and synergy effect between the text categorization and the word categorization is expected by combining them with each other; the two facts become motivations for this research. In this research, we define the operation on string vectors called semantic similarity, modify the KNN algorithm by replacing the exiting similarity metric by the proposed one, and apply it to the word categorization. The proposed KNN is empirically validated as the better approach in categorizing words in news articles and opinions. We need to define and characterize mathematically more operations on string vectors for modifying more advanced machine learning algorithms.**

***Keywords*-Word Categorization; String Vector; K Nearest Neighbor**

I. Introduction

Word categorization refers to the process of classifying words into a particular category or relevant categories as an instance of classification task. As its preliminary task, a list of categories is predefined and words labeled with one of the predefined categories are prepared as the sample data. The labeled words are encoded into their structured forms and the classification capacity is constructed by learning them. A novice word is encoded into its structured form, and classified into one of the predefined categories. In this research, we assume that the supervised learning algorithms are used as the approach to the word categorization, even if other types of approaches are available.

Let us mention some challenges with which this research tries to tackle. Many features are required in encoding words into numerical vectors for using the traditional classifiers in order to keep the robustness [29]. The numerical vectors which represent words tend to be sparse; zero values are usually dominant in each numerical vector [2][25]. In previous works, texts and words were encoded into tables as alternative representations to numerical vectors, but it is very expensive to compute the similarity between tables [2][25]. Hence, in this research, we try to solve the problems by encoding words into string vectors.

Let us mention what is proposed in this research as its ideas. We encode words into string vectors whose elements are text identifiers as alternative representations to numerical vectors. We define the operation on string vectors which corresponds to the cosine similarity between two numerical vectors. We modify the KNN (K Nearest Neighbor) into the string vector based version and apply it to the word categorization. Hence, in this research, the words are categorized based on their semantic relations by the KNN.

Let us mention some benefits which are expected from this research. The string vectors may become more representative representations of words than numerical vectors; the size of each string vector is smaller than that of each numerical vector. The discriminations among string vectors are much better than among numerical vectors, since the sparse distribution is not available in each string vector. We expect the better classification performance from this research by solving the problems caused by encoding words into numerical vectors. Therefore, the goal of this research is to implement the word categorization systems with the benefits.

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 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. Word Categorization and its Derived Tasks

This section is concerned with the cases of using the modern type of KNN algorithm to the tasks in the word categorization class. We mention the three tasks: topic based word classification, keyword extraction, and index optimization. We mention the KNN algorithm where each word is encoded into structured data alternative to the table. The topic based word categorization is to assign one or some among the predefined topics to each word, whereas the keyword extraction and the index optimization are classification tasks based on the word importance in the given text. This section is intended to present the cases of applying the modern types of KNN algorithms to the word classification tasks.

Let us present the cases of using the modernized KNN algorithm for the topic based word classification. The similarity among features was considered as well as feature values in using the KNN algorithm for the word classification [9]. Words were encoded into tables, instead of numerical vectors, in using the KNN for the word categorization [10]. The KNN was modified into the modernized version which receives the graph as the input vector, as the approach to the word categorization [11]. In the above literatures, the KNN was modified into the modernized version which receives a table, or a graph, instead of a numerical vector.

The keyword extraction was mapped into the binary classification of words: classifying a word into keyword or non-keyword. The KNN algorithm which considers the similarities among features was applied to the keyword extraction by mapping it so [12]. The trial of applying the KNN algorithm which classifies a table, instead of a numerical vector to the task, existed previously [13]. The KNN algorithm which classifies a graph directly was used for extracting keywords from a text, automatically [14]. The fact that the keyword extraction is a special type of word categorization is the reason of mentioning it in this study.

Let us derive the task, index optimization, from the word categorization and mention cases of applying the modernized versions of KNN algorithm to the task. The modernized KNN algorithm which considers the feature similarities was applied to the index optimization in [15]. The process of applying the modernized KNN version which processes tables directly to the index optimization was mentioned in [16]. The case of applying another modernized version which processes graphs directly to the index optimization was presented [7]. The index optimization in the above literatures was viewed as the classification of words which are indexed from a text into expansion, inclusion, or removal.

Let us mention some distinguished points of this research, from the above works. We explored the previous works where the modernized versions of KNN algorithm were applied to the word categorization and its derived tasks. We mentioned the three modernized versions of KNN algorithm: the version which considers the similarities among features in computing the similarity between two vectors, the version which classifies a table directly, and the version which processes graphs directly. The KNN version which is proposed in this research, receives a string vector directly, instead of a numerical vector. We will use it for classifying a word semantically into one among predefined topics.

B. Word and Text Encoding

This section is concerned with the survey on the previous cases of encoding words or texts into non-numerical vectors. Some issues such as huge dimensionality and sparse distribution in each numerical vector were pointed out in encoding words or texts into numerical vectors. There were trials to solve the above issues by encoding texts or words into alternative structured forms. In this research, we mention the tables, string vectors, and graphs as alternative structured data. This section is intended to mention the previous cases of encoding texts or words into non-numerical vectors in other tasks than ones which were mentioned in the previous section.

Let us mention the previous cases of encoding texts or words into tables. In using the AHC algorithm for clustering words, words were encoded into tables [17]. Texts were encoded into tables for modifying the KNN algorithm as the approach to the text categorization [18]. Texts were encoded so for modernizing the AHC algorithm as the approach to the text clustering [21]. The KNN algorithm and the AHC algorithm which are presented in the above works process tables directly.

Let us mention 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 semantically [19]. Texts were encoded into string vectors in using the KNN algorithm for classifying texts [20]. Texts were encoded so in using the AHC algorithm for clustering texts [22]. The cases of encoding texts or words into string vectors instead of numerical vectors in the above literatures.

Let us consider mapping words or texts into graphs. Words were encoded into graphs in using the AHC algorithm for clustering words [8]. Texts were encoded into graphs in using the KNN algorithm for classifying texts [23]. Texts were encoded so in using the AHC algorithm for clustering texts [24]. In the above literatures, we present cases of mapping raw data into graphs.

Let us mention some points of this research which are relevant to ones in the literatures mentioned, above. We adopt the scheme where words are encoded into string vectors. We define the similarity metric between two string vectors as the operation on them, and modify the KNN algorithm into the version which process string vectors directly based on the operation. We apply the modernized version of KNN algorithm for implementing the topic based word categorization. We validate the modernized KNN algorithm

in the word categorization in the four test sets, comparing with the traditional one.

C. Non-Numerical Vector based Machine Learning Algorithms

This section is concerned with the previous works on non-numerical vector based machine learning algorithms. In the previous section, we explored the cases of encoding words or texts into tables, string vectors, or graphs, in using the KNN algorithm or the AHC algorithm. In this section, we will mention the three machine learning algorithms which process other structured data as the approaches to the text categorization. Among them, the string kernel based SVM computes lexical similarity between raw texts rather than encoding texts into numerical vectors. This section is intended to survey previous works on machine learning algorithms which process non-numerical vectors.

Let us consider the string kernel based SVM(Support Vector Machine) where the lexical similarity between raw texts is computed, as the approach to the text classification. The version of SVM was initially proposed as a text classification tool by Lodhi et al. in 2002 [28]. It was used for the protein classification by Leslie et al. in 2004 [27]. It was applied to the sentence classification by Kate and Mooney in 2006 [26]. The string kernel based SVM was useful for classifying short texts or sentences, rather than long texts.

Let us mention the table based matching algorithm as another type of approach to the text categorization. It was initially proposed by Jo and Cho in 2008 [25]. It was applied to the soft categorization of texts where more than one category is allowed to be assigned to each text [2]. It was improved into the more robust and stable version in 2015 [5]. In using the table based matching algorithm which was mentioned in the above literatures, texts are encoded into tables.

Let us mention the Neural Text Categorizer, as the neural network model which is specialized for the text categorization. It was initially proposed as the approach to the text categorization by Jo in 2008 [3]. It was empirically validated in both soft categorization and hard categorization as the better approach than main approaches such as the Naive Bayes and the SVM [4]. It was used for classifying texts in Arabian by Abainia et al. in 2015 [1]. It was mentioned as an innovative approach by Vega and Mendez-Vazquez in 2016 [30].

Let us mention the three classification algorithms which deals with non-numerical vectors from the above literatures. The string kernel based SVM processes raw texts by themselves, the table matching algorithm processes tables, and the Neural Text Categorizer processes string vectors. In this research, words are encoded into string vectors whose elements are text identifiers. The KNN algorithm is modified into the version which processes string vectors, as the approach to the word categorization. The modified version will be compared with the traditional one in classifying words.

III. Proposed Approach

This section is concerned with encoding words into string vectors, modifying the KNN (K Nearest Neighbor) into the string vector based version and applying it to the word categorization, and consists of the three sections. In Section III-A, we deal with the process of encoding words into string vectors. In Section III-B, we describe formally the similarity matrix and the semantic operation on string vectors. In Section III-C, we do the string vector based KNN version as the approach to the word categorization, and in Section III-D, present the architecture of the system which we try to implement by adopting the proposed KNN. Therefore, this section is intended to describe the proposed KNN version as the word categorization tool.

A. Word Encoding

This section is concerned with the process of encoding words into string vectors. We present the previous works on string vector based machine learning algorithms as the cases of doing so in Section II-B and II-C. A word is encoded into a string vector as shown in Figure 1, 2, and 3, with the three steps: feature definition, feature matching analysis, and text identifier assignment. Its elements are given as text identifiers which are related with the word, in the string vector representing it. This section is intended to describe each of three steps of encoding words so.

The features for encoding words into string vectors are illustrated in Figure 1. It is assumed that the dimension of string vector which represents a word is d and the first paragraph is usually the key part of text. We define the four subgroups of features of the string vector: the frequency in the entire text, the TF-IDF (Term Frequency and Inverse Document Frequency) weight in the entre text, the frequency in the first paragraph, and the TF-IDF in the first paragraph. The quarter of dimension is assigned to each subgroup, and the frequency or the weight is ranked until $d/4$. It is manual to define the features of string vectors shown in Figure 1 in this research.

The process of analyzing feature matching is illustrated as the pseudo code in Figure 2. The features of string vectors are defined as illustrated in Figure 1 and one among features is given as an argument. An individual feature indicates a relationship between a text and a word, so for each text, relationship of word is generated. If the current relationship matches with the feature which is given as an argument, the current text is returned as the feature value. The feature may be viewed as a function which is given as an argument.

The process of assigning text identifiers according to the defined features as elements of string vectors. The features are defined as illustrated in Figure 1. By the process which is shown in Figure 2, text identifiers are assigned to their

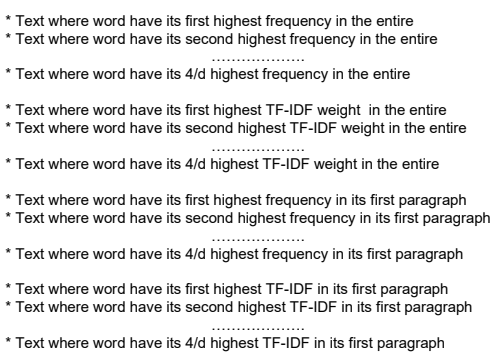

Figure 1. Defined Features

```
searchTextID(List textIDList, Feature featureItem, Word wordItem){
    for each textID in textIDList
        if isMatch(textID, featureItem, wordItem)
            return textID;
```

Figure 2. Feature Matching Analysis

own positions in the string vector. The string vector which is an ordered finite set of strings is generated as the word representation. Each string indicates an identifier of text which corresponds to its own feature.

A word is encoded into a string vector with the three steps which are presented in Figure 1, 2, and 3. A string vector is defined as an ordered finite set of strings; that fact that elements are given as strings instead of numerical values is the difference from a numerical vector. The features of each string vector are defined as relationships between a word and a text. A text identifier is given as feature value which corresponds to its own feature of a word. We need to define the operations on string vectors for modifying the machine learning algorithm into the version which processes them directly.

B. Semantic Similarity between String Vectors

This section is concerned with the similarity metric between two string vectors. In the previous section, we mentioned the process of transforming words into string vectors. We need to define the similarity metric between two string vectors as the operation on them, for modifying the KNN algorithm into the version which processes string vectors, directly. We introduce the concept of semantic operation and define the semantic similarity between two strings as the basis. This section is intended to describe the semantic similarity between two string vectors.

The semantic operations refers to the operations on strings based on their meanings. They were initially proposed as ones on strings by Jo in 2015 [6]. Every string is assumed to have its own meaning and the operations may be defined based on their meanings. In [6], the semantic similarity of two strings, the average semantic similarity over strings, and the average semantic variance were defined and simulated in text collections in their various domains. The first operation was adopted for modifying the KNN algorithm as the approach to the word categorization, in this research.

In Figure 4, the semantic similarity matrix between two texts is illustrated. The two texts are notated by d_i and d_j and the similarity between them is notated by $sim(d_i, d_j)$. The two texts, d_i and d_j, are expressed as the two sets of words, D_i and D_j, and $|D_i|$ and $|D_j|$ are cardinalities of the two sets. The similarity between two texts is computed

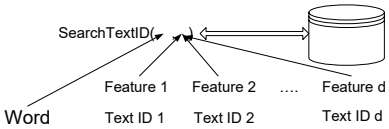

Figure 3. Text Identifier Assignment

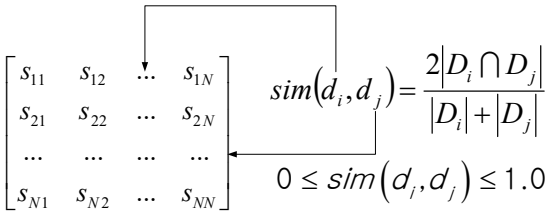

Figure 4. Similarity Matrix

by equation (1),

$$sim(d_i, d_j) = \frac{2|D_i \cap D_j|}{|D_i| + |D_j|} \tag{1}$$

and the similarity is always given as a normalized value between zero and one. The rows and the columns of the matrix which is presented in Figure 4, correspond to texts in the corpus, and each element becomes the similarity between corresponding texts.

A string vector is defined as an ordered finite set of strings as shown in equation (2),

$$\mathbf{str} = [str_1, str_2, ..., str_d] \tag{2}$$

The two string vectors are notated by equation (3) and (4),

$$\mathbf{str}_1 = [str_{11}, str_{12}, ..., str_{1d}] \tag{3}$$

$$\mathbf{str}_2 = [str_{21}, str_{22}, ..., str_{2d}] \tag{4}$$

The similarity between the two string vectors is defined as average over semantic similarities of one to one elements, as shown in equation (5),

$$sim(\mathbf{str}_1, \mathbf{str}_2) = \frac{1}{d} \sum_{i=1}^{d} sim(str_{1i}, str_{2i}) \tag{5}$$

The string vector which represents a word consists of text identifiers and the value of $sim(str_{1i}, str_{2i})$ is looked up from the similarity matrix which is presented in Figure 4. The similarity between the two string vectors, $\mathbf{str}_1$ and $\mathbf{str}_2$ is always given as a normalized value between zero and one.

We mentioned the similarity between two string vectors as a normalized value between zero and one. If the two string vectors are exactly same to each other as shown in equation (6),

$$\mathbf{str}_1 = \mathbf{str}_2 \tag{6}$$

the semantic similarity between them is 1.0 as shown in equation (7),

$$sim(\mathbf{str}_1, \mathbf{str}_2) = sim(\mathbf{str}_1, \mathbf{str}_1) = \frac{1}{d} \sum_{i=1}^{d} sim(str_{1i}, str_{1i}) = 1.0 \tag{7}$$

If the semantic similarities between elements of two string vectors are zeros, the sematic similarity between them is 0.0

as shown in equation (8),

$$sim(\mathbf{str}_1, \mathbf{str}_2) = \frac{1}{d}\sum_{i=1}^{d} sim(str_{1i}, str_{1i}) = \frac{0}{d} = 0.0 \quad (8)$$

Because $0 \leq sim(\mathbf{str}_1, \mathbf{str}_2) \leq 1$ the semantic similarity between them is always given as a normalized value between zero and one by equation (9),

$$\begin{aligned} 0 \leq sim(\mathbf{str}_1, \mathbf{str}_2) \leq 1 \\ 0 \leq \frac{1}{d}\sum_{i=1}^{d} sim(str_{1i}, str_{2i}) \leq 1 \end{aligned} \quad (9)$$

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

C. Proposed Version of KNN

The proposed version of KNN algorithm as the approach to the word categorization is illustrated in Figure 5. We described the process of encoding words into string vectors in Section III-A, and assumed that training examples and a novice item are given as string vectors. In the proposed version, the similarity metric between string vectors which was covered in Section III-B is used for selecting nearest neighbors. The label of a novice item is decided by voting ones of nearest neighbors and variants may be derived by modifying voting schemes in addition. This section is intended to describe the proposed version of KNN algorithm which deals with string vectors directly, and its variants.

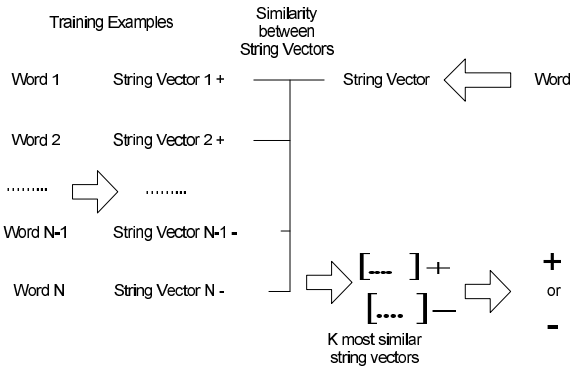

Figure 5. Proposed KNN Algorithm

Let us mention the process of selecting nearest neighbors among training examples as the references for deciding the label to a novice item. The training examples and the novice item are mapped into string vectors by the process which was described in Section III-A. The similarities of the novice item with the training examples are computed by the process which was described in Section III-B. The training examples are ranked by their similarities and the K similar ones are selected as the nearest neighbors. We adopt the rank based scheme in selecting nearest neighbor 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, $\mathbf{str}$, whose elements are given as tables and their target labels, by equation (10),

$$\begin{aligned} Ne_k(\mathbf{str}) = \{(\mathbf{str}_1, y_1), (\mathbf{str}_2, y_2), \ldots, (\mathbf{str}_k, y_k)\}, \\ y_i \in \{c_1, c_2, \ldots, c_m\} \end{aligned} \quad (10)$$

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 $Count(Ne_k(\mathbf{str}), c_i)$. The label of the novice item, $\mathbf{str}$, is decided by the majority of categories in the nearest neighbors, as expressed by equation (11),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}}\, Count(Ne_k(\mathbf{str}), c_i) \quad (11)$$

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, $sim(\mathbf{str}, \mathbf{str}_1), sim(\mathbf{str}, \mathbf{str}_2), \ldots, sim(\mathbf{str}, \mathbf{str}_k)$ as weights, $w_1, w_2, \ldots, w_k$ by equation (12),

$$w_i = sim(\mathbf{str}, \mathbf{str}_i) \quad (12)$$

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 (13),

$$Weight(Ne_k(\mathbf{str}), c_i) = \sum_{\mathbf{str}_j \in c_i}^{k} w_j \quad (13)$$

The label of the novice item, $\mathbf{str}$, is decided by the category which corresponds to the maximum sum of weights as shown in equation (14),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}}\, Weight(Ne_k(\mathbf{str}), c_i) \quad (14)$$

When the weights of nearest neighbors are set constantly, equation (14) is same to equation (11), as expressed in equation (15),

$$Weight(Ne_k(\mathbf{str}), c_i) = Count(Ne_k(\mathbf{str}), c_i) \quad (15)$$

We described the proposed version of the KNN algorithm in this section. In using the proposed KNN algorithm, raw data is encoded into string vectors, 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. Word Classification System

This section is concerned with the word classification system which adopts the string vector based KNN algorithm. In Section III-C, we described the KNN algorithm which processes string vectors directly. The preliminary tasks in this system are to predefine categories as a list and to gather sample labeled words. Words are encoded into string vectors and classified the KNN algorithm which was described in Section III-C. This section is intended to describe the word classification system with respect to its architecture.

The sample words are illustrated for implementing the topic based word classification by the proposed KNN algorithm in Figure 6. The topics are predefined as topic 1, topic 2, ..., topic M. The N words are allocated for each topic as the sample words. The balanced distribution over the categories is necessary for preventing the bias toward a particular topic. $M \times N$ sample words are encoded into string vectors by the process which is mentioned in Section III-A.

The entire architecture of the proposed word categorization system is illustrated in Figure 7. The sample words which are labeled with one of M categories and the unlabeled ones as novice items are encoded into string vectors. For each novice string vector, its similarities with the sample string vectors are computed by the metric which is mentioned in Section III-B, in the similarity computation module, and the k most similar sample ones are selected as the nearest neighbors. The label of the novice item is decided by voting ones of nearest neighbors in the voting module. This system consists of the three components: the encoding module, the similarity computation module, and the voting module.

The execution process of the proposed system is illustrated in Figure 8. The sample words which are collected by the process mentioned above and the word which is given as the input are encoded into string vectors. Its nearest neighbors are extracted from the samples through the similarity computation module. The category of the novice word is decided by voting ones of the nearest neighbors. The category of the novice word is decided as the final output in the system.

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. Words are encoded into string vectors, instead of numerical vectors. String vectors which represent novice words are classified directly by the proposed KNN algorithm. The classification performance is improved by what proposed in this research, as shown in Section IV. In the next research, we present

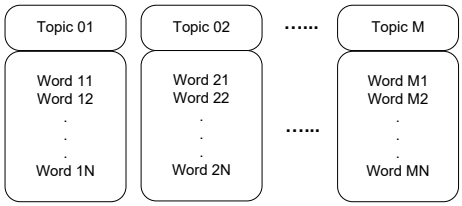

Figure 6. Sample Words

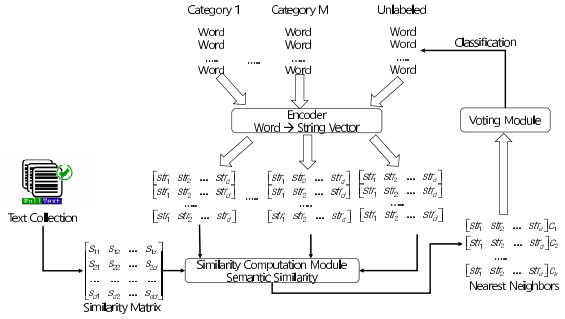

Figure 7. Proposed System Architecture

the graphical user interface and the source code 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 word categorization on the collection, NewsPage.com. In Section IV-B, we show the results from applying it for categorizing

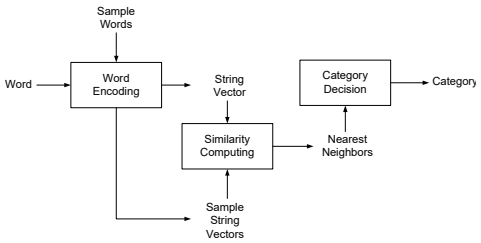

Figure 8. Execution Process of Proposed System

words 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 categorizing words 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. The four categories are predefined in this collection and from the collection, NewsPage.com, we gathered the words category by category as the labeled ones. Each word is allowed to be classified into only one of the four categories. In this set of experiments, we apply the traditional and proposed version of KNN to the classification task, without decompose it into binary classifications, and use the accuracy as the evaluation measure. In this section, we observe the performance of the both versions of KNN, by changing the input size.

In Table I, we specify NewsPage.com, which is the text collection as the source for extracting classified words in this set of experiments. The text collection was used in the previous works for evaluating approaches to text categorization [5]. In each category, we extract 375 important words for building the collection of labeled words for evaluating the approaches to word categorization. In each category, the set of 375 classified words is partitioned into the 300 words as training examples and the 75 words as test examples, as shown in Table I. We select words by their frequencies concentrated in the given category combined with subjectivity in building the word collection.

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

Category	#Texts	#Training Words	#Test Words
Business	500	300	75
Health	500	300	75
Internet	500	300	75
Sports	500	300	75
Total	2000	1200	300

Let us mention the empirical process for validating the proposed approach to the task of word categorization. We extract the important words from each category in the above text collection, and encode them into numerical vectors. For each text example, the KNN compute its similarities with the 1200 training examples by the cosine similarity, and select the three most similar training examples as its nearest neighbors. Each of the 300 test examples is classified into one of the four categories: Business, Sports, Internet, and Health, by voting the labels of its nearest neighbors. The classification accuracy is computed by dividing the number of correctly classified test examples by total number of test examples, for evaluating the both versions of KNN.

Figure 9 illustrates the experimental results from categorizing the words 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 is the input size as the dimension of numerical and string vectors which represent texts. In each group, the gray and black bar indicate the performance of the traditional and proposed version of KNN algorithm, respectively. The most right group indicates the average over accuracies of the left four cases.

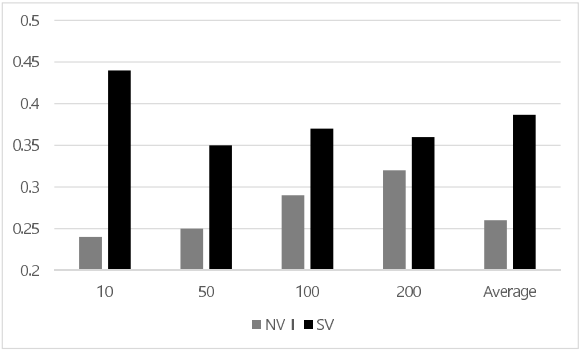

Figure 9. Results from Classifying Words in Text Collection: NewsPage.com

Let us make discussions on the results from doing the word categorization, using the both versions of KNN algorithm, as shown in Figure 9. The accuracy which are the performance measure of this classification task is in range between 0.24 and 0.44. The proposed version of KNN algorithm works better in the all input sizes; the accuracy of the proposed version reaches more than 0.4, in the input size

10. As the input size increases, the performance difference between both versions decreases; the performance of the traditional version improves proportional to the input size, but one of the proposed version stays around 0.35, except the input size 10. In this set of experiments, we conclude that the proposed version works outstandingly 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. In this set of experiments, the three categories are predefined in the collection, and we gather words category by category as the classified ones. Each word is classified exclusively into one of the three categories. The given classification is not decomposed into binary classifications and the accuracy is used as the evaluation measure. In this section, we observe the performances of the both versions of KNN algorithm with the different input sizes in the collection, Opniopsis.

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 375 important words from each category as the collection of the classified words for evaluating the approaches to word categorization. In each category, as shown in Table 2, we partition the set of words into the 300 words as the training set and the 75 words as the test set. We select the words from the collection, depending on their frequencies which are concentrated on their own categories.

Table II
THE NUMBER OF TEXTS AND WORDS IN OPINIOPSIS

Category	#Texts	#Training Words	#Test Words
Car	23	300	75
Electronic	16	300	75
Hotel	12	300	75
Total	51	900	225

We perform this set of experiments by the process which is described in section IV-A. We extract the 300 important words by scanning individual texts in each category, and encode them into numerical vectors and string vectors, with the input sizes: 10, 50, 100 and 200. For each test example, the both versions of KNN computes its similarities with the 900 training examples and select the three most similar training examples as its nearest neighbors. Each of the 225 test examples is classified into one of the three categories, by voting the labels of its nearest neighbors. 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 categorizing the words using the both versions of KNN on this collection. Like Figure 9, the y-axis indicates the accuracy and the x-axis does the group of two versions by an input size. In each group, the grey bar and the black bar indicate the results of the traditional version and the proposed version of KNN algorithm, respectively. In Figure 2, the most right group indicates the average over results of the left three groups. Therefore, Figure 10 presents the results from classifying the words into one of the three categories by both versions of KNN algorithm, on the collection, Opinosis.

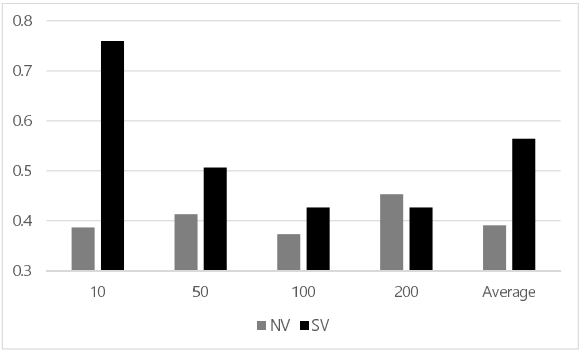

Figure 10. Results from Classifying Words in Text Collection: Opiniopsis

We discuss the results from doing the word categorization using the both versions of KNN algorithm, on Opinosis, shown in Figure 10. The accuracies of the both versions range between 0.35 and 0.75 in this task. The proposed version works better than the traditional one in the three input sizes: 10, 50, and 100. It is comparable with the traditional version in the other: 200. From this set of experiments, we conclude that the proposed one works outstandingly better in averaging over the four cases.

C. 20NewsGroups I: General Version

This section is concerned with one more set of experiments where the better performance of the proposed version is validated empirically on the text collection: 20NewsGroups I. In this set of experiments, we predefine the four general categories, and gather words from the collection category by category as the classified ones. Each word is classified exclusively into one of the four categories. We apply the KNN algorithms directly to the given task without decomposing it into binary classification, and use the accuracy as the evaluation measure. Therefore, in this section, we observe the performance of the both versions of KNN algorithm, with the different input sizes.

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 375 important words from them as the labeled words. The 375 words are partitioned into the 300 words as the training examples and the 75 words as the test ones, as shown in Table III. In the process of gathering the classified words, they are selected by their frequencies which are concentrated in their corresponding categories.

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

Category	#Texts	#Training Words	#Test Words
Comp	1000	300	75
Rec	1000	300	75
Sci	1000	300	75
Talk	1000	300	75
Total	4000	1200	300

The experimental process is identical is that in the previous sets of experiments. In each category, we extract the 375 important words and encode them into numerical and string vectors with the input sizes, 10, 50, 100, and 200. For each test example, we compute its similarities with the 1200 training examples, and select the three similar ones as its nearest neighbors. The versions of KNN algorithm classify each of 300 test examples into one of the four categories: comp, rec, sci, and talk, by voting the labels of its nearest neighbors. We also use the classification accuracy as the evaluation measure in this set of experiments.

In Figure 11, we illustrate the experimental results from categorizing words using the both versions on the broad version of 20NewsGroups. Figure 11 has the identical frame of presenting the results to those of Figure 1 and 2. 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. The performance is expressed as the accuracy of classifying words into one of the four categories. In this set of experiments, the classification task is not decomposed into binary classifications.

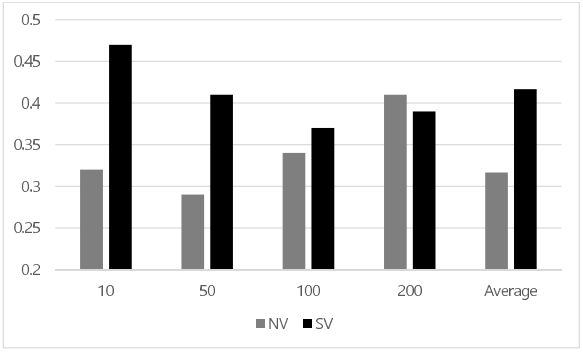

Figure 11. Results from Classifying Words in Text Collection: 20NewsGroup I

Let us discuss the results from doing the word categorization using the both versions on 20NewsGroups as shown in Figure 11. The accuracies of the both versions range between 0.28 and 0.47. The proposed version of KNN algorithm shows its better performances in the three of the four cases, but slightly less performance in the other. The inconsistent entries and the noisy values are the causes of degrading the performance of the proposed version, in the input size, 200. From this set of experiments, we conclude that the proposed version wins over the traditional one, in averaging over their four achievements, in spite of that.

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 different version of 20NewsGroups. In this set of experiments, the four specific categories are predefined and words are gathered from each topic as the classified ones. Like the previous section, each word is exclusively classified into one of the four categories. The two versions of KNN are applied directly to the classification task, without decomposing it into binary classifications, and we use the accuracy as the evaluation metric. Therefore, in this section, we observe the performances of the both versions with the different input sizes, in the specific version of 20NewsGroups.

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. We build the collection of labeled words by extracting the 375 important words from approximately 1000 texts in each specific category. In each category, the set of 375 words is partitioned into the training set with 300 words and the test set with 75 words. In building the test collection, the words are classified by process which is identical to that in the previous set of experiments.

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

Category	#Texts	#Training Words	#Test Words
Electro	1000	300	75
Medicine	1000	300	75
Script	1000	300	75
Space	1000	300	75
Total	4000	1200	300

The process of doing this set of experiments is same to that in the previous sets of experiments. We extract the identical number of words from all texts in each category, and encode them into numerical vectors and string vectors with their identical input sizes. We use the two versions of KNN algorithm for their comparisons. By the two versions, each of test examples is classified into one of the four specific categories which exist within the general category, 'sci': 'electro', 'medicine', 'script', and 'space'. We use

the classification accuracy as the evaluation metric, like the previous sets of experiments.

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. The words are classified directly to one of the four categories like the cases in the previous sets of experiments.

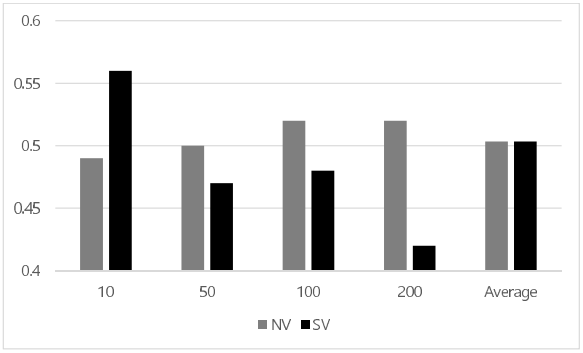

Figure 12. Results from Classifying Words in Text Collection: 20NewsGroup II

Let us discuss the results from classifying the words using the both versions of KNN algorithm on the specific version of 20NeewsGroups, as shown in Figure 12. The accuracies of both versions range between 0.41 and 0.56. The proposed version shows its better results in one of the four cases and its comparable ones in two. However, it is leaded by the traditional version in the input size, 200. From this set of experiments, we conclude that the proposed version is comparable to the traditional one by averaging over the accuracies of the four cases.

V. Conclusion

Let us discuss the entire results from classifying words using the two versions of KNN algorithm. We compare the two versions with each other in the four collections. The proposed versions show its better results in all of the three collections. On the four collections, the accuracies of the traditional version range between 0.24 and 0.52, while, those of the proposed version range between 0.35 and 0.52. Finally, through the three sets of experiments, we conclude that the proposed version of KNN algorithm improves the word categorization performance, as the contribution of this research.

Let us mention the remaining tasks for doing the further research. The proposed approach should be validated and specialized in the specific domains: medicine, engineering and economics. Other features such as grammatical and posting features may be considered for encoding words into string vectors as well as text identifiers. Other machine learning algorithms as well as the KNN may be modified into their string vector based versions. By adopting the proposed version of the KNN, we may implement the word categorization system as a real 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] 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, “Neural Text Categorizer for Exclusive Text Categorization”, 77-86, Journal of Information Processing Systems, Vol 4, No 2, 2008.

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

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

[6] 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.

[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”, pp67-78, Journal of Multimedia Information Systems, 2018.

[10] 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.

[11] T. Jo, “K Nearest Neighbor specialized for Word Categorization in Current Affairs by Graph based Version”, pp64-65, 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, “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.

[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, “Optimizing Index of News Articles by Table based Version of K Nearest Neighbors”, 1214-1217, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 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] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, “Mismatch String Kernels for Discriminative Protein Classification”, 467-476, Bioinformatics, Vol 20, No 4, 2004.

[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] F. Sebastiani, “Machine Learning in Automated Text Categorization”, 1-47, ACM Computing Survey, Vol 34, No 1, 2002.

[30] 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.