Journal of Multimedia Information System VOL. 5, NO. 2, June 2018 (pp. 67-78): ISSN 2383-7632(Online)
http://dx.doi.org/10.9717/JMIS.2018.5.2.67

Semantic Word Categorization using Feature Similarity based K Nearest Neighbor

Taeho Jo[1*]

Abstract

This article proposes the modified KNN (K Nearest Neighbor) algorithm which considers the feature similarity and is applied to the word categorization. The texts which are given as features for encoding words into numerical vectors are semantic related entities, rather than independent ones, and the synergy effect between the word categorization and the text categorization is expected by combining both of them with each other. In this research, we define the similarity metric between two vectors, including the feature 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. The significance of this research is to improve the classification performance by utilizing the feature similarities.

Key Words: Feature Similarity, Feature Value Similarity, Word Categorization, K Nearest Neighbor

I. INTRODUCTION

I. INTRODUCTION

The word categorization refers to the process of classifying the words into some or one of the predefined categories. As its preliminary task, we must define a list of topics as the classes, and allocates sample words to each topic. The sample words are encoded into numerical vectors as the preprocessing step, and the classification capacity is built by learning the labeled words. Afterward, the novice words are also encoded so, and they are classified into one or some of the predefined topics. The scope of this research is restricted to the classification of words by their topics; the POS tagging as a kind of word classification is set out of this research.

Let us consider some challenges with which this research tries to tackle. Previously, the dependencies among features were discovered, but it requires very complicated analysis for considering them [25]. The assumption of independencies among features for the simplicity causes requirement of many features for the robustness. Because each feature has very little coverage, we cannot avoid the sparse distribution where zero values are dominant with more than 95% in each numerical vector [7]. Therefore, this research is intended to solve the problems by considering the feature similarities as well as the feature value similarities.

Let us mention what we propose in this research as its idea. In this research, we consider the both feature similarity and feature value similarity for computing the similarity between numerical vectors. The KNN (K Nearest Neighbor) is modified into the version which accommodates the both similarity measures. We apply the modified version to implementing the word categorization system. Therefore, the goal of this research is to improve the word categorization performance, using smaller number of features by solving the above problems.

Let us mention the contributions of this research as follows:

Combination of feature value and feature similarities mitigates the poor discriminations among sparse numerical vectors.

The process of computing the feature similarity considers the fact that the features are similar among each other in context of text mining tasks.

The word categorization performances are improved by using the proposed similarity for modifying the KNN as shown in Section V.

The dimension of numerical vectors may be potentially reduced, maintaining the reliability.

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

Manuscript received February 27, 2018 ; Revised May 09, 2018 ; Accepted May 14, 2018. (ID No. JMIS-2018-0010)
Corresponding Author (*): Taeho Jo, 2836, Sejongro, Sejong-si, Republic of Korea 30016, +82-044-860-2125, tjo018@hongik.ac.kr
School of Game, Hongik University, Sejong, South Korea, tjo018@hongik.ac.kr

validate empirically the proposed approach by comparing it with the traditional one. In Section V, we mention the significances of this research and the remaining tasks as the conclusion

II. PREVOUS WORKS

This section is concerned with the previous works which are relevant to this research. In Section 1, we explore the previous cases of applying the KNN algorithm to text mining tasks. In Section 2, we survey the schemes of encoding texts or words into structured data. In Section 3, 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.

1. 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 mention the previous cases of applying the KNN algorithm to the word categorization and its similar tasks. In 2001, Kim et al. translated words using the KNN algorithm between English and Korean [21]. In 2003, Pekar and Staab classified words into their synonyms, using the KNN algorithm [27]. In 2016, Stauffer et al. represented optimal images of handwritten words into graphs and applied the KNN to the word recognition [29]. The word categorization in this research is the task of classifying words based on their topics or meaning; it should be distinguished from ones which are mentioned in the above literatures.

Let us mention the previous cases of applying the KNN algorithm to the text categorization. In 2001, Han et al. proposed the modified version of KNN which considers the feature importance for computing the similarity between numerical vectors [4]. In 2010, Khan et al. reviewed machine learning approaches including the KNN algorithm to the text categorization [19]. In 2014, Vishwanath et al. proposed the KNN version which computes the similarity between a test document and a class prototype and the Naive Bayes, as the approaches to text categorization [30]. Even recently, texts are still encoded into numerical vectors in using the KNN for the text categorization, in above literatures.

The spam mail filtering refers to a particular text categorization where each email is classified into spam or ham. In 2003, James et al. proposed the neural networks as the approach to the spam mail filtering and compared it with the KNN algorithm as the base approach [5]. In 2004, Lai et al. proposed the four machine learning algorithms including the KNN as the email categorization tools [22]. In 2010, Frite et al. used the KNN algorithm for the spam mail filtering with resampling methods [3]. In the above literatures, emails are regarded as texts and encoded into numerical vectors.

We survey the cases of using the KNN for the text mining tasks: word categorization, text categorization, and spam mail filtering. Textual data are encoded into numerical vectors in all above literatures. The KNN algorithm has been used as very popular approach to text mining tasks as well as any other kinds of classification tasks. Even if another approach has been proposed to the word and text categorization, it has been compared with the KNN algorithm. In this research, we adopt the KNN based on the above literatures and modify it into more suitable version for text mining tasks.

2. Encoding Schemes

This is concerned with the previous works on encoding words or texts into structured data. It has been assumed that the input data are always given as numerical vectors in applying machine learning algorithms to applications in any domain. In almost cases of applying machine learning algorithms to text mining texts, texts or words are encoded into numerical vectors. In order to solve the problems in encoding so, there were previous trials of encoding them into alternative structured data to numerical vectors. In this section, we explore the previous schemes of encoding texts or words into structured data.

It is known that texts or words are usually encoded into numerical vectors, by surveying articles on machine learning approaches to text categorization. In 1995, Winer encoded texts into approximately three hundreds dimensional numerical vectors, in applying the neural networks to the text categorization [31]. In 1999, Yang encoded texts absolutely into numerical vectors, in evaluating machine learning algorithms in apply them to the text categorization [32]. In 2002, Sebastiani surveyed the machine learning approaches to the text categorization, under the assumption of encoding texts into numerical vectors [28]. The fact that texts should be encoded into numerical vectors for doing the text mining tasks is confirmed through the above literatures.

There existed the previous trials of encoding text into tables, instead of tables for doing the text categorization and clustering. In 2008, Jo modified the single pass algorithm into its table based version for the text clustering [8]. In 2011, Jo invented the table marching algorithm as the method of categorizing texts in his patent [14]. In 2015, Jo improved the table matching algorithm into the stable and robust version [15]. The table becomes the alterative text representation to the numerical vector for doing the both tasks.

There also existed the previous trials of encoding texts into string vectors as one more alternative structured data for doing the both tasks. In 2009, Jo proposed the semantic similarity between string vectors for using the KNN and the SVM for ding the text categorization [10]. In 2010, Jo modified the k means algorithm into its string vector based version as the approach to the text clustering [11]. In 2015, Jo defined and characterized mathematically the numerical operations on string as the fundamental research for modifying other machine learning algorithms into their string vector based versions [16]. It requires to define and characterize more semantic operations for modifying more advanced machine learning algorithms.

In this research, we adopt the scheme where words are encoded into numerical vectors. Text identifiers are defined as features, and TF-IDF (Term Frequency-Inverse Document Frequency) and frequency are given as feature values. In order to avoid the poor discriminations among sparse numerical vectors, we need consider the similarity between texts which is the feature similarity. The feature similarity and the feature value similarity are combined with each other for computing the similarity between vectors. We modify the KNN into the version where both similarities are computed and combined with each other as the approach to the word categorization.

3. Specialized Machine Learning Algorithms

This section is concerned with the previous works on approaches to the text mining tasks where texts are encoded into alternative representations to numerical vectors. In Section 2, we explored the previous schemes of encoding texts into structured data and the modified versions of existing machine learning algorithms using the scheme. In this section, we introduce the string kernel which is a kernel function on string in using the SVM to the text categorization, and mention the new neural networks, NTC (Neural Text Categorizer) and NTSO (Neural Text Self Organizer) for doing the text categorization and the text clustering. It takes very much time for applying the string kernel version to the text categorization and needs more operations for advancing the created neural networks. In this section, we explore cases of using the string kernel based SVM and the two creative neural networks to text mining tasks.

The string kernel was proposed as a kernel function in using the SVM for classification tasks, in order to avoid problems in encoding texts into numerical vectors. In 2002, Lohi et al initially proposed the string kernel, in order solve the problems in doing so [24]. In 2004, Leslie et al applied the string kernel to protein classification where a protein is given as a string [23]. In 2006, Kate and Mooney used the string kernel for processing semantically sentences, instead of entire full texts [20]. The SVM with the string kernel works not successfully in the text categorization, but it works successfully in the protein classification.

The neural network, NTC (Neural Text Categorizer), was previously invented as a more suitable approach to the text categorization. In 2000, Jo initially proposed the NTC, but it used its weights which are fixed based on word frequencies [6]. In 2008 and 2010, Jo improved the NTC into the version where word weights are updated, and applied it to both the exclusive text categorizations and the soft ones [9][12]. In 2012, Pawar and Gawande mentioned the NTC as innovative approach to the text categorization, and in 2015, Abainia et al applied it for categorizing Arabic texts [1][26]. In future, the NTC needs to be expanded from the swallow version into deep version.

The NTSO (Neural Text Self Organizer) was previously created as a more suitable approach to the text clustering. In 2005, Jo and Japkowicz invented initially as the approach to the text clustering [18]. In 2006, Zehng et al mentioned the NTSO as one of the main text clustering tools [33]. In 2010, Jo validated empirically the NTSO performance by comparing it with the popular approaches such as the k means algorithm and the Kohonen Networks [13]. The NTSO will be modified into its supervised version and semi-supervised version in future, like the Kohonen Networks.

Recently, in 2015, Jo initially proposed the KNN which considers the feature similarity and the feature value similarity, as the approaches to the word categorization. However, it was not validated, empirically, and the work was published as a position paper, without the experiments, In this research, the proposed version of KNN will be validated empirically by comparing the traditional version for observing the effect of modifying the KNN, so.

III. PROPOSED WORKS

This article 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 it consists of the three sections. In section 1, we describe the process of encoding words into numerical vectors. In section 2, we do formally the proposed scheme of computing the similarity between two numerical vectors. In section 3, we mention the proposed version of KNN algorithm which considers the similarity among features as the approach to word categorization. Therefore, this article is intended to describe in detail the modified version of KNN algorithm and its application to the word categorization.

1. Word Encoding

This section is concerned with the process of encoding words into numerical vectors. Previously, texts each of which is consists of paragraphs were encoded into numerical vectors whose attributes are words. In this research, we attempt to encode words into numerical vectors whose attributes are text identifiers which include them. Encoding of words and texts into numerical vectors looks reverse to each other. In this section, we describe in detail the process of mapping words into numerical vectors, instead of texts.

In the first step of word encoding, a word-document matrix is constructed automatically from a text collection called corpus. In the corpus, each text is indexed into a list of words. For each word, we compute and assign its weight which is called TF-IDF (Term Frequency-Inverse Document Frequency) weight [1], by equation (1),

$$w_i = (1 + \log TF_i)(\log N - \log DF_i) \tag{1}$$

where TF_i is the total frequency in the given text, DF_i is the total number of documents including the word, and is the total number of documents in the corpus. The word-document matrix consists of TF-IDF weights as relations between a word and a document computed by equation (1). Note that the matrix is a very huge one which consists at least of several thousands of words and documents.

Let us consider the criterion of selecting text identifiers as features, given labeled sampled words and a text collection. We may set a portion of each text in the given sample words as a criteria for selecting features. We may use the total frequency of the sample words in each text as a selection criterion. However, in this research, we decided the total TF-IDF (Term Frequency and Inverse Document Frequency) which is computed by equation (1) as the criterion. We may combine more than two criteria with each other for selecting features.

Once some texts are selected as attributes, we need to consider the schemes of defining a value to each attribute. To each attribute, we may assign a binary value indicating whether the word present in the text which is given as the attribute, or not. We may use the relative frequency of the word in each text which is an attribute as a feature value. The weight of word to each attribute which is computed by equation (1) may be used as a feature value. Therefore, the attributes values of a numerical vector which represent a word are relationships between the word and the texts which are selected as features.

The feature selection and the feature value assignment for encoding words into numerical vectors depend strongly on the given corpus. When changing the corpus, different texts are selected by different values of the selection criterion as features. Even if same features are selected, different feature values are assigned. Only addition or deletion of texts in the given corpus may influence on the feature selection and the assignment of feature values. In order to avoid the dependency, we may consider the word net or the dictionary as alternatives to the corpus.

2. Feature Similarity

This section is concerned with the scheme of computing the similarity between numerical vectors as illustrated in Fig. 1. In this research, we call the traditional similarity measures such as cosine similarity and Euclidean distance feature value similarities where consider only feature values for computing it. In this research, we consider the feature similarity as well as the feature value similarity for computing it as the similarity measure which is specialized for text mining tasks. The numerical vectors which represent texts or words tend to be strongly sparse; only feature value similarity becomes easily fragile to the tendency. Therefore, in this section, as the solution to the problem, we describe the proposed scheme of computing the similarity between numerical vectors.

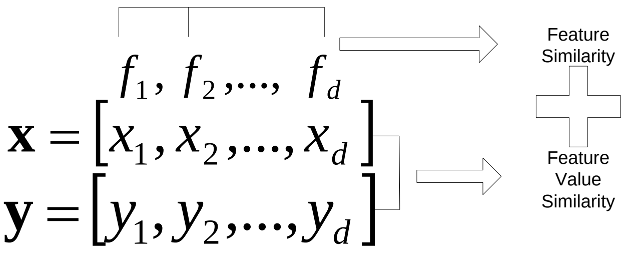
$f_1, f_2, \ldots, f_d$
$\mathbf{x} = [x_1, x_2, \ldots, x_d]$
$\mathbf{y} = [y_1, y_2, \ldots, y_d]$
Feature Similarity
Feature Value Similarity

Fig. 1. The Combination of Feature and Feature Value Similarity

Text identifiers are given as features for encoding words into numerical vectors. Texts are dependent on others rather than independent ones which are assumed in the traditional classifiers, especially in Naïve Bayes [24]. Previously, various schemes of computing the semantic similarity between texts were developed [2]. We need to assign nonzero similarity between two numerical vectors where non-zero elements are given to different features with their high similarity. It is expected to improve the discriminations among sparse vectors by considering the similarity among features.

We may build the similarity matrix among features automatically from a corpus. From the corpus, we extract easily a list of text identifiers. We compute the similarity between two texts by equation (2),

$$s_{ij} = sim(d_i, d_j) = \frac{2 \times tf(d_i, d_j)}{tf(d_i) + tf(d_i)}$$

where $tf(d_i, d_j)$ is the number of words which are shared by both texts, d_i and d_j, and $tf(d_i)$ is the number of words which are included in the text, d_i We build the similarity matrix which is consists of similarities between text identifiers given as features as follows:

$$S = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1k} \\ s_{21} & s_{22} & \cdots & s_{2k} \\ \cdots & \cdots & \cdots & \cdots \\ s_{k1} & s_{k2} & \cdots & s_{kk} \end{bmatrix}$$

The rows and columns in the above matrix, S , correspond to the d text identifiers which are selected as the features.

The texts, $d_1, d_2, .., d_k$ are given as the features, and the two words, t_1 and t_2 are encoded into the two numerical vectors as follows:

and

The features, are defined through the process which was described in section 1. We construct the by matrix as the similarity matrix of features by the process mentioned above. The similarity between the two vectors are computed with the assumption of availability of the feature similarities, by equation (2),

$$t_1 = [w_{11}, w_{12}, .., w_{1k}] \text{ and } t_2 = [w_{21}, w_{22}, .., w_{2k}]$$

The features, $d_1, d_2, .., d_k$ are defined through the process which was described in section 1. We construct the k by k matrix as the similarity matrix of features by the process mentioned above. The similarity between the two vectors are computed with the assumption of availability of the feature similarities, by equation (2),

$$sim(t_1, t_2) = \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} s_{ij} w_{1i} w_{2j}}{k\|t_1\| \cdot \|t_2\|}, \qquad (2)$$

where $\|t_1\| = \sqrt{\sum_{i=1}^{k} w_{1i}^2}$ and $\|t_2\| = \sqrt{\sum_{i=1}^{k} w_{2i}^2}$ We get the value of s_{ij} in equation (2) by looking up it from the similarity matrix.

The proposed scheme of computing the similarity by equation (2) has very high complexity as payment for obtaining the more discrimination among sparse vectors. Let us assume that two d dimensional numerical vectors are given as the input for computing the similarity between them. It takes only linear complexity, $O(d)$, to compute the cosine similarity as the traditional one. However, in the proposed scheme takes quadratic complexity, $O(d^2)$. We may reduce the complexity by computing similarities of some pairs of features, instead of all.

3. Proposed Version of KNN

This section is concerned with the version of K Nearest Neighbor which considers both the feature similarity and the feature value one. The sample words are encoded into numerical vectors whose features are texts by the scheme which was described in Section 1. The novice word is given as the classification target, and it is also encoded into a numerical vector. Its similarities with the sample words are computed by equation (3) for selecting nearest neighbors, in the proposed version. Therefore, in order to provide the detail algorithm, we describe the proposed KNN version, together with the traditional one.

The traditional KNN version is illustrated in Fig. 2. The sample words which are labeled with the positive class or the negative class are encoded into numerical vectors. The similarities of the numerical vector which represents a novice word with those representing sample words are computed using the Euclidean distance or the cosine similarity. The k most similar sample words are selected as the k nearest neighbors and the label of the novice entity is decided by voting their labels. However, note that the traditional KNN version is very fragile in computing the similarity between very sparse numerical vectors.

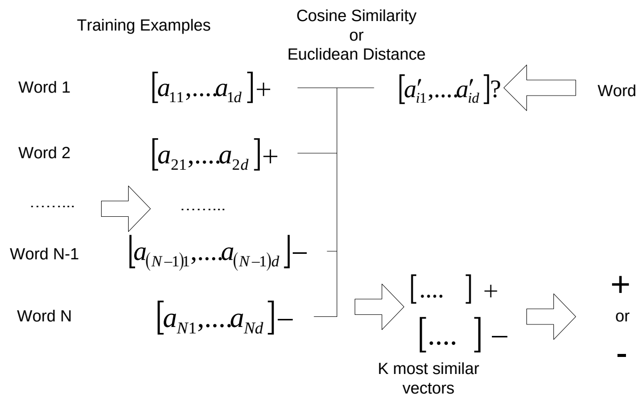

Fig. 2. The Traditional Version of KNN

The proposed KNN version is illustrated in Fig. 3. Like the traditional version, a word is given as an input and it is encoded into a numerical vector. The similarities of the novice word with the sample ones are computed by equation (3) which was presented in Section 2. Like the traditional version, k most similar samples are selected as the nearest neighbors, and the label of the novice is decided by voting their labels. The scheme of computing the similarity between numerical vectors is the essential difference between the two versions.

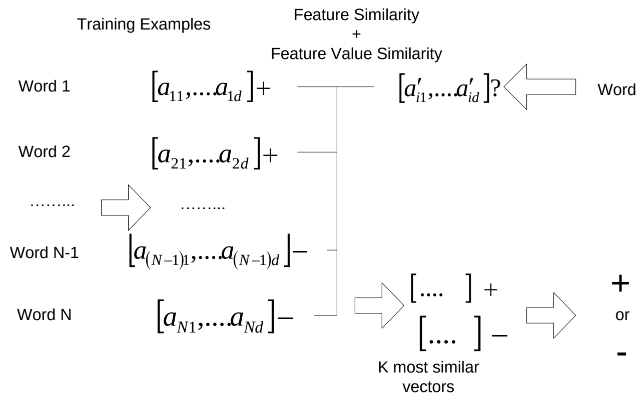

Fig. 3. The Proposed Version of KNN

We may derive some variants from the proposed KNN version. We may assign different weights to selected neighbors instead of identical ones: the highest weights to the first nearest neighbor and the lowest weight to the last one. Instead of a fixed number of nearest neighbors, we select any number of training examples within a hyper-sphere whose center is the given novice example as neighbors. The categorical scores are computed proportionally to similarities with training examples, instead of selecting nearest neighbors. We may also consider the variants where more than two variants are combined with each other.

Let us compare the both KNN versions with each other. In computing the similarity between two numerical vectors, the traditional version uses the Euclidean distance or cosine similarity mainly, whereas the proposed one uses the equation (3). Both versions are common in selecting k nearest neighbors and classifying a novice item by voting the labels of them. However, the proposed version is more tolerant to sparse numerical vectors in computing the similarities among them than the traditional version.

IV. EXPERIMENTS

This section is concerned with the empirical experiments for validating the proposed version of KNN, and consists of the four sections. In Section 1, we present the results from applying the proposed version of KNN to the word categorization on the collection, ‘NewsPage.com’. In Section 2, we show the results from applying it for categorizing words from the collection, ‘Opinosis’. In Section 3, we mention the results from comparing the two versions of KNN with each other in categorizing words from 20NewsGroups. In Section 4, we make the general discussions which is concerned with results from validating the proposed version of KNN, finally.

1. 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 1, 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 [15]. 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 1. We select words by their frequencies concentrated in the given category combined with subjectivities in building the word collection.

Table 1. 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	2,000	1,200	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.

Fig. 3 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 vectors which represent words. 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.

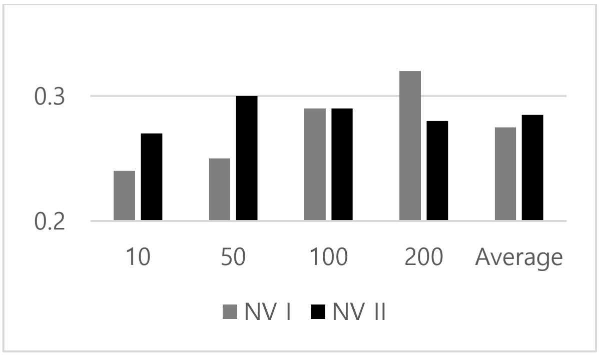

Fig. 3. Results from Classifying Words in Text Collection: NewsPage.com

Let us make discussions on the results from doing the word categorization, using the both versions of KNN algorithm, as shown in Fig. 3. The accuracy which are the performance measure of this classification task is in range between 0.24 and 0.32. The proposed version of KNN algorithm works better in the all input sizes. As the input size increases, the performance difference between the proposed and the traditional versions decreases. In this set of experiments, we conclude that the proposed version works slightly better than the traditional one, in averaging over the four cases.

2. Opiniopsis

This section is concerned with the set of experiments for validating the better performance of the proposed version on the collection: Opinopsis. 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, Opiniopsis.

In Table 2, we illustrate the text collection, Opiniopsis, 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 2. The Number of Texts and Words in Opiniopsis

Category	#Texts	#Training Words	#Test Words
Car	23	300	75
Electronics	16	300	75
Hotel	12	300	75
Total	51	900	125

We perform this set of experiments by the process which is described in Section 1. We extract the 300 important words by scanning individual texts in each category, and encode them into numerical vectors, with 10, 50, 100, and 200 dimensions. 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 Fig. 4, we illustrate the experimental results from categorizing the words using the both versions of KNN on this collection. Like Fig. 1, 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 Fig. 4, the most right group indicates the average over results of the left three groups. Therefore, Fig. 2 presents the results from classifying the words into one of the three categories by both versions of KNN algorithm, on the collection, Opinopsis.

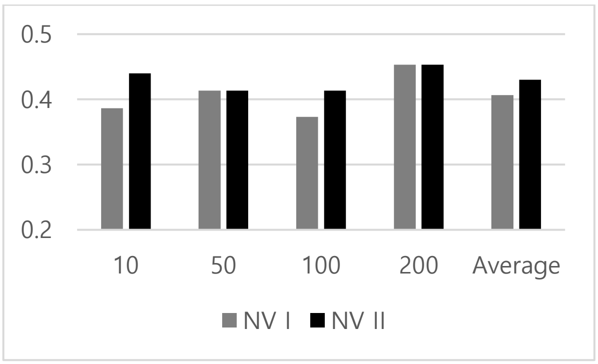

Fig. 4. Results from Classifying Words in Text Collection: Opiniopsis

We discuss the results from doing the word categorization using the both versions of KNN algorithm, on Opinosis, shown in Fig. 4. The accuracies of the both versions range between 0.4 and 0.45 in this task. The proposed version works better than the traditional one in the two input sizes: 10 and 100. It is comparable with the traditional version in the others: 50 and 200. From this set of experiments, we conclude that the proposed one works slightly better in averaging over the four cases.

3. 20NewsGroup

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 3, 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 3. 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 3. In the process of gathering the classified words, they are selected by their frequencies which are concentrated in their corresponding categories.

Table 3. The Number of Texts and Words in 20NewsGroup

Category	#Texts	#Training Words	#Test Words
Comp	1,000	300	75
Rec	1,000	300	75
Sci	1,000	300	75
Talk	1,000	300	75
Total	4,000	1,200	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 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.

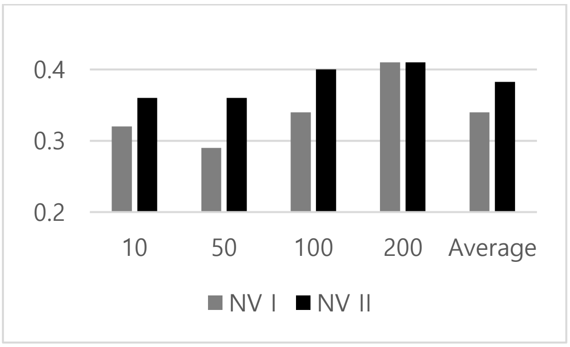

Fig. 5. Results from Classifying Words in Text Collection: 20NewsGroups

In Fig. 5, we illustrate the experimental results from categorizing words using the both versions on the broad version of 20NewsGroups. Fig. 5 has the identical frame of presenting the results to those of Fig. 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.

Let us discuss the results from doing the word categorization using the both versions on 20NewsGroups as shown in Fig. 5. The accuracies of the both versions range between 0.28 and 0.42. The proposed version of KNN algorithm shows its better performances in the three of the four cases. In the input size, 200, it is competitive with the traditional version. From this set of experiments, we conclude that the proposed version wins over the traditional one, in averaging over their four achievements.

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 three collections. The proposed versions show its better results in all of the three collections. On the three collections, the accuracies of the traditional version range between 0.24 and 0.45, while, those of the proposed version range between 0.26 and 0.45. 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. 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 word categorization system as a real program.

Acknowledgement

This work was supported by 2017 Hongik University Research Fund.

References

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

[2]R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval: The Concepts and Technology behind Search, Addison-Wesley, 2011.

[3] L. Firte, C. Lemnaru, and R. Potolea, “Spam detection filter using KNN algorithm and resampling”, pp27-33, in Proceedings of IEEE International Conference on Intelligent Computer Communication and Processing, pp. 27-33, 2010.

[4] E. Han, S. G. Karypis, and V. Kumar, “Text categorization using weight adjusted k-nearest neighbor classification”, in Proceedings of Pacific-asia conference on knowledge discovery and data mining, pp. 53-65, 2001.

[5] C. James, I. Koprinska, and J. Poon, “A neural network based approach to automated e-mail classification”, pp702-705, in Proceedings of IEEE International Conferences on Web Intelligence, pp.702-705, 2003.

[6] T. Jo, “NeuroTextCategorizer: A New Model of Neural Network for Text Categorization”, in Proceedings of ICONIP, pp. 280-285, 2000.

[7] T. Jo, “The Implementation of Dynamic Document Organization using Text Categorization and Text Clustering”, PhD Dissertation, Department of Computer Science, University of Ottawa, Ottawa, Canada, 2006.

[8] T. Jo, “Table based Single Pass Algorithm for Clustering News Articles”, International Journal of Fuzzy Logic and Intelligent Systems, vol. 8, no. 3, pp. 231-237, 2008.

[9] T. Jo, “Neural Text Categorizer for Exclusive Text Categorization”, Journal of Information Processing Systems, vol. 4, no 2, pp. 77-86, 2008.

[10] T. Jo, “Modification of Classification Algorithm in Favor of Text Categorization”, International Journal of Computer Science and Software Technology, vol. 2, no. 1, pp. 13-23, 2009.

[11] T. Jo, “Modification of Clustering Algorithms for Text Clustering”, International Journal of Computer Science and Software Technology, vol. 3, no. 1, pp.21-

33, 2010.

[12] T. Jo, "NTC (Neural Text Categorizer): Neural Network for Text Categorization", International Journal of Information Studies, vol. 2, no. 2, pp. 83-96, 2010.

[13] T. Jo, "NTSO (Neural Text Self Organizer): A New Neural Network for Text Clustering", pp31-43, Journal of Network Technology, pp. 31-43, vol. 1, no. 1, 2010.

[14] T. Jo, "Device and Method for Categorizing Electronic Document Automatically", 10-2009-0041272, 10-1071495, 2011.

[15] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", Soft Computing, vol. 19, no. 4, pp. 839-849, 2015.

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

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

[18] T. Jo and N. Japkowicz, "Text Clustering using NTSO", The Proceedings of IJCNN, pp. 558-563, 2005.

[19] A. Khan, B. Baharudin, L.H. Lee, and K. Khan, "A review of machine learning algorithms for text-documents classification", Journal of advances in information technology, vol 1, no 1, pp. 4-20, 2010.

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

[21] Y. Kim, B. Zhang, and Y.T. Kim, "Collocation dictionary optimization using WordNet and k-nearest neighbor learning", Machine Translation, vol. 16, no. 2, pp. 99-108, 2001.

[22] C. Lai and M. Tsai, "An empirical performance comparison of machine learning methods for spam e-mail categorization", in Proceedings of IEEE International Conference on Hybrid Intelligent Systems, pp. 44-48, 2004.

[23] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, "Mismatch String Kernels for Discriminative Protein Classification", Bioinformatics, vol. 20, no. 4, pp. 467-476, 2004.

[24] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text Classification with String Kernels", Journal of Machine Learning Research, vol. 2, no. 2, pp. 419-444, 2002.

[25] T. Mitchell, Machine Learning, McGraw-Hill, 1997.

[26] P. Y. Pawar and S. H. Gawande, "A Comparative Study on Different Types of Approaches to Text Categorization", International Journal of Machine Learning and Computing, vol. 2, no. 4, pp. 423-426, 2012.

[27] V. Pekar and S. Staab, "Word classification based on combined measures of distributional and semantic similarity", in Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics, pp. 147-150, 2003.

[28] F. Sebastiani, "Machine learning in automated text categorization", ACM Computing Survey, vol. 34, no. 1, pp. 1-47, 2002.

[29] M. Stauffer and A. Fischer and K. Riesen, "A novel graph database for handwritten word images", Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition, pp. 553-563, 2016.

[30] B. Vishwanath, V. Kumar, P. Kumari, and J. Pascual, "KNN based machine learning approach for text and document mining", International Journal of Database Theory and Application, vol. 7, no. 1, pp. 61-70, 2014.

[31] E. D. Wiener, "A Neural Network Approach to Topic Spotting in Text", Master Thesis, the Faculty of the Graduate School of the University of Colorado, 1995.

[32] Y. Yang, "An evaluation of statistical approaches to text categorization", Information retrieval, vol. 1, no. 1, pp. 69-90, 1999.

[33] Y. Zheng, X. Cheng, R. Huang, and Y. Man, "A comparative study on text clustering methods", Advanced Data Mining and Applications, pp. 644-651, 2006.

Author

Taeho Jo is currently working for Hongik University as a faculty member. He received his Bachelor degree from Korea University in 1994, his Master degree from Pohang University of Science and Technology in 1997, and his PhD degree from University of Ottawa in 2006. His research area spans mainly over text mining, neural networks, machine learning, and information retrieval. He has the four years experience of working for industrial organizations and ten years experience of working for academic ones. In 2016, he was awarded in the biography dictionary, "Marquis Who's Who in the World".