Modification of K Nearest Neighbor by Graph Similarity Metric for Keyword Extraction

Taeho Jo
President
Alpha AI Publication
Cheongju, South Korea
tjo018@hongik.ac.kr

***Abstract*—This article proposes the modified KNN (K Nearest Neighbor) algorithm which receives a graph as its input data and is applied to the keyword extraction. The graph is more graphical for representing a word and the keyword extraction is able to be mapped into the binary classification where each word is classified into keyword or non-keyword. In the proposed system, a text which is given as the input is indexed into a list of words, each word is classified by the proposed KNN version, and the words which are classified into keyword are extracted ad the output. The proposed KNN version is empirically validated as the better approach in deciding whether each word is a keyword or non-keyword in news articles and opinions. In this article, a word is encoded into a weighted and undirected graph and it is represented into a list of edges.**

I. Introduction

Keyword extraction refers to the process of extracting the important words from an article as its keywords. The keywords are very important indications for performing the tasks involved in information retrieval, so the researchers and developers of information retrieval systems are interested in developing the schemes of extraction keywords, automatically. In this research, the task is viewed into a binary word classification where each word is classified into a keyword or a non-keyword. We prepare the sample words which are labeled with one of 'keyword' or 'non-keyword', and construct the classification capacity by learning the sample words. In this research, we assume that the supervised learning algorithms are used as the approach to the classification which is derived from the keyword extraction.

Let us consider some facts which become the motivations for doing this research. Requirement of many features for the robustness in encoding words or texts into numerical vectors causes too much computation time [3]. The sparse distribution in each numerical vector as the additional effect of using too many features for encoding words into numerical vectors causes very poor discriminations among vectors [3]. Recently, previous works proposed that knowledge should be transformed into ontologies which are given graphs[2][30]. Therefore, in this research, motivated by the above facts, we attempt to encode words into graphs and modify the machine learning algorithm into its graph based version.

Let us consider some points which this research proposes as its ideas. In this research, each word is encoded into a graph with its vertices which indicate text identifiers and with its edges which indicate their semantic relations. In this research, the keyword extraction is viewed into an instance of classification task, and a similarity measure between two graphs is defined. We modify the KNN (K Nearest Neighbors) into its graph based version where a graph is given as the input data by itself, and use it as the approach to the keyword extraction. Even if the keyword extraction is interpreted into the word classification, it should be distinguished from the task of classifying words into one of the predefined topics.

Let us mention what we expect from this research as the benefits. We may expect the more semantic and graphical representations as indicated inherently by graphs. We may also expect the improved discrimination among graphs by avoiding completely the sparse distribution among numerical vectors which represented words in previous works. We expect the better performance by encoding words into alternative representations to numerical vectors; the problems which are caused by encoding words into numerical vectors are solved completely. Hence, the goal of this research is to develop the keyword extraction system with the benefits as a module or an independent program.

This article is organized into the five sections. In Section II, we survey the relevant previous works. In Section III, we describe in detail what we propose in this research. In Section IV, we validate empirically the proposed approach by comparing it with the traditional one. In Section V, we mention the general discussion on the empirical validations and remaining tasks for doing the further research.

II. Previous Works

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

vectors to numerical vectors. Therefore, in this section, we provide the history about this research, by surveying the relevant previous works.

A. Applications to Word Classification Tasks

This section is concerned with the previous cases of applying the modern KNN algorithm for the keyword extraction and its similar tasks. We mention the topic based word categorization and the index optimization, together with the keyword extraction, as the kinds of word classification task. The KNN algorithm is modified into the version which solves the problem, in encoding words into numerical vectors, completely. We survey the successful results in applying the modern version of KNN algorithm to the tasks. This section is intended to explore the previous works with the successful results in applying the modernized version to the tasks.

Let us mention some works on the modernized KNN algorithms which are tools of the topic based word categorization. The KNN algorithm which was modernized by considering the similarities among features was applied to the word categorization [9]. The modernized KNN algorithm which classifies a table directly was proposed as the approach to the word classification [10]. The KNN algorithm was modified into the version which classifies a string vector, instead of a numerical vector, in using it for the word categorization [11]. In the mentioned literatures, the KNN algorithms which were modernized with their different directions were applied to the word categorization.

Let us explore the previous works on applying the modernized KNN algorithm to the keyword extraction which is covered in this study. The KNN algorithm which uses the similarity metric considering the feature similarities was applied to the keyword extraction [12]. The KNN algorithm which classifies a table directly was proposed as an approach to the keyword extraction [13]. The KNN algorithm which receives a string vector as its input data was considered for using it to the keyword extraction [14]. The keyword extraction was mapped into the binary classification task for applying the supervised learning algorithm in the above literatures.

The text summarization is considered as the one which is similar as the keyword extraction, in selecting essential parts. The KNN algorithm which is modernized by consider the feature similarities in computing a similarity between numerical vectors was applied to the text summarization [15]. The one which is modernized into the version which classifies a table directly was used for the text summarization [20]. One more modernized version which classify a string vector directly was proposed as the approach to the text summarization [21]. In the above literatures, the text summarization is viewed as the classification of each paragraph into summary or non-summary.

Let us mention some points which distinguish this research from the previous works. We explored the previous cases of using the three types of modernized KNN algorithms for the keyword extraction and its related tasks. We mentioned the word categorization which is the source from which the keyword extraction is derived as a specific instance and the text summarization where a paragraph is classified based on its importance degree in a given text. The modernized version of KNN algorithm which is proposed as the approach to the keyword extraction, classifies a graph directly. The keyword extraction is mapped into a binary classification of words, following the style in the previous works, and the proposed version is applied to the task, in this study.

B. Word and Text Encoding

This section is concerned with the schemes of encoding texts or words. The problems in encoding texts or words into numerical vectors, such as the huge dimensionality and the sparse distribution in each numerical vector, were pointed out. The previous works challenged against the problems by encoding words or texts into other structured forms. The tables, the string vectors, and the graphs are mentioned as scope of other structured form in this section. This section is intended to explore the previous cases of encoding texts or words into one of other structured forms.

Let us survey the previous works on mapping texts or words into tables. Words were mapped into tables in using the AHC algorithm for the word clustering [16]. Texts were mapped into tables in using the KNN algorithm for the text categorization [17]. In using the AHC algorithm for the text clustering, texts mapped so [22]. In the above literatures, in using the AHC algorithm and the KNN algorithm texts or words were encoded into tables.

Let us mention the previous cases of encoding words or texts into string vectors. It was proposed that words should be encoded into string vectors in using the AHC algorithm for clustering words [18]. In using the KNN algorithm for the text categorization, it was proposed that texts should be encoded into string vectors [19]. In using the AHC algorithm for clustering texts, texts were encoded into string vectors [23]. The above literatures presented the previous cases of encoding raw data into string vectors.

Let us explore the previous works on encoding texts or words into graphs. Words were encoded into graphs in applying the AHC algorithm to the semantic word clustering [8]. Texts were encoded into graphs in applying the KNN algorithm to the text categorization [24]. In applying the AHC algorithm to the text clustering, texts were encoded so [25]. In the above literatures, we present the cases of mapping raw data into graphs.

We mentioned the three types of structured data for representing words or texts through the previous works. We adopt the third type of structured data, called graphs, as word

representations. We define the similarity metric between graphs, and modifies the KNN algorithm into version based on it. We use the modified version of KNN algorithm for implementing the keyword extraction system. We empirically validate the modified version in keyword extractions on different test sets, comparing it with the traditional version.

C. Non-Numerical Vector based Machine Learning Algorithms

This section is concerned with the previous works on the non-numerical vector based machine learning algorithms. In the previous section, we presented the cases of encoding words or texts into non-numerical vectors, in using the KNN algorithm and the AHC algorithm. As the approach to the text categorization which process non-numerical vectors, we mention the string kernel based Support Vector Machine, and the table matching algorithm, and the Neural Text Categorizer, in this section. Because the keyword extraction is mapped into the binary classification, the task belongs to the classification task, together with the text classification. This section is intended to survey the previous works on the three machine learning algorithms which process non-numerical vectors.

Let us mention the string kernel as the similarity metric between two raw texts. The string kernel was initially proposed by Lodhi et al. in 2006 for modifying the SVM as the approach to the text classification [29]. It was utilized for modifying the k means algorithm as the approach to the text clustering by Karatzoglou and Feinerer in 2006 [28]. The string kernel based SVM was applied to the sentence classification tasks by Kate et al. in 2006 [27]. The string kernel which was proposed or used in the above literatures is the lexical similarity metric between raw texts based on characters.

Let us explore the previous works on another non-numerical vector based classification algorithm which is called table based marching algorithm. It was initially proposed as the approach to the text classification by Jo and Cho in 2008 [26]. It was used for the soft text categorization where it is possible to classify each text into more than one category [4]. It was upgraded as a more robust and stable version by Jo in 2015 [7]. In using the approach to the text categorization which is mentioned in the above literatures, texts should be encoded into tables.

Let us mention the Neural Text Categorizer as the neural networks which are specialized for the text categorization. It was initially proposed as the approach to the text categorization by Jo in 2008 [5]. It was validated in both the soft text categorization and the hard text categorization by Jo in 2010 [6]. It was used for classifying Arabian texts by Abainia et al. in 2015 [1]. It was mentioned as an innovative neural networks by Vega and Medez-Vasquez [31].

We mentioned the three classification algorithms which deal with non-numerical vectors as approaches to the text categorization. In the above literatures, the non-numerical vector based algorithms were applied to the text categorization. In this research, words are encoded into graphs as one more type of non-numerical vectors. The KNN algorithm will be modified into the version which processes directly graphs. Its performance will be validated compared with the traditional KNN algorithm in the classifications tasks which are mapped from the keyword extractions.

III. Proposed Approach

This section is concerned with encoding words into graphs, modifying the KNN (K Nearest Neighbor) into the graph based version and applying it to the keyword extraction, and consists of the four sections. In Section III-A, we deal with the process of encoding words into graphs. In Section III-B, we describe formally the process of computing the similarity between to graphs. In Section III-C, we do the graph vector based KNN version as the approach to the keyword extraction. In Section III-D, we explain the system architecture of the keyword extraction system where the proposed KNN is adopted.

A. Word Encoding

This section is concerned with mapping words into graphs. We surveyed the previous works on mapping raw data into graphs in Section II-B. A word is encoded into a graph with the three steps: vertex set definition, edge set definition, and edge weighting. In the graph which represents a word, a vertex indicates a text identifier which relates it, and an edge indicates a similarity between texts. This section is intended to describe the three steps which are presented in Figure 1-3.

The process of defining a vertex set in encoding a word into a graph is illustrated in Figure 1. The corpus as the source and a word as the encoding target are initially given. Texts which include the word are extracted as vertices. Only some texts are selected among them, if too many texts are extracted and more texts may be added using associated words to the existing set of vertices, if very few texts are extracted. The manipulation on the number of vertices becomes the cause of overestimation or underestimation in computing the similarity between graphs.

The edges in encoding a word into a graph are illustrated in Figure 2. Texts which are relevant to the word are extracted from the corpus as the vertices of the graph in the previous step. The complete links are defined as the edge candidates to the vertices and some are selected among them by their weights. The weight which is assigned to each edge indicates the similarity between two texts. If very small number of vertices is extracted, we may use all edge candidates.

The $N \times N$ matrix and the equation for weighting edges are illustrated in Figure 3. It is assumed that the N texts are defined as the vertices in the first step. In the matrix,

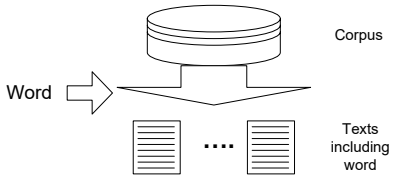

Figure 1. Word Indexing

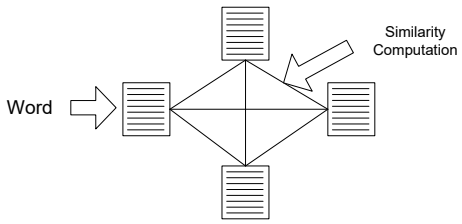

Figure 2. Word Representation: Graph

the diagonal elements are absolutely given as 1.0 and off-diagonal ones are computed by the equation in Figure 3, as similarities among texts between zero and one. The off-diagonal elements of the matrix by weights of complete links among vertices. Some with higher weights are selected as the final weighted edges, among off-diagonal elements.

We mentioned the graph which represents a word with the three stops which are presented in Figure 1-3. In encoding a word into a graph, the vertices are given as texts which are relevant to the word, and the edges are given as the similarities among texts. A graph is viewed as the set of edges each of which consists of two nodes and the weight, in the implementation level. The graph which represents a word belong to the weighted undirected graph where each edge between two nodes is bidirectional and a value is assigned between them. The weight which is assigned to each edge is given as a normalized value between zero and one.

B. Similarity Metric

This section is concerned with the similarity metric between two graphs. In the previous section, we studied the process of converting words into graphs. We need to define the similarity metric between two graphs for modifying the KNN algorithm as the approach to the keyword extraction. We view a graph as a set of edges and starts with defining the similarity between two edges. This section is intended to describe the computation of similarity between two graphs.

Let us mention the computation of similarity between two edges as the basis for computing one between two graphs. Each edge is expressed as an entry of three values as shown in equation (1),

$$e \equiv (node_1, node_2, weight) \tag{1}$$

the two edges, e_1 and e_2 are expressed as equation (2) and (3),

$$e_1 = (node_{11}, node_{12}, weight_1) \tag{2}$$

$$e_2 = (node_{21}, node_{22}, weight_2) \tag{3}$$

and $weight_1$ and $weight_2$ are given as normalized values between zero and one.

We consider the three possible cases between two edges as both nodes are same to each other, either of them is same,

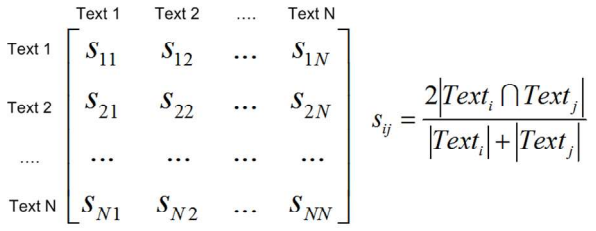

Figure 3. Similarity Matrix

and neither of them is so. The similarity between two edges is defined on the three conditions:

- In the two edges, if both nodes are same to each other, the similarity between them is defined by equation(4),

$$sim(e_1, e_2) = \frac{1}{2}(weight_1 + weight_2) \tag{4}$$

- In the two edges, if only either of two nodes are same to each other, the similarity between them is defined by equation(5),

$$sim(e_1, e_2) = (weight_1 \times weight_2) \tag{5}$$

- In the two edges, if no node are same to each other, the similarity between them is zero.

The edge similarity will be used for computing the similarity between an edge and a graph, next.

The similarity between two edges is expanded into one between an edge and a graph. The graph, G is expressed as a set of edges, $G = \{e_1, e_2, \ldots, e_{|G|}\}$. The similarity, $sim(e_i, G)$, is computed by equation (6),

$$sim(e_i, G) = \max_{k=1}^{G} sim(e_i, e_k) \tag{6}$$

The similarity between an edge and a graph is the maximum among its similarities with ones in the graph, as shown in equation (6). When only edge e_r in the graph, G, have both identical, assuming that all weights are constant between zero and one, the similarity between an edge and a graph is expressed by equation (7),

$$sim(e_i, G) = sim(e_i, e_r) \tag{7}$$

The similarity between an edge and a graph is expanded into one between two graphs,further. The two graphs are notated by G_1 and G_2, and they are viewed as edge sets, as shown in equation (8) and (9),

$$G_1 = \{e_{11}, e_{12}, \ldots, e_{1|G_1|}\} \tag{8}$$

$$G_2 = \{e_{21}, e_{22}, \ldots, e_{2|G|_2}\} \tag{9}$$

For each edge in the graph, G_1, its similarity with the graph, G_2 is computed by equation (6). The similarity between the two graphs, G_1 and G_2 is computed by equation (10),

$$sim(G_1, G_2) = \frac{1}{|G_1|} \sum_{i=1}^{|G_1|} sim(e_{1i}, G_2) \tag{10}$$

The similarity between the two graphs, G_1 and G_2, is always given as a normalized value between zero and one.

We mentioned the similarity between two graphs as a normalized value between zero and one, and let us assume that all edges are weighted as 1.0 in both graphs. If $G_1 = G_2$, the similarity between two graphs is given as 1.0 by equation (11),

$$\begin{aligned} sim(e_{1i}, G_2) &= 1.0 \\ sim(G_1, G_2) = \frac{1}{|G_1|} \sum_{i=1}^{G_1} sim(e_{1i}, G_2) &= \frac{|G_1|}{|G_1|} = 1.0 \end{aligned} \tag{11}$$

If no vertex shared by two graphs, the similarity between two graphs given as zero by equation (12),

$$\begin{aligned} sim(e_{1i}, G_2) &= 0.0 \\ sim(G_1, G_2) = \frac{1}{|G_1|} \sum_{i=1}^{G_1} sim(e_{1i}, G_2) &= \frac{0}{|G_1|} = 0.0 \end{aligned} \tag{12}$$

The similarity between two graphs is always given as a normalized value between zero and one by equation (13),

$$\begin{aligned} G_1 \cap G_2 \subseteq G_1, G_1 \cap G_2 \subseteq G_2 \\ 0 \leq sim(e_{1i}, G_2) \leq 1.0 \\ 0 \leq \frac{1}{|G_1|} \sum_{i=1}^{G_1} sim(e_{1i}, G_2) \leq 1.0 \\ 0 \leq sim(G_1, G_2) \leq 1.0 \end{aligned} \tag{13}$$

Each edge is usually weighted between zero and one, so the similarity between two graphs is clearly given as a normalized value.

C. Proposed Version of KNN

The proposed version of KNN algorithm as the approach to the keyword extraction is illustrated in Figure 4. We mentioned the process of encoding words into graphs in Section III-A, and assume that the training examples and a novice item are given as graphs. We use the similarity metric between graphs which is described in Section III-B for selecting nearest neighbors from the training examples. In addition, variants may be derived by defining more selection schemes and more voting ones. This section is intend to describe the proposed version of KNN algorithm as the approach to the keyword extraction.

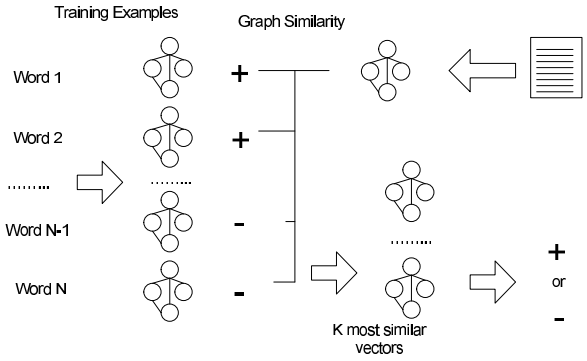

Figure 4. The Proposed Version of KNN

Let us mention the process of selecting nearest neighbors from the training examples as the main step of using the KNN algorithm for a classification task. The training words and a novice word are mapped into graphs by the process which is covered in Section III-A. The similarities of a novice word with the training ones are computed by equation (10). The training examples are ranked by their similarities and the k most similar ones are selected as the nearest neighbors. We adopt the rank based scheme in selecting nearest neighbors.

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, G , whose elements are given as tables and their target labels, by equation (14),

$$Ne_k(G) = \{(G_1, y_1), (G_2, y_2), \ldots, (G_k, y_k)\}, \quad y_i \in \{c_1, c_2, \ldots, c_m\} \tag{14}$$

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 $Count(Ne_k(G), c_i)$. The label of the novice item, G, is decided by the majority of categories in the nearest neighbors, as expressed by equation (15),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Count(Ne_k(G), c_i) \tag{15}$$

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, $sim(G, G_1), sim(G, G_2), \ldots, sim(G, G_k)$ as weights, $w_1, w_2, \ldots, w_k$ by equation (16),

$$w_i = sim(G, G_i) \tag{16}$$

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 (17),

$$Weight(Ne_k(G), c_i) = \sum_{G_j \in c_i}^{k} w_j \tag{17}$$

The label of the novice item, G, is decided by the category which corresponds to the maximum sum of weights as shown in equation (18),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Weight(Ne_k(G), c_i) \tag{18}$$

When the weights of nearest neighbors are set constantly, equation (18) is same to equation (15), as expressed in equation (19),

$$Weight(Ne_k(G), c_i) = Count(Ne_k(G), c_i) \tag{19}$$

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

D. Keyword Extraction System

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

The sample words which are labeled with keyword or non-keyword for implementing the keyword extraction system are illustrated in Figure 5. The keyword extraction is viewed into the binary classification where each word is classified into keyword or non-keyword. The topic based word classification belongs to the domain independent classification where the word is classified identically with regardless of domain, whereas the keyword extraction belongs the domain dependent one where the word is classified differently depending on the domain. Domain by domain, words are collected randomly and labeled manually with keyword or non-keyword. Presenting domain of input text by its tag is required for executing the keyword extraction in the system.

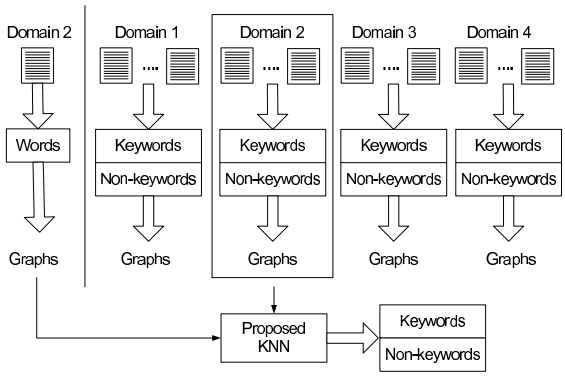

Figure 5. Sample Words

The entire architecture of the proposed keyword extraction system is illustrated in Figure 6. A text is given as the input, and words are extracted from it in the indexing module. The sample words in the keyword group and the non-keyword group and ones which are indexed from the text are mapped into graphs in the encoding module. The words which indexed from the text are classified into one of the two categories in the similarity computation module and the voting module. The system consists of the four modules: the indexing module, the encoding module, the similarity computation module, and the voting module.

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

Let us make some remarks on the proposed system which is illustrated in Figure 6 as the architecture. The keyword extraction is defined as the binary classification where each word is classified into keyword or non-keyword. Each word is encoded into a graph instead of a numerical vector and

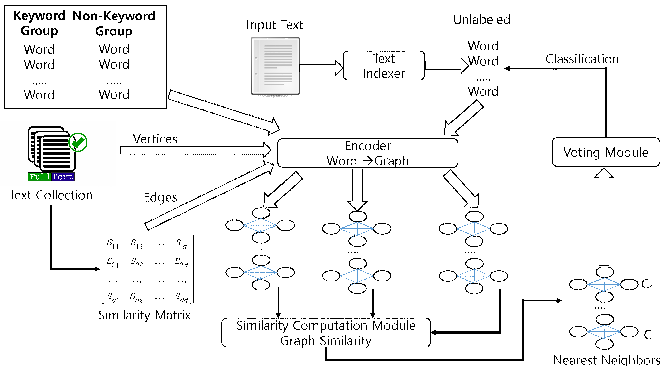

Figure 6. Proposed System Architecture

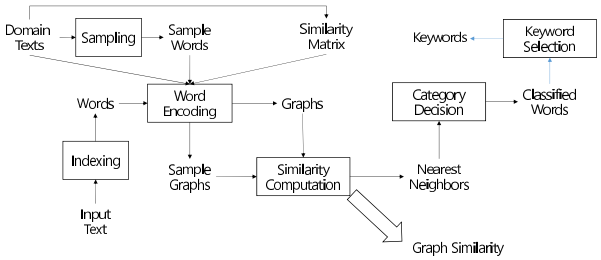

Figure 7. Execution Process of Proposed System

a graph is classified directly. Among words included in the input text, ones which are classified into keyword are selected as keywords of the input text. We need to add the text classification module, in order to predict the domain of the input text.

IV. Experiments

This section is concerned with the empirical experiments for validating the proposed version of KNN, and consists of the four sections. In Section IV-A, we present the results from applying the proposed version of KNN to the keyword extraction on the collection, NewsPage.com. In Section IV-B and IV-C, we mention the results from comparing the two versions of KNN with each other in the task of keyword extraction from 20NewsGroups.

A. NewsPage.com

This section is concerned with the experiments for validating the better performance of the proposed version on the collection: NewsPage.com. We interpret the keyword extraction into the binary classification where each word is classified into keyword or non-keyword, and gather words

which are labeled with one of the two categories, from the collection, topic by topic. Each word is allowed to be classified into one of the two labels, exclusively. We fix the input size as 50 of numerical vectors and graphs, and use the accuracy as the evaluation measure. Therefore, this section is intended to observe the performance of the both versions of KNN in the four different domains.

In Table I, we specify NewsPage.com which is used as the source for extracting the classified words, in this set of experiments. The text collection was used for evaluating approaches to text categorization, in previous works [7]. In each topic, we extracted 125 words labeled with keyword, and 125 words labeled with non-keyword. The set of 250 words in each topic is partitioned into the 200 words as training ones and the 50 words as the test ones, keeping the complete balanced distribution over the two labels, as shown in Table I. In building the test collection of words, we decide whether each word is a keyword or not, depending on its frequency concentrated in the given category combining with the subjectivity, in scanning articles.

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

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

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

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

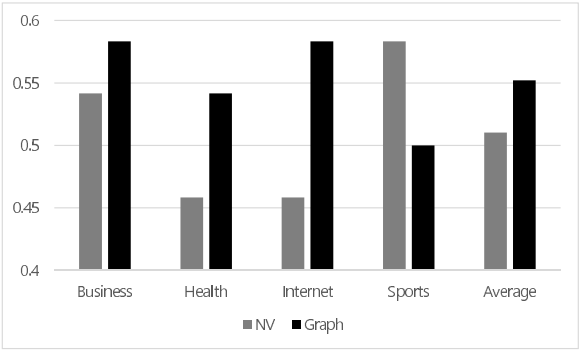

Figure 8. Results from Recognizing Keywords in Text Collection: NewsPage.com

Let us make the discussions on the results from doing the keyword extraction using the both versions of KNN algorithm, as shown in Figure 8. The accuracy which is the performance measure of the classified task is in the range between 0.45 and 0.58. The proposed version of the KNN algorithm works better in the two domains: Health and Internet. It matches with the traditional version in the domain, Business, but is lost in the domain, Sports. However, from this set of experiments, we conclude the proposed version works better than the traditional one, in averaging over the four cases.

B. 20NewsGroups I: General Version

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

In Table II, we specify the general version of 20NewsGroups which is used for evaluating the two versions of KNN algorithm. In 20NewsGroup, the hierarchical classification system is defined with the two levels; in the first level, the six categories, alt, comp, rec, sci, talk, misc, and soc, are defined, and among them, the four categories are selected, as shown in Table II. In each category, we select 1000 texts at random and extract 250 words from them. Among the 250 words, one half of them is labeled with ’keyword’, and the other half is labeled with ’non-keyword’. As shown in

Table II, the 250 words is partitioned into the 200 words in the training set, and the 50 words in the test set, keeping the complete balance over them. In the process of gathering the classified words, each of them is labeled manually into one of the two categories by scanning individual texts.

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

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

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

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

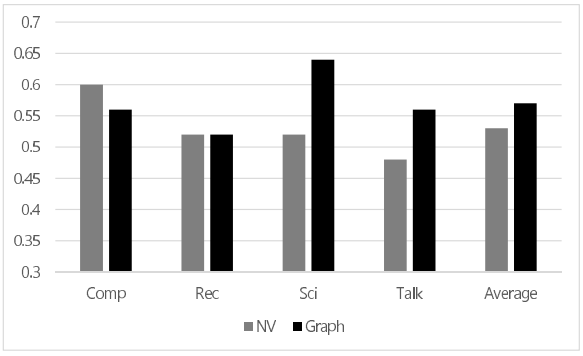

Figure 9. Results from Recognizing Keywords in Text Collection: 20NewsGroup I

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

C. 20NewsGroups II: Specific Version

This section is concerned with one more set of experiments where the better performance of the proposed version is validated on another version of 20NewsGroups. We gather the words which are labeled with 'keyword' or 'non-keyword'. We map the keyword extraction into a binary classification, and carry out the independent four binary classification tasks as many as topics, in this set of experiments. We fix the input size in representing words to 50, and use the accuracy as the evaluation metric. Therefore, in this section, we observe the performances of the both versions of the KNN with the four different domains.

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

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

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

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

We present the experimental results from classifying the words using the both versions of KNN algorithm on the specific version of 20NewsGroups. The frame of illustrating the classification results is identical to the previous ones. In each group, the gray bar and the black bar stand for

the achievements of the traditional version and the proposed version, respectively. The y-axis in Figure 10, indicates the classification accuracy which is used as the performance metric. In this set of experiments, we execute the four independent classification tasks which correspond to their own domains, where each word is classified into 'keyword' or 'non-keyword'.

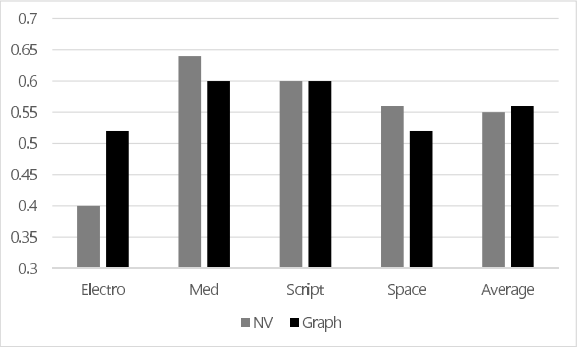

Figure 10. Results from Recognizing Keywords in Text Collection: 20NewsGroup II

Let us discuss on the results from doing the keyword extraction on the specific version of 20NewsGroups, as shown in Figure 10. The accuracies of both versions of KNN algorithm range between 0.40 and 0.64. The proposed version shows its better results in the domain, 'electro'. It shows its comparable one in two of the four domains, and it is leaded in the other. From this set of experiments, it is concluded that the proposed version is slightly better by averaging over the accuracies of the four domains.

V. Conclusion

Let us discuss the entire results from extracting keywords using the two versions of KNN algorithm. The both versions are compared with each other in the task of word classification which is mapped from the keyword extraction, in these sets of experiments. The proposed version shows its better results in all of the three collections. The accuracies of the traditional version range between 0.40 and 0.64 and those of the proposed version range between 0.51 and 0.64. From the three sets of experiments, we conclude the proposed version improved the keyword extraction performance as the contribution of this research.

Let us mention some remaining tasks for doing the further research. We need to validate more the proposed approach in extracting keywords in specific domains such as medicine, engineering, and economics, and customize it correspondingly. We need to consider other schemes of encoding words into graphs and other similarity measures between graphs. We modify other machine learning algorithms into their graph based versions where a graph is given by itself as the input data. We implement a keyword extraction system by adopting the proposed approach.

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] D. Allemang and J. Hendler, Semantic Web for the Working Ontologies, Mrgan Kaufmann, 2011.

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

[4] 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.

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

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

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

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

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

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

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

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

[16] 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.

[17] 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.

[18] 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.

[19] 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.

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

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

[22] 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.

[23] 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.

[24] 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.

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

[26] 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.

[27] 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.

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

[29] 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.

[30] N.F. Noy and C. D. Hafner, ”State of the Art in Ontology Design”, AI Magazine, Vol 18, No 3, 1997.

[31] 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.