Content based Text Segmentation using Feature Similarity based K Nearest Neighbor

Taeho Jo
President
Alpha AI Publication
Cheongju, South Korea
tjo018@naver.com

***Abstract*—This article proposes the modified KNN (K Nearest Neighbor) algorithm which considers the feature similarity and is applied to the text segmentation. The words which are given as features for encoding words into numerical vectors have their own meanings and semantic relations with others, and the text segmentation is able to be viewed into a binary classification where each adjacent paragraph pair is classified into boundary or continuance. In the proposed system, a list of adjacent paragraph pairs is generated by sliding a text with the two sized window, each pair is classified by the proposed KNN version, and the boundary is put between the pairs which are classified into boundary. The proposed KNN version is empirically validated as the better approach in deciding whether each pair should be separated from each other or not in news articles and opinions. The significance of this research is to improve the classification performance by utilizing the feature similarities.**

I. Introduction

Text segmentation refers to the process of segmenting an article into its several parts based on its content. Because in the information retrieval systems, a long text tends to be retrieved most frequently by overestimation of its relevancy to a query, we need to segment it into its several parts, in order to avoid the problem. In this task, the text is given as the input and segmented into paragraphs, a list of pairs of adjacent paragraphs is generated, and each pair is judged whether we put the topic boundary between them, or not. The task is interpreted into a binary classification where each pair of paragraphs is classified into separation or non-separation. However, in next research, it will be considered to segment speech text into paragraphs or sentences.

Some problems are caused by encoding texts into numerical vectors and computing their similarities based on only attribute values. Many features are required for encoding texts into numerical vectors, assuming that words are given as features, in order to maintain the enough system robustness [2]. The dominance of zero feature values in each numerical vector causes the very poor environment for computing their similarities because of very weak discriminations among numerical vectors [2]. In the previous works, the similarity between numerical vectors representing texts has been computed, assuming the independence among features, even if the words which indicate the features have their very strong semantic relations [1]. Therefore, in this research, as the challenge against the problems, we consider both the semantic relations among features and differences among feature values for computing the similarity between two texts.

Let us mention what we propose in this research as some agenda. In this research, we assume that words are given as features of numerical vectors in encoding texts, and they have their semantic relations with others. Based on the assumption, we define the similarity measure for computing the similarity between feature vectors, considering both feature values and features. We modify the KNN into the version where both the feature similarity and the feature value similarity are used, and apply it to the classification task mapped from the text segmentation. As benefits from this research, we expect its more tolerance to the sparse distributions and the potential avoidance of the huge dimensionality.

Let us mention what is expected from this research as benefits by implementing the above ideas. We may cut down the dimensionality in encoding texts into numerical vectors, potentially. The information loss in computing the similarity between texts may be reduced by reflecting the similarities among the features. The proposed approach becomes less sensitive to the sparse distribution of numerical vectors, because the similarity among features is captured as well as among feature values. Therefore, we expect both the better performance of the classification task which is mapped from the text segmentation and the more efficient text representations, from this research.

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 **??**, we survey previous works on semantic operations. In Section II-C and II-D, we survey previous works on the semantic word similarity and the semantic word clustering which are relevant to the process of computing the feature similarity. Therefore, in this section, we provide the history about this research, by surveying the relevant previous works.

A. Applications to Related Tasks

This section is concerned with the text segmentation and its related tasks where the modernized KNN algorithm is applied. We mention the text categorization as the base task to which the modernized KNN algorithm is applied and the two tasks are derived from it. We mention the text summarization as a task which is derived from the text categorization, and explore the cases of applying the modernized KNN algorithm for it. We consider the text segmentation which is covered in this research as another task which is derived from the text categorization, and present the cases of applying the modernized KNN algorithm for it. This section is intended to survey the previous cases of applying the modernized KNN algorithm for the text categorization and its derived tasks.

Let us present the previous cases of applying the KNN algorithm which is modernized by considering the feature similarity for the text categorization as the base ask for the text segmentation. In 2018, it was initially proposed that the KNN algorithm should be modified by doing so [21]. In 2019, the modernized KNN algorithm was compared with the traditional version in categorizing texts into one or some among the predefined categories in a small text collection [27]. Its better performance was validated in the text categorization in the three text collections, in 2019 [28]. In the above literatures, we present the effectiveness of the modified KNN algorithm in the text categorization.

Let us survey the cases of applying the modernized KNN version for the text summarization as a relevant task. It was initially asserted that the modernized version should be used for the text summarization by Jo in 2017 [19]. The modernized version was compared with the traditional version, and its better performance was initially observed in summarizing texts in 2018 [22]. Validating empirically its better performance in the text summarization was recently finalized, but it is not published yet [30]. In the above literatures, we mention the application of the modernized KNN version to the text summarization and its better performance.

Let us mention the previous works where the proposed KNN algorithm is applied to the text segmentation. Its application to the task which is viewed as a binary classification was initiated by Jo in 2017 [20]. Its better performance was discovered in comparing it with the traditional version in the text segmentation, in 2019 [29]. This research is aimed to finalize the empirical validation of the proposed version as a more desirable approach to the text segmentation. In the above literatures, the text segmentation is viewed into a binary classification, where each adjacent paragraph pair is classified into boundary or continuance.

We surveyed the previous cases of applying the proposed KNN version to the three relevant tasks. The text segmentation which is covered in this research is the process of setting a boundary between paragraphs as a topic transition. The KNN version which is used as the approach to the text segmentation is one where the similarities of a novice example with the training ones are computed, considering the feature similarities. The KNN version which is adopted in this research was applied in the above literatures to the text categorization and the text summarization, as well as the text segmentation. Even if the research about the modified KNN version for the text segmentation has progressed, we need to complete the empirical validation of its better performance through the real experiments.

B. Semantic Operations

This section is concerned with the previous works on the semantic operations on strings. They are ones on strings based on the meanings, under the assumption of each string with its own meaning. In surveying the previous works, we mention the three operations; the semantic similarity as the base operation, the semantic similarity average as the derived one, and the semantic string set which generates a string set. In this research, the semantic similarity is used for computing the similarities among features which are given as words. This section is intended to explore the previous works on the three semantic similarities, in order to provide the background for defining the feature similarities.

Let us survey the previous works where the semantic similarity is defined as the base semantic operation, and used for modifying the existing machine learning algorithms and creating a new one. The string vector kernel was proposed by modifying the SVM (Support Vector Machine) by expanding the operation in 2008 [3]. The semantic similarity between strings was used for creating the unsupervised neural networks, called NTSO (Neural Text Self Organizer), as the approach to the text clustering, in 2010 [4]. Using the base semantic operation, the KNN algorithm was modified as the approach to the text categorization [23]. In the above literatures, we present that the semantic similarity is defined and used for modifying the machine learning algorithms.

The semantic operation which is called SSM (Semantic Similarity Mean) was initially defined by Jo in 2015 [5]. The SSM is the semantic operation for generating the average over the semantic similarities of all possible pairs of strings. Any number of strings is taken as a group, the semantic similarities of all possible pairs are computed, and they are averaged. The averaged semantic similarity is given as a normalized value between zero and one, and indicates the semantic cohesion of the strings. It takes the quadratic complexity to the number of strings for this operation.

Let us mention the semantic operation on strings which generates a string set, as the output. The operation was initially defined for creating an unsupervised neural networks, called NTSO (Neural Text Self Organizer), as the approach to the text clustering, in 2010 [4]. A set of string which are more semantically similarity than the semantic similarity between the two input strings is generated by the operation. It is used for updating weights of the neural networks which are given as the strings. More mathematical characterizations are needed for modifying existing neural networks into their string vector based versions or creation string vector based deep learning algorithms.

We surveyed the previous works on the semantic operations among which it is used for computing the feature similarities. The semantic similarity between two strings is the base semantic operation on them. The semantic similarity may expanded into the SSM for analyzing the semantic cohesion among strings. The semantic string set generates strings with their more similarities than one between two input strings, as a set. In this research, the base semantic operation will be used for computing the feature similarity.

C. Word Similarities

This section is concerned with the previous works on the schemes of computing a similarity between words. It is assumed that words are given as features in encoding texts into numerical vectors, so in this research, we need the similarity metric between words for computing the feature similarities. In the previous works which are surveyed in this section, the similarity between words is computed by encoding them into structured data. We use the proposed similarity metric between numerical vectors for modifying the KNN algorithm as the approach to the text segmentation. This section is intended to survey the previous works on the process of computing the similarity between words as the feature similarity.

Let us survey the previous works where the similarity between words is computed by encoding them into tables. The KNN algorithm where the similarity between words is computed so was initially proposed as the approach to the word categorization, in 2016 [7]. The AHC algorithm was modified into its table based version where the similarity between words is computed by doing so, as the approach to the text clustering, in 2016 [8]. The table based version of the KNN algorithm was applied to the keyword extraction which is mapped into a binary classification of each word into keyword or non-keyword, in 2016 [9]. In the above literatures, we presented the similarity metric between tables as one between words.

Let us mention the previous works on the computation of the similarity between words by encoding them into string vectors. In 2016, the similarity between string vectors was defined as one between words, and used for modifying the KNN algorithm as the approach to the word categorization [10]. In 2016, the AHC algorithm was modified using the similarity between string vectors, as the approach to the word clustering [11]. The modified KNN algorithm was applied to the keyword extraction which is mapped into the binary classification of words into keyword or non-keyword [12]. In the above literatures, we mention the scheme of computing the similarity between words by encoding them into string vectors.

Let us survey the previous works on the computation of the similarity between words by encoding them into graphs. The KNN algorithm was modified into the graph based version as the approach to the word categorization, using the similarity between graphs as one between words, in 2016 [13]. The modified KNN algorithm is applied to the keyword extraction which is mapped into the binary classification of each word into keyword or non-keyword, in 2016 [14]. The AHC algorithm was modified so using the similarity between graphs as the approach to the word clustering, in 2016 [15]. In the above literatures, we present the process of computing the similarity between words by encoding them into graphs.

We surveyed the previous works on the computation schemes of the similarity between words. Because words are given as features in encoding texts into numerical vectors, we need the scheme of computing the similarity between words for doing the feature similarity. In the previous works which are surveyed above, it is required to encode words into tables, string vectors, or graphs, for computing the similarity between words. The similarity metrics between words which are defined in the above literatures are used for modifying the KNN algorithm and the AHC algorithm as the approaches to the word categorization and the word clustering. In this research, the similarity between words is computed as the feature similarity based on their collocations in same texts.

D. Semantic Word Clustering

This section is concerned with the previous works on the semantic word clustering as an expansion of the semantic word similarity. In Section II-C, we already surveyed the schemes of computing the semantic similarity between words. The semantic word similarity is expanded into the semantic word clustering, and the previous works on it are explored. Even if the semantic word association is considered as another expansion of the word similarity, it is excluded from the scope of this section. This section is intended to survey the previous works on the semantic word clustering.

Let us survey the previous works on clustering words by encoding them into tables. The table based AHC algorithm where the similarity between words is computed by encoding them into tables was initially proposed as the approach to the word clustering, in 2016 [16]. Its better clustering performance was discovered in a toy experiment of text clustering in 2018 [24]. The research on the empirical validation of its clustering performance in the real experiments is progressed, currently [31]. In the above literatures, we present the proposal and the validation of the table based version of the AHC algorithm as the approach to the text clustering.

Let us explore the progress of research on the word clustering, depending on the similarity between string vectors. The string vector based AHC algorithm which processes string vectors directly was initially proposed as the approach to the text clustering, in 2016 [17]. Its better performance of the string vector based version than the traditional version was discovered in toy experiments on text clustering, in 2018 [25]. The research on the final validation of the better clustering performance on real experiments is progressing, currently [32]. We present the progress of the research on the string vector based AHC algorithm which was proposed as the approach to the text clustering.

Let us review the previous works on the word clustering based on the similarity between graphs. The version of AHC algorithm which processes graphs directly was initially proposed as the approach to the text clustering, in 2016 [18]. Its better performance, compared with the traditional version which processes numerical vectors directly, was discovered n toy experiments, in 2018 [26]. The research on the complete validation of its clustering performance in the real experiments progresses currently [33].. In the above literatures, we present the progress of research on the word clustering by encoding words into graphs.

We explored the previous works on the word clustering as the expansion of the word similarity. Words were encoded into tables, graphs, and string vectors, in the above literatures. The AHC algorithm was modified into the three versions: the table base version, the string vector based version, and the graph based version. The research about each AHC version progressed with three steps: the initial version, the initial validation on the toy experiments, and final validation on the real experiments. The word clusters instead of individual words may be considered as features in encoding texts into numerical vectors.

III. Proposed Approach

This section is concerned with the KNN (K Nearest Neighbor) algorithm which considers both the feature similarity and feature value one, and it consists of the four sections. In Section III-A, we describe the process of encoding texts into numerical vectors as the text preprocessing. In Section III-B, we present the equation with which the similarity between two numerical vectors is computed, considering the feature similarity. In Section III-C, we mention the proposed version where the similarity is computed by the proposed scheme with respect to its learning process. In Section III-D, we present the system architecture and the execution flow of the proposed system.

A. Text Encoding

This section is concerned with the process of mapping a text into a numerical vector. The features of a numerical vector which represents a text are given as words and each feature value indicates the relationship between a word and a text. The feature extraction, the feature selection, and the feature value assignment are the steps of encoding a text into a numerical vector. Because the numerical vectors which represent texts tend to be sparse, this research will challenge against it. This section is intended to explain the steps of encoding a text into a numerical vector.

The three steps of mapping texts into a corpus into words which are given as feature candidates are illustrated in Figure 1. Tokens are generated by segmenting a text by the white space or the punctuation marks, removing special characters in the tokenization. Grammatical variations are removed by mapping plural nouns into singular ones and mapping variated verbs into their root forms, in the stemming. Stopwords which function only grammatically, irrelevantly to the text contents; they belong to the preposition, the conjunction, and pronouns. After the steps which are presented in Figure 1, nouns, verbs, and adjectives remain, and they are given as feature candidates.

The process of selecting some words as the features with the three steps is illustrated in Figure 2. The N words which are generated as the feature candidates by the process which is illustrated in Figure 1, are prepared. A weight is assigned to each of the N words, and they are ranked by their weights. The d words with their maximal weights are selected as the features in the third step in Figure 2. The number of words which are selected as features, d is the dimension of the numerical vector which represents a text.

The assignment of the three kinds of values to the selected features is illustrated in Figure 3. A corpus is indexed into words as the feature candidates, and the d words are selected as the features in the previous steps. There are the three schemes of assigning values to the features: the binary value which indicates the presence or the absence of each word in the text, the frequency of each word in the text, and the TF-IDF weight of each word Term Frequency and Inverse Document Frequency. The three kinds of values indicate the relationship between the text as the encoding target and the word which corresponds to a feature. The words become the features in the d dimensional numerical vector which represents a text.

Let us make some remarks on the process of encoding texts into numerical vectors. Words are used as features in encoding texts into numerical vectors; the feature candidates

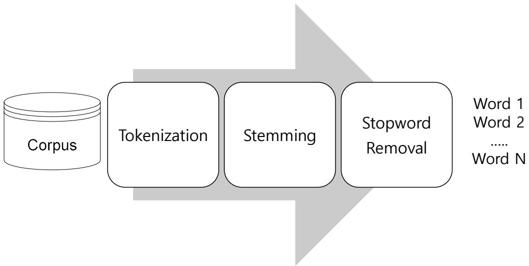

Figure 1. The Process of indexing a Text

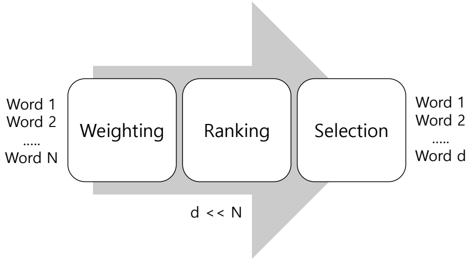

Figure 2. The Process of selecting Features

are generated by indexing a corpus, and only some are selected among them as features. The value which is assigned to each feature indicates the relationship between a word as a feature and a text as an encoding target. The TF-IDF weight is adopted as the feature value in this research; the weight is assigned to each feature in the experiments in Section 4. It costs the higher complexity for computing a similarity between numerical vectors, as the payment.

B. Similarity Metric

This section is concerned with the process of computing the similarity between two numerical vectors. The inverse Euclidean distance and the cosine similarity were mentioned as the traditional similarity metric. In this research, we propose the similarity metric which considers the feature similarities, in order to avoid the fragility to the sparseness of the numerical vectors. The similarity metric is used for modifying the KNN algorithm as the approach to the text segmentation. This section is intended to describe the proposed similarity metric between numerical vectors, in detail.

The frame of computing a similarity between two vectors is illustrated in Figure 4. The two numerical vectors and the features are notated by $\mathbf{x}_1 = [x_{11}\ \ x_{12}\ \ \ldots\ \ x_{1d}]$, $\mathbf{x}_2 = [x_{21}\ x_{22}\ \ldots\ x_{2d}]$, and $f_1, f_2, \ldots, f_d$. The feature similarity, $sim(f_i, f_j)$, is defined as the similarity among features, and the feature value similarity, $sim(x_{1i}, x_{2j})$, stands for the similarity between x_{1i} and x_{2j}. We consider the two kinds of similarities, the feature similarities and the feature value ones, for computing the similarity between two vectors. The feature similarity,$sim(f_i, f_j)$ is assumed to be always a normalized value between zero and one, as shown in equation (1),

$$0 \leq sim(f_i, f_j) \leq 1 \qquad (1)$$

The similarity matrix of the features which are given as texts is illustrated in Figure 5. The d words, $word_1, word_2, \ldots, word_d$, are selected as the features by the process which was described in Section III-A. The similarity between the two words, $word_i$ and $word_j$, is computed based on their collocations in same texts, by equation (2),

$$sim(word_i, word_j) = \frac{2 \cdot D(word_i, word_j)}{D(word_i) + D(word_j)} \qquad (2)$$

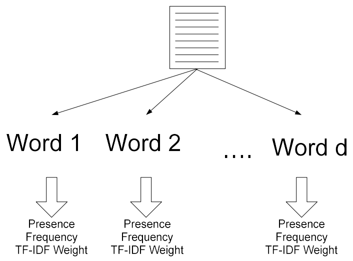

Figure 3. Three Kinds of Feature Values assigned to Selected Features

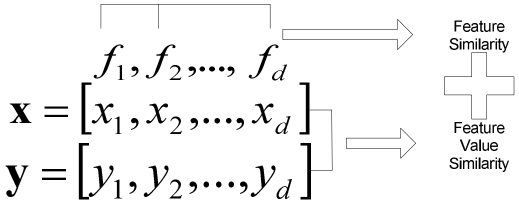
$f_1, f_2, \ldots, f_d$
$\mathbf{x} = [x_1, x_2, \ldots, x_d]$
$\mathbf{y} = [y_1, y_2, \ldots, y_d]$
Feature Similarity
Feature Value Similarity

Figure 4. Frame of Computing Similarity between Two Vectors

where $D(word_i, word_j)$ is the number of texts which include both $word_i$ and $word_J$ in the corpus, $D(word_i)$ is the number of texts which include $word_i$, and $D(word_j)$ is the number of texts which include $word_j$. The similarity between two words is always given as a normalized value between zero and one, and it is proportional to the number of texts which have the both words in the corpus. The similarity matrix is constructed as a $d \times d$ matrix by computing the similarities of the all possible pairs of the d words.

$$\begin{array}{c|cccc} & word_1 & word_2 & \cdots & word_d \\ \hline word_1 & s_{11} & s_{12} & \cdots & s_{1d} \\ word_2 & s_{21} & s_{22} & \cdots & s_{2d} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ word_d & s_{d1} & s_{d2} & \cdots & s_{dd} \end{array} \quad s_{ij} = sim(word_i, word_j)$$

Figure 5. Similarity Matrix

Let us derive the equation for computing the proposed similarity metric with the feature similarities. Equation (2) is notated into its simplified form as expressed in equation (3),

$$sim(word_i, word_j) = f_{ij} \tag{3}$$

The two words are encoded into the two d dimensional numerical vectors, $\mathbf{x} = [x_1\ x_2\ \ldots\ x_d]$ and $\mathbf{y} = [y_1\ y_2\ \ldots\ y_d]$. The similarity between the two numerical vectors is computed by equation (4),

$$sim(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{d} \sum_{j=1}^{d} f_{ij} \cdot x_i \cdot y_j}{d\|\mathbf{x}\|\|\mathbf{y}\|} \tag{4}$$

where $\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{d} x_i^2}$ and $\|\mathbf{y}\| = \sqrt{\sum_{i=1}^{d} y_i^2}$. It takes the quadratic complexity to the d dimensional vector for computing the similarity by equation (4), as the payment.

Let us make some remarks on the semantic similarity between numerical vectors which is proposed in this research. The two kinds of similarities, the feature similarities and the feature value similarities, as the frame of computing the similarity between numerical vectors. The $d \times d$ similarity matrix which consists of the feature similarities is constructed by computing the similarities of all possible pairs of the features by equation (2). The similarity metric between numerical vectors which represent texts is defined as equation (4). The semantic similarity metric is utilized for modifying the KNN algorithm as the approach to the text summarization.

C. Proposed Version of KNN

This section is concerned with the proposed version of the KNN algorithm as the approach to the text segmentation. In the previous section, we described the proposed metric between numerical vectors, considering the feature similarities and the feature value similarities. The KNN algorithm is modified, using the proposed similarity metric; the similarities of a novice item with the training ones is computed using the similarity metric in the proposed version of the KNN algorithm. The text segmentation is mapped into a binary classification of adjacent paragraph pairs, the proposed KNN is applied to the task. This section is intended to describe the proposed version of the KNN algorithm which is applied to the classification task which is mapped from the text segmentation.

Figure 6 illustrated that the similarities of a novice vector with the sample vectors are computed for selecting nearest neighbors. A novice text is encoded into the vector, $\mathbf{x}_{nov}$, the predefined categories are notated by $C = \{c_1, c_2, \ldots, c_{|C|}\}$, and the training set which consists of n sample vectors which represent the sample texts is notated by $Tr = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in C$. The similarities of the novice vector, $\mathbf{x}_{nov}$ with the sample vectors, $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, are computed by equation (3), as $sim(\mathbf{x}_{nov}, \mathbf{x}_1), sim(\mathbf{x}_{nov}, \mathbf{x}_2), \ldots, sim(\mathbf{x}_{nov}, \mathbf{x}_n)$ in the proposed KNN algorithm. The similarity between the novice vector, $\mathbf{x}_{nov}$, and a sample vector, is given as a normalized value between zero and one. The similarities, $sim(\mathbf{x}_{nov}, \mathbf{x}_1), sim(\mathbf{x}_{nov}, \mathbf{x}_2), \ldots, sim(\mathbf{x}_{nov}, \mathbf{x}_n)$ are ranked by their values for selecting nearest neighbors.

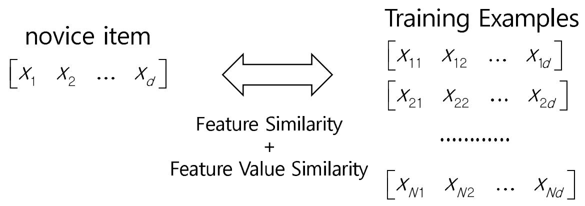
novice item
$[x_1 \ x_2 \ \cdots \ x_d]$
Training Examples
$[x_{11} \ x_{12} \ \cdots \ x_{1d}]$
$[x_{21} \ x_{22} \ \cdots \ x_{2d}]$
....
$[x_{N1} \ x_{N2} \ \cdots \ x_{Nd}]$
Feature Similarity
+
Feature Value Similarity

Figure 6. Similarities of a Novice Vector with Sample Ones

The process of selecting nearest neighbors after computing their similarities with the novice item is illustrated in Figure 7. The similarities which are computed by equation (3) are ranked into ones, $sim(\mathbf{x}_{nov}, \mathbf{x}'_1), sim(\mathbf{x}_{nov}, \mathbf{x}'_2), \ldots, sim(\mathbf{x}_{nov}, \mathbf{x}'_n)$. The K items with their highest similarities with the novice item are selected as its nearest neighbors, as expressed in equation (5),

$$Near(K, \mathbf{x}_{nov}) = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_K\} K \ll N \tag{5}$$

As an alternative way, we may consider selecting items with their higher similarities than a given threshold. We use the nearest neighbors, $\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_K$ from the training examples, for deciding the label of the novice vector, $\mathbf{x}_{nov}$.

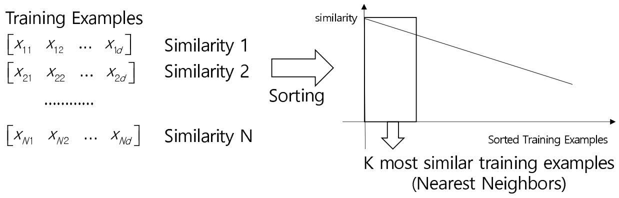
Training Examples
$[x_{11} \ x_{12} \ \cdots \ x_{1d}]$ Similarity 1
$[x_{21} \ x_{22} \ \cdots \ x_{2d}]$ Similarity 2
....
$[x_{N1} \ x_{N2} \ \cdots \ x_{Nd}]$ Similarity N
Sorting
similarity
Sorted Training Examples
K most similar training examples
(Nearest Neighbors)

Figure 7. Selection of Nearest Neighbors from Training Examples

The process of voting the labels of the nearest neighbors for deciding the label of the novice item is illustrated in Figure 8. The nearest neighbors are selected by the process which is illustrated in Figure 8, as a set, $Ne = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_K\}$, and the function for weighting a nearest neighbor by a category is defined as equation (6),

$$w(C_i, \mathbf{x}'_j) = \begin{cases} 1 & \text{if } \mathbf{x}'_j \in C_i \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

For each category, the number of nearest neighbors which belong it is counted as shown in equation (7),

$$Count(C_i, Ne) = \sum_{j=1}^{K} w(C_i, \mathbf{x}'_j) \tag{7}$$

The label of a novice item is decided by the label with the majority of the nearest neighbors, $C_{\max}$, as shown in equation (8),

$$C_{\max} = \underset{i=1}{\overset{|C|}{\text{argmax}}}\, Count(C_i, Ne) \tag{8}$$

The function, $w(C_i, \mathbf{x}'_j)$ may be expanded into $w(C_i, \mathbf{x}'_j, \mathbf{x}_{nov})$ by augmenting the novice item, if the weight is dependent on the distance between the nearest neighbor and the novice item.

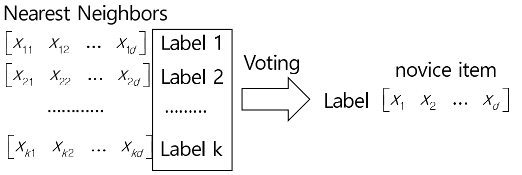
Nearest Neighbors
$[x_{11} \ x_{12} \ \cdots \ x_{1d}]$ Label 1
$[x_{21} \ x_{22} \ \cdots \ x_{2d}]$ Label 2
....
$[x_{k1} \ x_{k2} \ \cdots \ x_{kd}]$ Label k
Voting
novice item
Label $[x_1 \ x_2 \ \cdots \ x_d]$

Figure 8. Voting Labels of Training Examples for deciding One of Novice Example

Let us make some remarks on the proposed version of the KNN algorithm as the approach to the text segmentation. The similarity metric which was described in Section III-B is used for computing the similarities of the novice item with the sample items. The sample items are ranked by their similarities with the novice item, and the K samples with their highest similarities are selected as the nearest neighbors. The text segmentation is mapped into the binary classification of adjacent paragraph pair into boundary or continuance, and the proposed version of the KNN algorithm is applied to the task, in this research.

D. Text Segmentation System

This section is concerned with the system architecture and the execution flow of the text segmentation system. The text segmentation is mapped into the binary classification of adjacent paragraph pairs, and the KNN algorithm which is described in Section III-C is adopted as the approach. Adjacent paragraph pairs are generated by sliding the text which is given as the input, they are classified into boundary or continuance, and the boundaries are put between adjacent paragraph pair which are classified into boundary. Even if the system architecture and the execution flow for designing the system are presented, the implementation of the system which is given in Java or Python will be omitted. This section is intended to describe ones which are necessary for designing the system.

Gathering sample paragraph pairs and classifying a novice one is illustrated in Figure 9. Because even a same paragraph pair may be classified differently, depend on its domain, the classification task is called domain dependent task. The sample paragraph pairs are gathered domain by domain, and each pair is labeled with boundary or continuance within each domain. Paragraph pairs are taken from a text which is tagged with its domain, and each of them is classified into boundary or continuance. The text categorization is a domain independent task where a same item is classified identically, whereas the text segmentation is a domain dependent task where a same item may be classified differently depending on its domain.

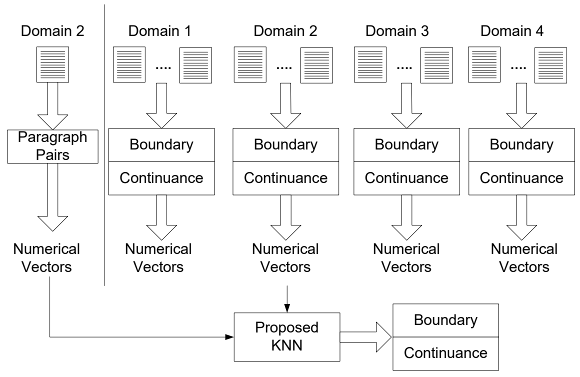

Figure 9. Process of Collecting Sample Paragraph Pairs

The system architecture of the text segmentation system is illustrated in Figure 10. The module of text partition & sliding partitions an input text into paragraphs and generates adjacent paragraph pair by sliding the two sized window, and the encoding module encodes the adjacent paragraph pairs into numerical vectors by the process which was described in Section III-A. The similarity computation module computes the similarity of a paragraph pair with the samples which are labeled with boundary or continuance, and selects some with their highest similarities as the nearest neighbors. The voting module votes the labels of the nearest neighbors, in order to decide one of the novice one. In this system, the adjacent paragraph pairs are classified into boundary or continuance by the KNN algorithm which was described in Section III-C, and the boundary is put between paragraphs in each pair which is classified into boundary.

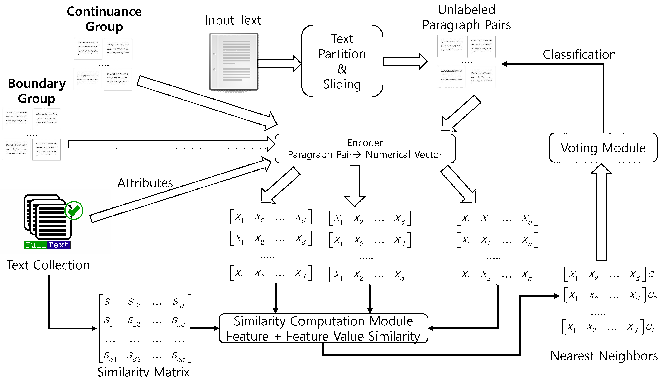
Continuance Group
Boundary Group
Input Text
Text Partition & Sliding
Unlabeled Paragraph Pairs
Classification
Encoder
Paragraph Pair→ Numerical Vector
Voting Module
Attributes
Full Text
Text Collection
$[x_1\ x_2\ \dots\ x_d]$
Similarity Computation Module
Feature + Feature Value Similarity
Similarity Matrix
$[x_1\ x_2\ \dots\ x_d]c_1$
$[x_1\ x_2\ \dots\ x_d]c_2$
$[x_1\ x_2\ \dots\ x_d]c_k$
Nearest Neighbors

Figure 10. System Architecture

The execution flow of the text segmentation system is illustrated in Figure 11. Texts are initially collected within a domain, adjacent paragraph pairs are extracted from texts, and they are labeled with boundary or continuance, manually. Novice adjacent paragraph pairs are extracted from an input text, and they are encoded into numerical vectors, together with the sample paragraph pairs. They are classified into boundary or continuance, and there are two groups of paragraph pairs in the text: the boundary group and the continuance group. A boundary is marked between paragraphs in each pair in the boundary group, and text is partitioned by the marked boundary into subtexts as the final output of this system.

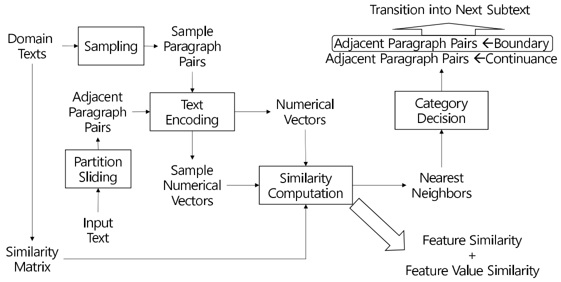

Figure 11. Execution Process

Let us make some remarks on the system architecture and the execution flow of the text segmentation system. In this research, the text segmentation is viewed into a binary classification of adjacent paragraph pairs, and the similarity metric between two numerical vectors which was described in Section III-B is proposed. The KNN algorithm is modified with the similarity metric, and adopted for implementing the text segmentation system. We present the system architecture and the execution flow of the system; this indicates that this research stays in the general design step of the system. In the next research, we consider the detail design and the implementation of the entire text summarization system.

IV. Experiments

This section is concerned with the empirical experiments for validating the proposed version of KNN, and consists of the five sections. In Section IV-A, we present the results from applying the proposed version of KNN to the text segmentation on the collection, NewsPage.com. In Section IV-B, we show the results from applying it for classifying paragraph pairs into boundary or continuance, from the collection, Opinosis. In Section IV-C and IV-D, we mention the results from comparing the two versions of KNN with each other in the task of text segmentation 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 text segmentation into the binary classification where each adjacent paragraph pair is classified into boundary and continuance, and, by sliding window on paragraphs of each text, gather the paragraph pairs which are labeled with one of the two categories, from the collection, topic by topic. Each paragraph pair is classified exclusively into one of the two labels. We fix the input size as 50 dimensions of numerical vectors, and use the accuracy as the evaluation measure. Therefore, this section is intended to observe the performance of the both versions of KNN in the four different domains.

In Table I, we specify the text collection, NewsPage.com, which is used in this set of experiments. The collection was used for evaluating approaches to text categorization tasks in previous works [6]. In each category, we extract 250 adjacent paragraph pairs and label them with boundary or continuance, keeping the complete balance over the two labels. In each category, the set of 250 paragraph pairs is partitioned into the training set of 200 ones and the test set of 50 ones. Each text is segmented into paragraphs by a carriage return, and adjacent paragraph pairs are generated by sliding two sized window on the list of paragraphs.

Table I
The Number of Texts and Paragraph Pairs in NewsPage.com

Category	#Texts	#Training Pairs	#Test Pairs
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 for validating empirically the proposed approach to the task of text segmentation. We collect the sample paragraphs which are labeled with boundary or continuance in each of the four topics: Business, Sports, Internet, and Health, and encode them into numerical vectors. For each of 50 examples, the KNN computes its similarities with the 200 training examples, and selects the three similarity training examples as its nearest neighbors. This set of experiments consists of the four independent binary classifications each of in which each paragraph is classified into one of the two labels by the two versions of KNN algorithm. We compute the classification accuracy by dividing the number of correctly classified test examples by the number of test examples, for evaluating the both versions.

In Figure 12, we illustrate the experimental results from classifying each adjacent paragraph pair into boundary or continuance, 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 means the domain within which the text summarization which is viewed as a binary classification is performed, independently. In each group, the gray bar and the black bar indicate the accuracies of the traditional version and the proposed version of the KNN algorithm. The most right group in Figure 12 consists of the averages over the accuracies of the left four groups, and the input size which is the dimension of numerical vectors is set to 50.

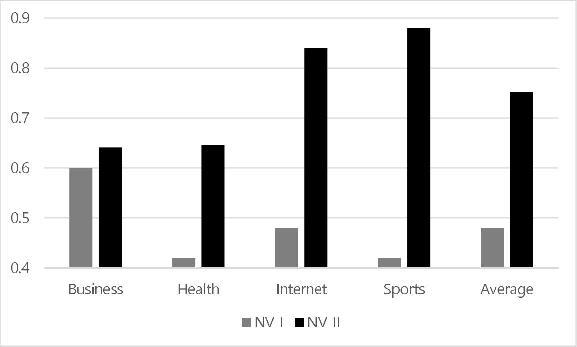

Figure 12. Results from Segmenting Texts in Text Collection: NewsPage.com

Let us make the discussions on the results from doing the text segmentation, using the both versions of KNN algorithm, as shown in Figure 12. The accuracy which is the performance measure of this classification task is in the range between 0.4 and 0.9. The proposed version of KNN algorithm works strongly better in the three domains, Health, Internet, and Sports. However, it loses in the domain, Business. In spite of that, from this set of experiments, we conclude the proposed version works better than traditional one, in averaging over the four cases.

B. Opinopsis

This section is concerned with the set of experiments for validating the better performance of the proposed version on the collection, Opinosis. We view the text segmentation into a binary classification where each adjacent paragraph pair is classified into boundary or continuance, and collect the paragraphs pairs, sliding paragraphs in each text by two sized window and labeling manually with one of boundary and continuance from the collection. Each paragraph pair is exclusively classified into one of the two labels. We fix the input size to 50 and use the accuracy as the evaluation measure. In this section, we observe the performance of the both versions of KNN algorithm, in the three experiments as many as topics.

In Table II, we specify the text collection, Opinosis, which is used in this set of experiments. The test collection is used in previous works for evaluating approaches to text categorization. We extract the 50 adjacent paragraph pairs in each topic, and label them with 'boundary' or 'continuance', keeping the complete balance. The set of 50 paragraph pairs is portioned into the 40 as the training set and the 10 as the test set, in each topic. In the process of generating the paragraph pairs, each text is segmented into paragraphs by the carriage return, the adjacent paragraph pairs are generated by sliding the paragraphs.

Table II
THE NUMBER OF TEXTS AND PARAGRAPH PAIRS IN OPINIOPSIS

Category	#Texts	#Training Pairs	#Test Pairs
Car	23	40 (20+20)	10 (5+5)
Electronic	16	40 (20+20)	10 (5+5)
Hotel	12	40 (20+20)	10 (5+5)

We perform this set of experiments by the process which is described in section IV-A. We collect sample adjacent paragraph pairs which are labeled with 'boundary' and 'continuance' in each of the three domains: 'Car', 'Electronics', and 'Hotel', and we encode them into 50 sized numerical vectors. For each test example, the both versions of KNN computes its similarities with the 40 training examples and select the three most similar training examples as its nearest neighbors. Each test example is classified into 'boundary' or 'continuance' by the two versions of KNN algorithm; we performed the three independent experiments as many as the domains. The classification accuracy is computed by the number of correctly classified test examples by the number of the test examples for evaluating the both versions of KNN algorithm.

In Figure 13, we illustrate the experimental results from the text segmentation which is mapped into a classification task, using the both versions of KNN algorithm. Like Figure 12, the y-axis indicates the value of accuracy, and the x-axis indicates the group of two versions by a domain of Opniopsis. In each group, the gray bar and the black bar indicate the results of the traditional version and the proposed version of KNN algorithm. In Figure 13, the most right group indicates the averages of the both version over their results of the left three groups. Therefore, Figure 13 shows the results from classifying adjacent paragraph pairs into one of 'boundary', and 'continuance', by the both versions.

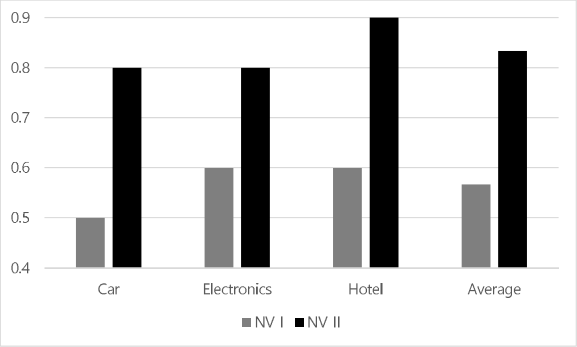

Figure 13. Results from Segmenting Texts in Text Collection: Opiniopsis

We discuss the results from doing the text segmentation which is mapped into a binary classification, using the both versions of KNN algorithm, shown in Figure 13. The accuracy values of the both versions range between 0.5 and closely to 0.9. The proposed version works better than the traditional one in the all domains. The accuracy of the proposed version reaches closely to 0.9, in the domain, Hotel. From this set of experiments, we conclude that the proposed one works outstandingly better in averaging the three cases.

C. 20NewsGroups I: General Version

This section is concerned with one more set of experiments for validating the better performance of the proposed version on text collection, 20NewsGroup I. We gather adjacent paragraph pairs which are labeled with 'boundary' or 'continuance', from each broad category of 20NewsGroups I, by viewing the text segmentation into a binary classification. The task of this set of experiments is to classify each paragraph pair exclusively into one of the two labels in each topic which is called domain. We fix the input size to 50 in encoding paragraph pairs and use the accuracy as the evaluation measure. Therefore, in this section, we observe the performances of the both versions in the four different domains.

In Table III, we specify the general version of 20NewsGroups which is used for evaluating the two versions of KNN algorithm. In 20NewsGroup, the hierarchical classification system is defined with the two levels; in the first level, the six categories, alt, comp, rec, sci, talk, misc, and soc, are defined, and among them, the four categories are selected, as shown in Table III. In each category, we extract 250 adjacent paragraph pairs from 4000 or 5000 texts; the first half is labeled with ‘boundary’, and the other half is labeled with ‘continuance’. The 250 paragraphs pairs is partitioned into the 200 ones in the training set and the 50 ones in the test sets, as shown in Table III. In the process of gathering the classified paragraph pairs, each of them is labeled manually into one of the two categories by scanning individual texts.

Table III
The Number of Texts and Paragraph Pairs in 20NewsGroups I

Category	#Texts	#Training Pairs	#Test Pairs
Comp	5000	200 (100+100)	50 (25+25)
Rec	4000	200 (100+100)	50 (25+25)
Sci	4000	200 (100+100)	50 (25+25)
Talk	4000	200 (100+100)	50 (25+25)

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

In Figure 14, we illustrate the experimental results from deciding whether we put a boundary, or not, between two adjacent paragraphs, on the broad version of 20NewsGroups. Figure 14 has the identical frame of presenting the results to those of Figure 12 and 13. 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. In the x-axis, each group indicates the domain within which each paragraph pair is classified into ‘boundary’, or ‘continuance’. This set of experiments consists of the four binary classifications in each of which it is done so.

Let us discuss the results from doing the text segmentation using the both versions of KNN algorithm as shown in Figure 14. The accuracies of both versions range between 0.45 and 0.63. The proposed version shows its better performances in all of the four domains. It shows its outstanding difference from the traditional version in the domain, comp. From this set of experiments, the proposed version wins over the traditional one, certainly, in averaging its achievements of the four domains.

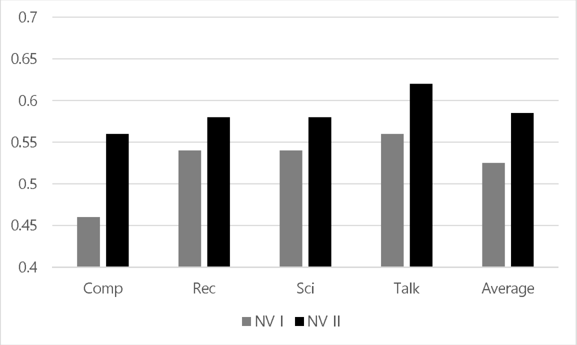

Figure 14. Results from Segmenting Texts in Text Collection: 20NewsGroup I

D. 20NewsGroups II: Specific Version

This section is concerned with one more set of experiments where the better performance of the proposed version is validated on another version of 20NewsGroups. From each specific topic, separately, we gather the adjacent paragraph pairs which are labeled with ‘continuance’ or ‘boundary’. In this set of experiments, we view the text segmentation into a binary classification, and carry out the four binary classifications, independently of each other. We fix the input size of representing the paragraph pairs to 50 and use the accuracy as the evaluation metric. Therefore, in this section, we observe the performances of the both versions of KNN algorithm in the four different domains.

In Table IV, we specify the specific version of 20NewsGroups which is used as the test collection, in this set of experiments. Within the general category, sci, we predefine the four categories: ‘electro’, ‘medicine’, ‘script’, and ‘space’. In each topic, we extract 250 adjacent paragraph pairs from approximately 1000 texts and label each of them with ‘boundary’ or ‘continuance’, maintaining the complete balance. The set of 250 paragraph pairs is partitioned into the training set of 200 ones and the test set of 50 ones, as shown in Table IV. We use the accuracy as the metric for evaluating the results from classifying them.

Table IV
The Number of Texts and Paragraph Pairs in 20NewsGroups II

Category	#Texts	#Training Pairs	#Test Pairs
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 gather sample paragraph pairs which are labeled with ‘boundary’ or ‘continuance’, in each of the four domains: ‘electro’, ‘medicine’, ‘script’, and ‘space’, and encode them with the fixed input size: 50. We use the two versions of KNN algorithm for their comparisons. Each test paragraph pair is classified into one of the labels in each domain. We use the accuracy as the evaluation metric.

We present the experimental results from classifying the paragraph pairs 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 15, 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 paragraph pair is classified into 'boundary' or 'continuance'.

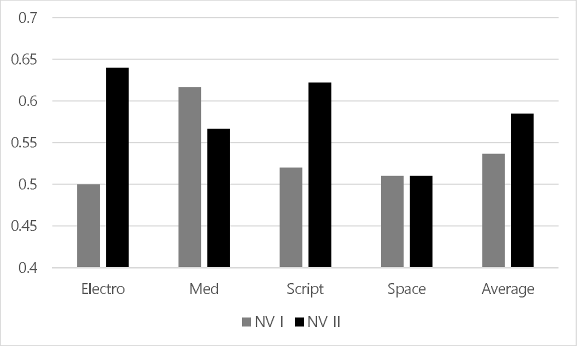

Figure 15. Results from Segmenting Texts in Text Collection: 20NewsGroup II

Let us discuss the results from classifying the adjacent paragraph pairs using the both versions of KNN algorithm on the specific version of 20NewsGroups, as shown in Figure 15. The accuracies as the performance metrics of this classification task which is mapped from the text segmentation range between 0.45 and 0.64. The proposed version shows its better results in two of the four domains: 'electro' and 'script'. It maintain its matching results in the domain, 'space', but is leaded in the domain, 'medicine'. From this set of experiments, it is concluded that the proposed version have its better performance by averaging over the accuracies of the four domains.

V. Conclusion

Let us discuss the results from segmenting a text using the two versions of KNN algorithm. In these sets of experiments, we compare the two versions with each other in the classification tasks which is mapped from the text segmentations. The proposed version shows its better results in all of the four collections. The classification accuracies of the traditional version range between 0.41 and 0.62, while those of the proposed version range between 0.52 and 0.90. From the four sets of experiments, we conclude that the proposed version improves the text segmentation performance, as the contribution of this research.

The proposed approach should be applied and validated in the specialized domains: engineering, medicine, science, and law, and it should be customized to the suitable version. We may consider similarities among only some essential features rather than among all features, to cut down the computation time. We develop and combine various schemes of computing the similarities among features. By adopting the proposed approach, we will develop the text segmentation system as a real version.

References

[1] T. Mitchell, "Machine Learning", MIT Press, 1997

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

[3] T. Jo, "Modified Version of SVM for Text Categorization", 52-60, International Journal of Fuzzy Logic and Intelligent Systems, Vol 8, No1, 2008.

[4] T. Jo, "NTSO (Neural Text Self Organizer): A New Neural Network for Text Clustering", 31-43, Journal of Network Technology, Vol 1, No 1, 2010.

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

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

[7] T. Jo, "Table based KNN for Categorizing Words", 696-700, The Proceedings of 18th International Conference on Advanced Communication Technology, 2016.

[8] T. Jo, "Table based AHC Algorithm for Clustering Words", 574-579, The Proceedings of 18th International Conference on Advanced Communication Technology, 2016.

[9] T. Jo, "Table based KNN for Extracting Keywords", 812-817, The Proceedings of 18th International Conference on Advanced Communication Technology, 2016.

[10] T. Jo, "Encoding Words into String Vectors for Word Categorization", 271-276, The Proceedings of 18th International Conference on Artificial Intelligence, 2016.

[11] T. Jo, "String Vector based AHC as Approach to Word Clustering", 133-138, The Proceedings of 12th International Conference on Data Mining, 2016.

[12] T. Jo, "Using String Vector based KNN for Keyword Extraction", 27-32, The Proceedings of 15th International Conference on Advances in Information and Knowledge Engineering, 2016.

[13] T. Jo, "Graph based KNN for Content based Word Classification", 24-29, The Proceedings of 12th International Conference on Multimedia Information Technology and Applications, 2016.

[14] 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.

[15] T. Jo, "Extracting Keywords by Graph based KNN", 96-101, The Proceedings of 12th International Conference on Multimedia Information Technology and Applications, 2016.

[16] T. Jo, “Table based AHC Algorithm for Clustering Words”, 574-579, The Proceedings of 18th International Conference on Advanced Communication Technology, 2016.

[17] T. Jo, “String Vector based AHC as Approach to Word Clustering”, 133-138, The Proceedings of 12th International Conference on Data Mining, 2016.

[18] 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.

[19] T. Jo, “K Nearest Neighbor for Text Summarization using Feature Similarity”, DOI: 10.1109/ICCCCEE.2017.7866705, Proceedings of International Conference on Communication, Control, Computing and Electronics Engineering, 2017.

[20] T. Jo, “K Nearest Neighbors for Text Segmentation with Feature Similarity”, DOI: 10.1109/ICCCCEE.2017.7866706, Proceedings of International Conference on Communication, Control, Computing and Electronics Engineering, 2017.

[21] T. Jo, “Text Categorization using K Nearest Neighbor with Feature Similarity”, 76-80, The Proceedings of International Conference on Green and Human Information Technology, 2018.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] T. Jo, “Clustering Words from News Articles by Graph based AHC Algorithm”, 66-67, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.

[27] T. Jo, “K Nearest Neighbor for Text Categorization using Feature Similarity”, 99-104, The Proceedings of 2nd International Conference on Advanced Engineering and ICT Convergence, 2019.

[28] T. Jo, “Text Classification using Feature Similarity based K Nearest Neighbor”, 13-21, AS Medical Science, Vol 3, No 4, 2019.

[29] T. Jo, “Content based Segmentation of News Articles using Feature Similarity based K Nearest Neighbor”, 61-64 The Proceedings of 19st International Conference on Information and Knowledge Engineering, 2019.

[30] T. Jo, “Text Summarization using Feature Similarity based K Nearest Neighbor”, unpublished, 2020.

[31] T. Jo, “Applying Table based AHC Algorithm to Semantic Word Clustering”, in Progress, 2020.

[32] T. Jo, “String Vector based AHC Algorithm for Clustering Words Semantically, in Progress, 2020.

[33] T. Jo, “Clustering Words Semantically by Graph based Version of AHC Algorithm”, in Progress, 2020.