Word Categorization with KNN Variants considering Feature Similarity and Feature Value Similarity

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

***Abstract*—In this research, we propose the three KNN variants which considers the feature similarity, as the approaches to the word categorization. The initial version of the KNN algorithm which does so was previously proposed as the tool of the task. We mention the three KNN variants: one which discriminates its selected nearest neighbors by their distances, another which does attributes by their correlations with the target outputs, and the other which does the training examples by their credits. The feature similarity is applied to the three KNN variants as well as the initial version. The classification performance is improved by applying the feature similarity to the KNN variants as the improved KNN versions.**

I. Introduction

The word categorization is defined as the classification of words into one or some of the predefined categories by their meanings. The process of classifying words by their spellings is called lexical classification or syntactical classification, but this research does not cover the kind of word classification. The classification of words by their meanings is called semantic word classification, and the categories into which they are classified are called topics. The task of this research is the hard word classification where each word is classified into only one category, and the task will be expanded into the soft word classification or the hierarchical word classification in subsequent researches. In a text, words are arranged grammatically for building a sentence, and sentences are assembled into a paragraph, semantically.

This research is motivated by the successful results from applying the KNN algorithm which considers the feature similarities to the word categorization. The sparse representations of texts and the semantical dependencies among the features which are given as texts are the results from encoding them into numerical vectors. The similarity between two numerical vectors is computed by considering both the similarities among features and ones among feature values. In classifying words semantically, the KNN algorithm which considers the feature similarities was empirically validated as its better performance than the traditional version. We derive the variants from the KNN algorithm and apply them to the word categorization.

The idea of this research is to modify the KNN variant into the feature similarity-based versions and apply them to the word categorization. In the previous work, the initial version of the KNN algorithm was modified into the version which considers the feature similarity and applied to the word categorization. In this research, we mention the three KNN variants: the version where nearest neighbors are discriminated by their distances, the version where attributes are discriminated in computing similarity between two numerical vectors by their correlations with target output values, and the version where training examples are discriminated by their quality. The three variants are modified into the versions which consider the feature similarities and applied to the topic-based word classification. The better performances are expected from the KNN variants in this task.

Let us mention some benefits from this research. The poor discrimination among numerical vectors is solved by utilizing the dependency among features. The classification performance is upgraded by proposing the KNN variants which nearest neighbors, training examples, and attributes are discriminated. Additionally, it is upgraded by applying the similarity metric between two numerical vectors considering the feature similarities. Mo recommendable approaches are provided for implementing the word categorization system by improving their performances.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. Previous Works

This section is concerned with the previous works which are relevant to this research. It is intended to expand the style of modifying the KNN algorithm to its three variants. The directions of exploring previous works are derivation of KNN variants, modification of the standard KNN algorithm, and the cases of using the feature similarities for improving machine learning algorithms. The distinguishable points of

this research are derivation of three KNN variants from the standard version, modification of them with the feature similarities, and application of them to the word categorization. This article is intended to explore previous works for providing the background for this research.

Let us explore the previous works on KNN variants. In 2001, the KNN variant where nearest neighbors are discriminated by their distances from or similarities as a novice item was applied to the text classification by Han et al. [1]. In 2008, MKNN (Modified K Nearest Neighbor) the KNN variants where training examples are discriminated by their validness, was proposed by Parvin et al. [3]. In 2018, the KNN variant where the attributes are discriminated by their correlations were proposed by Nababan and Sitompul [8]. However, this research uses different specific metrics for discriminating attributes, nearest neighbors, and training examples.

Let us explore the previous works on applying the modified version of KNN algorithm to the word categorization. In 2015, it was initially proposed that the modified version of KNN algorithm should be applied to the word categorization as a position paper by Jo [4]. In 2018, the modified KNN algorithm was validated in the task by Jo [5]. In 2018, the journal article which describes in detail the modified KNN algorithm and its application to the word classification was finally published by Jo [6]. This research is based on the three previous works.

Let us mention the previous works on the cases of using the feature similarities as the basis for modifying the KNN variants in this research. In 2002, feature similarities were used for selecting some features for using unsupervised learning algorithms by Mitra et al. [2]. In 2018, feature similarities were used for modifying the AHC algorithm as an approach to text clustering by Jo [7]. In 2018, the KNN algorithm which was modified based on the feature similarities was used for classifying texts [9]. Feature similarities are used for solving the sparseness in numerical vectors.

Let us mention the differences of this research from the previous works which are explored above. The KNN variants which are mentioned in the previous works do not consider the feature similarities, and in this research, we modify them into their versions which consider the feature similarities. As the expansion of [6], we modify and adopt the KNN variants for implementing the word categorization system. The feature similarities are used for modifying the KNN variants as well as the KNN algorithm and the AHC algorithm as the approach to the word classification. The modified KNN variants will be described in detail in Section III.

III. Proposed Approach

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding a word into a numerical vector and the proposed similarity metric. In Section III-B, we describe the standard version of KNN algorithm where the proposed similarity metric is adopted. In Section III-C, we derive the three varaints from the standard version. In Section III-D, we present the system architecture of the word classification system.

A. Similarity Metric

This section is concerned with the process of encoding a word into a numerical vector. Texts are used as features for encoding a word so, and some are selected from a text collection. A similarity between texts is computed based on their shared words, and it is used as a similarity between features. The similarities among features are considered for computing the semantic similarity between words. In this section, we describe the process of encoding a word into a numerical vector and the process of computing the similarity between words.

The process of encoding words into numerical vectors is illustrated in Figure 1. Texts are gathered as a collection from external sources as feature candidates. Only some texts are selected by a criterion among the gathered texts. In each word, its weights in the texts are computed as elements of the numerical vector. Refer to [9] for studying the process and the selection criteria.

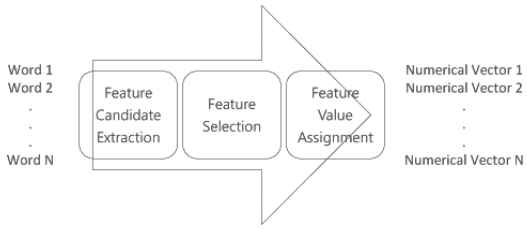

Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1\ x_2\ \ldots\ x_d]$, $\mathbf{y} = [y_1\ y_2\ \ldots\ y_d]$, and $f_1, f_2, \ldots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $sim(f_1, f_2)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as consume similarity. The similarity between words is computed by considering the two kinds of similarities.

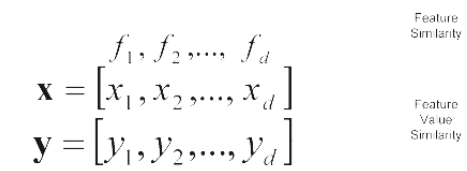

Figure 2. Frame of Computing Similarity between Numerical Vectors

Let us mention the process of computing the similarity between numerical vectors as the semantic similarity between words. The similarity between features, f_i and f_j, $sim(f_i, f_j)$, is abbreviated into s_{ij}. The similarity between features is computed by equation (1),

$$s_{ij} = \frac{2 \times |D_i \cap D_j|}{|D_i| + |D_j|} \tag{1}$$

where D_i is the set of words in the feature, f_i, and D_j is the set of words in the feature, f_j, and the similarity matrix with the d features is constructed as a $d \times d$ square matrix, as shown in equation (2),

$$\mathbf{S} = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}. \tag{2}$$

The similarity between two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (3),

$$sim(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{d} \sum_{j=1}^{d} s_{ij} x_i y_j}{d \cdot \|\mathbf{x}\| \cdot \|\mathbf{y}\|}. \tag{3}$$

Equation (3) is used for computing the similarity between a novice word and a sample word in the proposed version of the KNN algorithm.

Let us make some remarks on the encoding process the process of encoding words into numerical vectors and computing the similarity between them. A word is encoded into a numerical vector by the feature candidate extraction, the feature selection, and the feature value assignment. The similarity between features which are given as texts is computed by equation (1). The similarity between numerical vectors which represent words is computed, considering the feature similarities by equation (3). In future, we will consider computing the similarities among some features rather than all features, for improving the computation speed.

B. Feature Similarity based KNN Algorithm

This section is concerned with the initial version of KNN algorithm which considers the feature similarities. The version was initially proposed as the approach to the text classification by Jo in 2019 [9]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between a novice vector and a training example. The labels of its nearest neighbors are voted with their identical weights for decoding the label of a novice vector. This section is intended to describe the initial version of the KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into numerical vectors, and they are notated by a set $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. A novice word is encoded into a numerical vector notated by $\mathbf{x}$. Its similarities with the numerical vectors which represent the sample words are computed by equation (3), as $sim(\mathbf{x}, \mathbf{x}_1), sim(\mathbf{x}, \mathbf{x}_2), \dots, sim(\mathbf{x}, \mathbf{x}_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

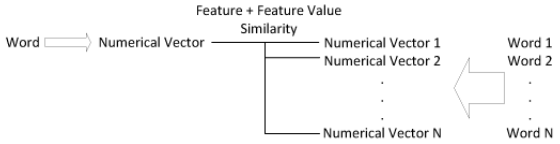

Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (4),

$$Nr = Select_k(Tr, \mathbf{x}), Nr \subseteq Tr \tag{4}$$

The set of nearest neighbors, Nr, is composed with elements as expressed in equation (5),

$$Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}\} \tag{5}$$

The elements in the set, Nr, are used for voting their labels in the next step.

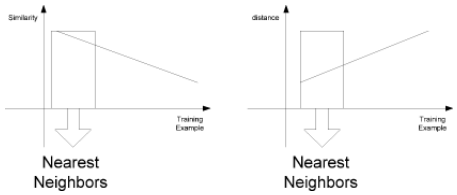

Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = label(\mathbf{x}_i^{near})$. The number of nearest neighbors which belong to the category, c_j, is notated by $Count(Nr, c_j)$. The label of the novice item, $\mathbf{x}$, is decided by equation (6),

$$label(\mathbf{x}) = \underset{j=1}{\overset{m}{\operatorname{argmax}}} Count(Nr, c_j) \tag{6}$$

The process of deciding the label of a novice item by equation (6), is called voting.

Let us make some remarks on the initial version of KNN algorithm from which we derive some variants. It computes the similarities of a novice item with the training examples.

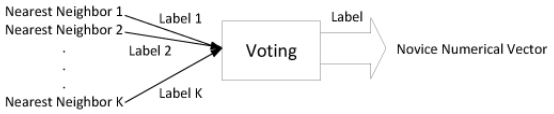

Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. KNN Variants

This section is concerned with the variants which are derived from the KNN algorithm which was covered in Section III-B. The difference from the traditional version is to adopt similarity metric which is expressed in equation (3) for computing the similarity between a novice item and a training example. In this section, we derive the three variants by discriminating nearest neighbors, attributes, or training examples. The three variants of KNN algorithm are adopted for implementing the word classification system. This section is intended to describe the three variants.

Let us mention the KNN variant which is derived by discriminating the nearest neighbors. A set of nearest neighbors is notated by $Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \ldots, \mathbf{x}_k^{near}\}$, and its elements are assumed as $sim(\mathbf{x}, \mathbf{x}_1^{near}) \geq sim(\mathbf{x}, \mathbf{x}_2^{near}) \geq \ldots \geq sim(\mathbf{x}, \mathbf{x}_k^{near})$. The weights, $w_1, w_2, \ldots, w_k$, are assigned to the nearest neighbors, $\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \ldots, \mathbf{x}_k^{near}$, following, $w_1 \geq w_2 \geq \ldots \geq w_k$, and the total weights are computed for the category, c_j by equation (7),

$$CS(Nr, c_j) = \sum_{\mathbf{x}_i \in Nr} w_i \tag{7}$$

The label of the novice item, $\mathbf{x}$, is decided by equation (8),

$$label(\mathbf{x}) = \underset{j=1}{\overset{m}{\operatorname{argmax}}} CS(Nr, c_j) \tag{8}$$

There are two ways of assigning weights to the nearest neighbors: exponentially decreasing way and arithmetic decreasing way as shown in Figure 6.

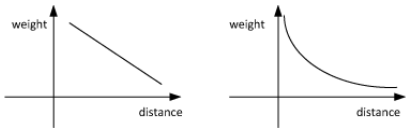

Figure 6. Discrimination over Nearest Neighbors

Let us mention the second KNN variant where the attributes are discriminated for computing the similarity between a novice item and a training example as shown in Figure 7. If the number of training examples is N, and the dimension of numerical vector representing a training example is , the attributes are notated as a set, $A = \{A_1, A_2, \ldots, A_d\}$, and each attribute is viewed as a set of values, $A_i = \{x_{1i}, x_{2i}, \ldots, x_{Ni}\}$. Each class is codified into its own integer, the correlation coefficient between an attribute A_i, and the classes is computed by equation (9),

$$r_i = \frac{\sum_{j=1}^{N}(x_{ji} - \bar{x}_j)(c_j - \bar{c})}{\sqrt{\sum_{j=1}^{N}(x_{ji} - \bar{x}_j)^2}\sqrt{\sum_{j=1}^{N}(c_j - \bar{c})^2}} \tag{9}$$

and the weight, which is assigned to the attribute, A_i, is set by the absolute value of the correlation coefficient, as expressed in equation (10),

$$w_i = |r_i| \tag{10}$$

The similarity between the novice input vector, $\mathbf{x}$ and $\mathbf{x}_i$, is computed by equation (11),

$$sim(\mathbf{x}, \mathbf{x}_i) = \frac{\sum_{j=1}^{d}\sum_{k=1}^{d} s_{jk} x_j w_k x_{ik}}{d \cdot \|\mathbf{x}\| \cdot \|\mathbf{x}_i\|}. \tag{11}$$

In this variant, equation (3) is replaced by equation (11) for computing the similarity.

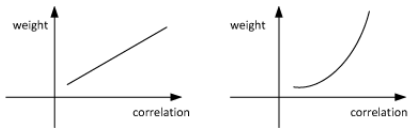

Figure 7. Discrimination over Attributes

Let us mention the third KNN variant where the training examples are discriminated for computing the similarity between a novice item and a training example and/or voting the nearest neighbors for deciding the label as shown in Figure 8. The hypersphere radius is used as the parameter, and for each training example, other training examples are taken within the radius from it as its nearest neighbors. The number of nearest neighbors and the number of training examples which are labeled identically to the training example, $\mathbf{x}_i$, are notated respectively by r_i and $r_i^{=}$, and the weight is computed by equation (12),

$$w_i = \frac{r_i^{=}}{r_i} \tag{12}$$

The similarity between the novice item, $\mathbf{x}$, and the training example, $\mathbf{x}_i$, is computed by equation (13),

$$sim(\mathbf{x}, \mathbf{x}_i) = \frac{w_i \sum_{j=1}^{d}\sum_{k=1}^{d} s_{jk} x_j x_{ik}}{d \cdot \|\mathbf{x}\| \cdot \|\mathbf{x}_i\|}. \tag{13}$$

and the total weight is computed for the category, c_j, by equation (14), in voting,

$$CS(Nr, c_j) = \sum_{\mathbf{x}_i \in Nr} w_i \tag{14}$$

Equation (13) and (14) are used respectively for computing the similarity between a novice item and a training example and voting the nearest neighbors for each category.

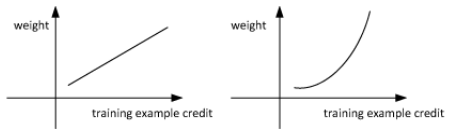

Figure 8. Discrimination over Training Examples

Let us make some remarks on the three variants of KNN algorithm which are proposed as the approaches to the word classification. In the first variant, the nearest neighbors are discriminated for voting their labels. In the second variant, the attributes are discriminated for computing the similarity between a novice input and a training example. In the third variant, the training examples are discriminated for voting the labels of the nearest neighbors and/or computing the similarity between numerical vectors. We may consider the trainable KNN algorithm which optimizes the weights of the nearest neighbors, the attributes, and the training examples for minimizing the training error.

D. System Architecture

This section is concerned with the architecture and the execution flow of the word classification system. In Section III-C, we described the three KNN variants which are adopted for implementing the word classification system. The initial version of KNN algorithm which is mentioned in Section III-B is replaced by its variants in the architecture of the word classification system which was previously proposed. The process of executing the system is to encode a word into a numerical vector and classify it into one among the predefined categories. This section is intended to describe the word classification system which is implemented in this study.

The process of collecting sample words and encoding them into numerical vectors is illustrated in Figure 9. The topics are predefined as topic 1, topic 2, ..., topic M. The K words are allocated for each topic as sample words. We need the balanced distribution over the categories for preventing the bias toward a particular topic. The $M \times K$ sample words are encoded into numerical vectors by the process which is mentioned in Section III-B.

The entire architecture of the proposed word classification system is illustrated in Figure 10. The sample words and novice words are encoded into numerical vectors. For each novice vector, its similarities with the sample words are computed by the similarity metric, which is mentioned in Section III-A, in the similarity computation model, and the k most similar training examples are selected as the nearest neighbors. The label of a novice item is decided by voting ones of the nearest neighbors in the voting module. The difference from the system architecture which is proposed

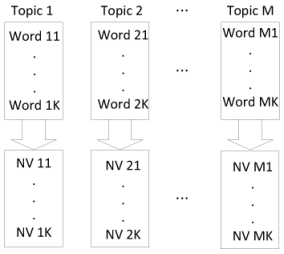

Figure 9. Preparation of Training Examples

in [6] is to replace the initial version of KNN algorithm by its variants.

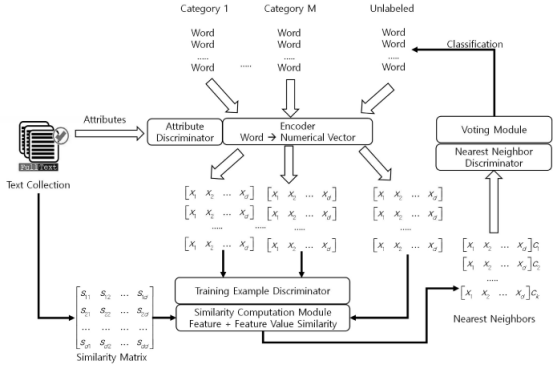

Figure 10. System Architecture

The execution process of the proposed system is illustrated in Figure 11. The sample words and a novice word are encoded into numerical vectors. The difference from the previous version is to compute weights of the attributes, the training examples, and the nearest neighbors. The category of a novice word is decided by voting ones of its nearest neighbors with their discriminations. It is the final output in this system.

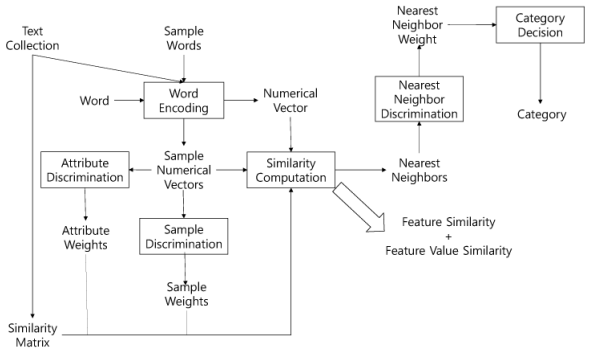

Figure 11. System Flow

Let us make some remarks on the proposed system which is illustrated in Figure 10 as its architecture. The M topics are predefined, words each of which is labeled with

one among them are collected, and they are encoded into numerical vectors. Novice words are classified into one of the three variants which are described in Section III-C. The difference from the previous version is to select the nearest neighbor discrimination, the attribute discrimination, and the training example discrimination. We expect the better performance by replacing the initial version by its variants.

IV. Conclusion

Let us mention the significances of this research as the conclusion. We apply the proposed similarity metric to the KNN variants as well as the standard version. We derive the three variants from the standard version by discriminating the nearest neighbors, the attributes, and the training examples. We design the word classification system by adopt the KNN variants with the proposed similarity metric. In the next research, we will implement the word classification system as a real system.

Let us mention the remaining tasks as the further discussions on this research. We need to improve the high complexity in computing proposed the similarity metric by allowing only some features to compute their similarities. Other machine learning algorithms such as Naïve Bayes and SVM (Support Vector Machine) are also modified by applying the proposed similarity metric. The proposed versions of KNN variants are applied to other tasks such as text classification and text summarization. The word categorization should be implemented by adopting the proposed KNN variants as a real problem.

References

[1] E.H. Han, G. Karypis and V. Kumar, "Text Categorization Using Weight Adjusted k-Nearest Neighbour Classification" pp53-64, in Advances in Knowledge Discovery and Data Mining. PAKDD 2001, Berlin, Heidelberg:Springer, 2035, 2001.

[2] P. Mitra, C. A. Murthy, and S. K. Pal. "Unsupervised feature selection using feature similarity", 301-312, IEEE transactions on pattern analysis and machine intelligence 24.3 2002.

[3] H. Parvin, A. Hosein, and M.B. Behrouz, "MKNN: Modified k-nearest neighbor" Proceedings of the World Congress on Engineering and Computer Science. Vol. 1. Newswood Limited, 2008.

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

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

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

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

[8] A.A. Nababan and O. S. Sitompul, "Attribute weighting based K-nearest neighbor using Gain Ratio", Journal of Physics: Conference Series, 1007, 1, 2018.

[9] T. Jo, "Text Classification using Feature Similarity based K Nearest Neighbor", 13-21, AS Medical Science, 3, 4, 2019.