Table based KNN Variants for Categorizing Words

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

***Abstract*—In this research, we propose the table based KNN variants, as the approach to the word categorization. The initial KNN version which receives a table as its input data was previously proposed as the tool of such task. In this research, we mention the three KNN variants: one where the selected nearest neighbors are discriminated by their similarities with a novice example, one where the attributes are discriminated by their correlations with the target outputs, and one where the training examples are discriminated by their credits. In this research, we modify the three KNN variants as well as the initial version of the KNN algorithm. As the goal of this research, we try to improve the classification performance by modifying the KNN variants so.**

I. Introduction

The word categorization is defined as the process of classifying words into one or some among the predefined categories. The classification of words by their spellings is called lexical classification or syntactical classification, but such kind of word classification is excluded in this research. The kind of word classification which is covered in this research and is mentioned in its definition is called semantic word classification. The categories into which words are classified are defined as topics. The hard word classification where each word is classified into only one category is the task of this research, and it is expanded into the soft word classification or the hierarchical word classification, in subsequent researches.

This research is motivated by the successful results from applying the table based KNN algorithm to the word categorization in previous works. The sparse distribution and the big dimensionality are the main problems in encoding words or texts into numerical vectors. Words are encoded into tables, instead of numerical vectors, and the KNN algorithm is modified into the version which process tables directly. The table based KNN algorithm was empirically validated into the better approach to the word categorization than the traditional KNN algorithm, in previous works. We derive the KNN variants and modify them into the table-based versions as the approaches to the word categorization.

The idea of this research is to modify the KNN variants into the table-based versions as the approaches to the word classification. In the previous work, the initial version of KNN algorithm was modified into the version which processes tables directly as the approach to the word classification. We mention the table based KNN variants: one where nearest neighbors are discriminated by their distances, one where entries are discriminated by their correlations with the target outputs, and one where the training examples are discriminated by their qualities. They are proposed as the approaches to the word categorization in this research. Their performances will be improved in the word categorization, as well as the table-based version of the KNN algorithm.

Let us consider some benefits from this research. The main problems such as the sparseness and the huge dimensionality in encoding words into numerical vectors are solved by encoding them into tables. The classification performance is upgraded by proposing the KNN variants where nearest neighbors, training examples, and table entries are discriminated. Additionally, it is upgraded by modifying the KNN variants into their table-based versions. More reliable approaches are provided for implementing the word categorization system by this research.

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 into its table-based version, and the previous case of encoding a text into a table. The distinguishable points of this research are derivation of three KNN variants from the standard version, modification of them into their table-based versions, 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 2016, it was initially proposed that the table-based version of KNN algorithm should be applied to the word categorization as a position paper by Jo [6]. In 2018, the modified KNN algorithm was validated in the task by Jo [7]. The journal article which describes in detail the modified KNN algorithm and its application to the word classification is finally prepared for its publication by Jo [9]. This research is based on the three previous works.

Let us explore the previous works on the table matching algorithm as the early case of encoding a text into a table. In 2008, the table based matching was proposed as an approach to the text classification as the initial case of encoding a text into a table, instead of a numerical vector [2]. In 2011, the table based matching algorithm was registered as a patent by Jo [4]. In 2015, it was upgraded into its more reliable and stable version by Jo [5]. The similarity metric between two tables is provided by the above previous works for modifying the standard KNN algorithm and its variants.

Let us mention the differences of this research from the previous works which were explored above. The KNN variants in the previous works are numerical vector-based versions, and they will be modified into table-based versions in this research. As the expansion of [9], we modify and adopt the KNN variants for implementing the word categorization system. The modified versions of KNN variants become more advanced ones, compared with the table based matching algorithm. 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 table 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 table as the representation of a word. The table is defined as a set of entries, each of which consists of an element and its weight. The table which represents a word consists of entries each of which contains a text identifier and its weight. The similarity between tables is computed based on shared words as the semantic similarity between words. This section is intended to describe the process of encoding a word into a table, and the process of computing the similarity between tables.

The process of encoding words into tables is illustrated in Figure 1. In the inverted indexing, each word is linked with texts which include itself. In the text-word weighting, its weights in the lined texts are computed; a weight indicates a relationship between a word and a text. Entries with their lower weights are removed for downsizing the table. Refer to [7] for studying the encoding process in more detail.

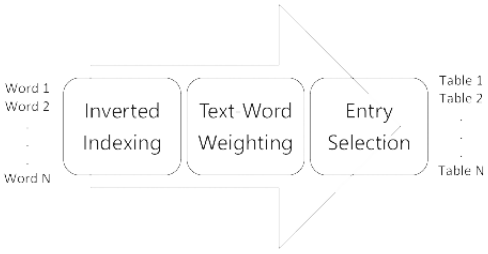

Figure 1. Process of Encoding Words into Tables

Let us mention the function of a table which maps it into a set of text identifiers as shown in Figure 2. The table which represents a word is expressed as entry set as shown in equation (1),

$$T = \{(d_1, w_1), (d_2, w_2), \ldots, (d_{|T|}, w_{|T|})\} \tag{1}$$

where d_i is a text identifier and w_i is the weight of the word, T, in the text, d_i. The function for generating a list of text identifiers is expressed as equation (2),

$$F(T) = \{d_1, d_2, \ldots, d_{|T|}\} \tag{2}$$

The function is the operation which takes only text identifiers without their weights as a set. The operation is applied to computing the similarity between tables which represent words.

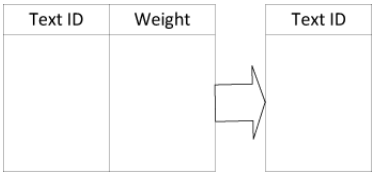

Figure 2. Conversion of Table into Text Set

Let us mention the process of computing the similarity between two tables as a semantic similarity between words. The two tables which represent words are notated respectively by $T_1 = \{(d_{11}, w_{11}), (d_{12}, w_{12}), \ldots, (d_{1|T|}, w_{1|T|})\}$ and $T_2 = \{(d_{21}, w_{21}), (d_{22}, w_{22}), \ldots, (d_{2|T|}, w_{2|T|})\}$. The

intersection between the two sets which is generated by applying the function, $F(\cdot)$ as shown in equation (3),

$$F(T_1) \cap F(T_2) = \{d_{s1}, d_{s2}, \ldots, d_{s|F(T_1)\cap F(T_2)|}\} \quad (3)$$

and the table with entries each of which consists of a shared text identifier, d_{si}, its weight from the table, T_1, w^1_{si}, and its weight from the table, T_2, w^2_{si}, is expressed as equation (4),

$$ST_{12} = \{(d_{s1}, w^1_{s1}, w^2_{s1}), (d_{s2}, w^1_{s2}, w^2_{s2}), \ldots, (d_{s|F(T_1)\cap F(T_2)|}, w^1_{s|F(T_1)\cap F(T_2)|}, w^2_{s|F(T_1)\cap F(T_2)|})\} \quad (4)$$

The similarity between tables, T_1 and T_2 is computed by equation (5),

$$sim(T_1, T_2) = \frac{2(\sum_{i=1}^{|F(T_1)\cap F(T_2)|} w^1_{si} + \sum_{i=1}^{|F(T_1)\cap F(T_2)|} w^2_{si})}{\sum_{i=1}^{|T_1|} w_{1i} + \sum_{i=1}^{|T_1|} w_{2i}}. \quad (5)$$

The value which is computed from equation (5) is always given as a normalized value between zero and one as shown in equation (6),

$$0 \leq sim(T_1, T_2) \leq 1. \quad (6)$$

Let us make some remarks on the encoding process and the computation of similarity between words. A word is encoded into table by the inverted indexing, the text identifier weighting, and the entry selection. A table is expressed as a set of entries and filtered into a set of text identifiers by the function. The similarity between tables is computed based on their shared entries by equation (5). In the future research, we consider representing a word into multiple tables.

B. Initial Version of Table based KNN Algorithm

This section is concerned with the initial version of table based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 [7]. The similarity metric between tables which was described in the previous section is used for computing the similarity between a novice table and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of 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 tables, and they are notated by a set $Tr = \{T_1, T_2, \ldots, T_N\}$. A novice word is encoded into a table notated by T. Its similarities with the tables which represent the sample words are computed by equation (5), as $sim(T, T_1), sim(T, T_2), \ldots, sim(T, T_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

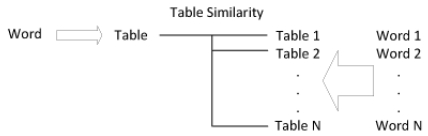

Figure 3. Similarities of Novice Input with Training Examples

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 (7),

$$Nr = Select_k(Tr, T), Nr \subseteq Tr \quad (7)$$

The set of nearest neighbors, Nr, is composed with elements as expressed in equation (8),

$$Nr = \{T^{near}_1, T^{near}_2, \ldots, T^{near}_k\} \quad (8)$$

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

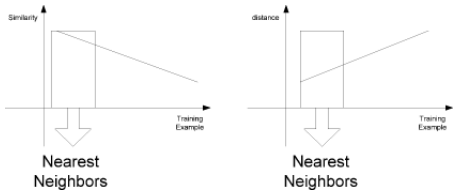

Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

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 $c_1, c_2, \ldots, c_m$, and the label of a nearest neighbor is notated by $c_j = label(T^{near}_i)$. 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, T, is decided by equation (9),

$$label(T) = \underset{j=1}{\overset{M}{\operatorname{argmax}}} Count(Nr, c_j) \quad (9)$$

The process of deciding the label of a novice item by equation (9), 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. 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

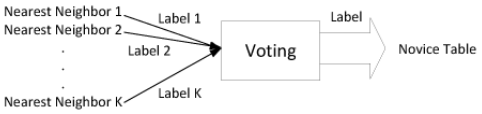

Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

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 between tables for computing the similarity between a novice item and a training example. In this section, we derive the three variants by discriminating nearest neighbors, entries, 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 $Nr = \{T_1^{near}, T_2^{near}, \ldots, T_k^{near}\}$, and its elements are assumed as $sim(T, T_1^{near}) \geq sim(T, T_2^{near}) \geq \ldots \geq sim(T, T_k^{near})$. The weights, $w_1, w_2, \ldots, w_k$, are assigned to the nearest neighbors, $T_1^{near}, T_2^{near}, \ldots, T_k^{near}$, following, $w_1 \geq w_2 \geq \ldots \geq w_k$, and the total weights are computed for the category, c_j by equation (10),

$$CS(Nr, c_j) = \sum_{T_i \in Nr} w_i \tag{10}$$

The label of the novice item, T, is decided by equation (11),

$$label(T) = \underset{j=1}{\overset{m}{\operatorname{argmax}}} CS(Nr, c_j) \tag{11}$$

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

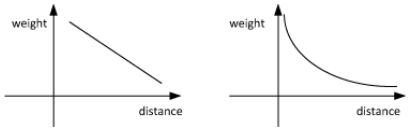

Figure 6. Discrimination over Nearest Neighbors

Let us mention the second table based KNN variant where the entries are discriminated for computing the similarity between a novice item and a training example as shown in Figure 7. The occurrences of the text, d_i, in the words which are labeled with the category, c_j, is $f(d_i|c_j)$, and the total occurrences of the text, d_i in the sample words is $f(d_i)$. The influence of the text, d_i, on the category, c_j, is computed by equation (12),

$$I(d_i, c_j) = \frac{f(d_i|c_j)}{f(d_i)} \tag{12}$$

and the weight, w_i, is computed by equation (13),

$$w_i = \max_{j=1}^{m} I(d_i, c_j). \tag{13}$$

Equation (5) for computing the similarity between tables is modified into equation (14),

$$sim(T_1, T_2) = \frac{2(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_i w_{si}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_i w_{si}^2)}{\sum_{i=1}^{|T_1|} w_i w_{1i} + \sum_{i=1}^{|T_1|} w_i w_{2i}}. \tag{14}$$

In this variant, equation (5) is replaced by equation (14) for computing the similarity between a novice item and a training example.

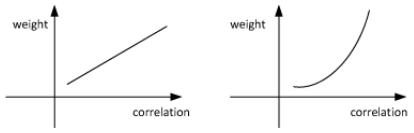

Figure 7. Discrimination over Table Entries

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 similarity threshold is used as the parameter, and for each training example, other training examples are taken within the threshold 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, T_i, are notated respectively by r_i and $r_i^=$, and the weight is computed by equation (15),

$$w_i = \frac{r_i^=}{r_i} \tag{15}$$

The similarity between the novice item, T, and the training example, T_i, is computed by equation (16),

$$sim(T_1, T_2) = w_i \left(\frac{2(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^2)}{\sum_{i=1}^{|T_1|} w_{1i} + \sum_{i=1}^{|T_1|} w_{2i}} \right), \tag{16}$$

and the total weight is computed for the category, c_j, by equation (17), in voting,

$$CS(Nr, c_j) = \sum_{T_i \in Nr} w_i. \tag{17}$$

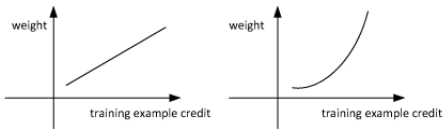

Figure 8. Discrimination over Training Examples

Equation (16) and (17) are used respectively for computing the similarity between a novice item and a training example and voting the nearest neighbors for each category.

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 entries 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 tables. 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 table 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 tables 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 tables 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 tables. 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

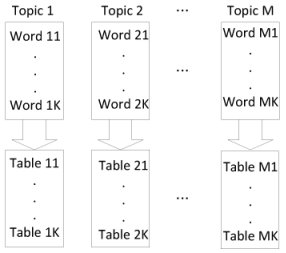

Figure 9. Preparation of Training Examples

the system architecture which is proposed in [7] is to replace the initial version of KNN algorithm by its variants.

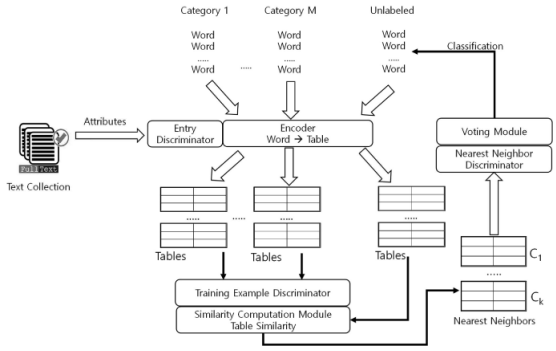

Figure 10. System Architecture

The execution process of the proposed system is illustrated in Figure 11. The sample words and a novice word are encoded into tables. The difference from the previous version is to compute weights of the entries, 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.

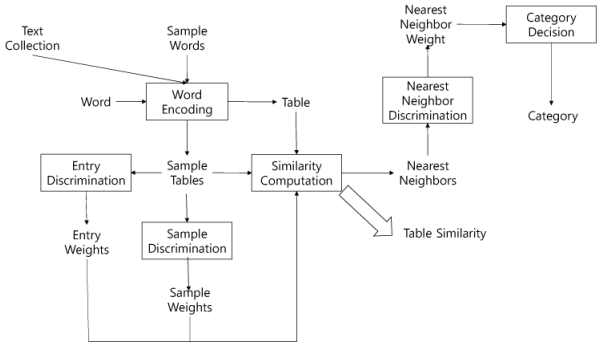

Figure 11. System Flow

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 tables. 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 entry 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 encode words into tables and apply the similarity metric between tables 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 entries, 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 speed of computing the similarity between tables in the implementation level. Other machine learning algorithms such as the Naïve Bayes and the SVM (Support Vector Machine) are modified into the table-based versions. The proposed versions of the KNN variants are applied to other tasks such as text classification and text summarization. The word categorization system should be implemented by adopting the proposed approaches as a real program.

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] T. Jo, "Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578", 875-882, Journal of Korea Multimedia Society, Vol 11, No 6, 2008.

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

[5] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", 839-849, Soft Computing, 19, 4, 2015.

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

[7] T. Jo, "Table based K Nearest Neighbor for Word Categorization in News Articles", The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 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, "Using Table based Version of K Nearest Neighbor for Classifying Words Semantically", in preparation, 2023.