AHC based Word Clustering considering Feature Similarity

Taeho Jo
Department of Computer and Information Engineering, Inha University, Incheon, South Korea

Abstract—*In this research, we propose that the AHC (Agglomerative Hierarchical Clustering) algorithm should be used for clustering words, considering the feature similarities. Among the features, their dependencies and relations are available in the reality; texts which are features for encoding words into numerical vectors have own similarities with others. In this research, we define the similarity measure which considers both the features and the feature values, and use it for modifying the AHC algorithm as the approach to the word clustering. As the benefits from this research, we may obtain the potential possibility of more compact word representations and the more tolerance to the sparse distributions of numerical vectors. Therefore, the goal of this research is to implement the word clustering systems with the benefits.*

Keywords: Word Clustering, Feature Similarity

1. Introduction

The word clustering refers to the process of segmenting a group of words into subgroups of content based similar words. The group of words is encoded into their structured forms and a similarity measure between them is defined. The words are arranged into their closet clusters based on the similarity measure, as the clustering proceeds. The results from clustering the words are unnamed clusters and cluster naming and cluster prototype definition are regarded as other tasks in this research. The scope of this research is restricted to cluster words by their meanings.

Let us mention some challenges which this research tries to solve. The strong dependency among features exists especially in the text mining tasks, so the Bayesian networks which considers it was proposed as the approach, but it requires very much complicated analysis for using it [1]. If the independences among features are assumed, it requires many features for encoding words or texts into numerical vectors. Since each feature has very little coverage in the domain of text mining, we cannot avoid the sparse distribution of numerical vectors which represent words or texts[3]. Therefore, this research is intended to solve the problems by considering the feature similarity as well as the feature value one.

Let us mention what we propose in this research as its idea. In this research, we consider the both similarity measures, feature similarity and feature value similarity, for computing the similarity between numerical vectors. The AHC (Agglomerate Hierarchical Clustering) algorithm is modified into the version which accommodates the both similarity measures. The modified version was applied to the word clustering task. Therefore, the goal of this research is to improve the word clustering performance by solving the above problems.

Let us mention the benefits which we expect from this research. The consideration of both the feature similarity and the feature value similarity provides the way of reducing the dimensionality of numerical vectors, potentially. We discover semantic relations among words through this research for performing other text mining tasks. The improvement of discriminations among even sparse numerical vectors is caused by computing the similarity between numerical vectors using the two measures. Therefore, the goal of this research is to pursue the benefits for implementing the text clustering systems.

This article is organized into the four sections. In Section **??**, we survey the relevant previous works. In Section 3, we describe in detail what we propose in this research. In Section 4, we mention the remaining tasks for doing the further research.

2. Previous Works

Let us survey the previous cases of encoding texts into structured forms for using the machine learning algorithms to text mining tasks. The three main problems, huge dimensionality, sparse distribution, and poor transparency, have existed inherently in encoding them into numerical vectors. In previous works, various schemes of preprocessing texts have been proposed, in order to solve the problems. In this survey, we focus on the process of encoding texts into alternative structured forms to numerical vectors. In other words, this section is intended to explore previous works on solutions to the problems.

Let us mention the popularity of encoding texts into numerical vectors, and the proposal and the application of string kernels as the solution to the above problems. In 2002, Sebastiani presented the numerical vectors are the standard representations of texts in applying the machine learning algorithms to the text classifications [4]. In 2002, Lodhi et al. proposed the string kernel as a kernel function of raw texts in using the SVM (Support Vector Machine) to the text classification [5]. In 2004, Lesile et al. used the version of SVM which proposed by Lodhi et al. to the protein classification [6]. In 2004, Kate and Mooney used also the SVM version for classifying sentences by their meanings [7].

It was proposed that texts are encoded into tables instead of numerical vectors, as the solutions to the above problems. In 2008, Jo and Cho proposed the table matching algorithm as the approach to text classification [8]. In 2008, Jo applied also his proposed approach to the text clustering, as well as the text categorization [12]. In 2011, Jo described as the technique of automatic text classification in his patent document [10]. In 2015, Jo improved the table matching algorithm into its more stable version [11].

Previously, it was proposed that texts should be encoded into string vectors as other structured forms. In 2008, Jo modified the k means algorithm into the version which processes string vectors as the approach to the text clustering[12]. In 2010, Jo modified the two supervised learning algorithms, the KNN and the SVM, into the version as the improved approaches to the text classification [13]. In 2010, Jo proposed the unsupervised neural networks, called Neural Text Self Organizer, which receives the string vector as its input data [14]. In 2010, Jo applied the supervised neural networks, called Neural Text Categorizer, which gets a string vector as its input, as the approach to the text classification [15].

The above previous works proposed the string kernel as the kernel function of raw texts in the SVM, and tables and string vectors as representations of texts, in order to solve the problems. Because the string kernel takes very much computation time for computing their values, it was used for processing short strings or sentences rather than texts. In the previous works on encoding texts into tables, only table matching algorithm was proposed; there is no attempt to modify the machine algorithms into their table based version. In the previous works on encoding texts into string vectors, only frequency was considered for defining features of string vectors. Texts which are used as features of numerical vectors which represent words have their semantic similarities among them, so the similarities will be used for processing sparse numerical vectors, in this research.

3. Proposed Approach

This section is concerned with modifying the AHC (Agglomerative Hierarchical Clustering) algorithm into the version which considers the similarities among features as well as feature values, and it consists of the three sections. In Section 3.1, we describe the process of encoding words into numerical vectors. In Section 3.2, we do formally the proposed scheme of computing the similarity between two numerical vectors. In Section 3.3, we mention the proposed version of AHC algorithm which considers the similarity among features as the approach to word clustering. Therefore, this article is intended to describe in detail the modified version of KNN algorithm and its application to the word clustering.

3.1 Word Encoding

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

In the first step of word encoding, a word-document matrix is constructed automatically from a text collection called corpus. In the corpus, each text is indexed into a list of words. For each word, we compute and assign its weight which is called TF-IDF (Term Frequency-Inverse Document Frequency) weight [2], by equation (1),

$$w_i = TF_i(\log_2 N - \log_2 DF_i + 1) \tag{1}$$

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

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

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

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

3.2 Feature Similarity

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

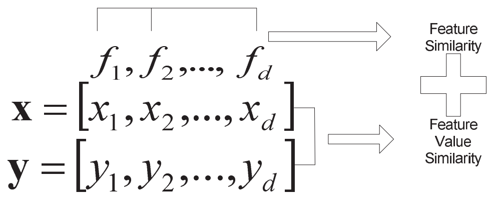
$f_1, f_2, ..., f_d$
$\mathbf{x} = [x_1, x_2, ..., x_d]$
$\mathbf{y} = [y_1, y_2, ..., y_d]$
Feature Similarity
Feature Value Similarity

Fig. 1
THE COMBINATION OF FEATURE AND FEATURE VALUE SIMILARITY

Text identifiers are given as features for encoding words into numerical vectors. Texts are dependent on others rather than independent ones which are assumed in the traditional classifiers, especially in Naive Bayes [1]. Previously, various schemes of computing the semantic similarity between texts were developed [2]. We need to assign nonzero similarity between two numerical vectors where non-zero elements are given to different features with their high similarity. It is expected to improve the discriminations among sparse vectors by considering the similarity among features.

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

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

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

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

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

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

$$t_1 = [w_{11}, w_{12}, ..., w_{1d}]$$
$$t_2 = [w_{21}, w_{22}, ..., w_{2d}].$$

The features,$d_1, d_2, ..., d_d$ are defined through the process which was described in Section 3.1. We construct the d by d matrix as the similarity matrix of features by the process mentioned above. The similarity between the two vectors are computed with the assumption of availability of the feature similarities, by equation (3),

$$sim(t_1, t_2) = \frac{\sum_{i=1}^{d}\sum_{j=1}^{d} s_{ij} w_{1i} w_{2j}}{d \cdot \|t_1\| \cdot \|t_2\|} \tag{3}$$

where $\|t_1\| = \sqrt{\sum_{i=1}^{d} w_{1i}^2}$ and $\|t_2\| = \sqrt{\sum_{i=1}^{d} w_{2i}^2}$. We get the value of s_{ij} by equation (2).

The proposed scheme of computing the similarity by equation (3) has the higher complexity as payment for obtaining the more discrimination among sparse vectors. Let us assume that two d dimensional numerical vectors are given as the input for computing the similarity between them. It takes only linear complexity, $O(d)$, to compute the cosine similarity as the traditional one. However, in the proposed scheme takes the quadratic complexity, $O(d^2)$. We may reduce the complexity by computing similarities of some pairs of features, instead of all.

3.3 Proposed Version of AHC Algorithm

This section is concerned with the modified version of AHC algorithm which considers both the feature similarity and the feature value one. The words which are given as clustering targets are encoded into numerical vectors whose features are texts by the scheme which was described in section **??**. The numerical vectors which represent words or texts tend to be sparse, inherently; zero values in each vector tend to be dominant over 90%. In the proposed version, the similarities among the numerical vectors are computed by equation (3). In order to provide the detail explanation, we describe the proposed AHC version, together with the traditional one.

The traditional version of AHC algorithm is illustrated in Figure 2. Words are encoded into numerical vectors, and it begins with unit clusters each of which has only single item. The similarity of every pairs of clusters is computed using the Euclidean distance or the cosine similarity, and the pair with its maximum similarity is merged into a cluster. The clustering by the ACH algorithm proceeds by merging cluster pairs and decrementing number of clusters by one. If the similarities among the sparse numerical vectors are computed, the traditional version becomes very fragile from the poor discriminations among them.

The proposed AHC version is illustrated in Figure 3. Words are encoded into numerical vectors, and the clustering begins with individual items. The similarities among numerical vectors are computed by equation (3) which was presented in section 3.2. Clustering proceeds by merging the pair with its maximum similarity. By replacing the Euclidean distance or the cosine similarity by equation (3), it is expected to improve the discriminations among even sparse numerical vectors.

We may consider several schemes of computing a similarity between clusters. We may compute similarities of all possible pairs of items between two clusters and average over them as the cluster similarity. The maximum or the minimum among

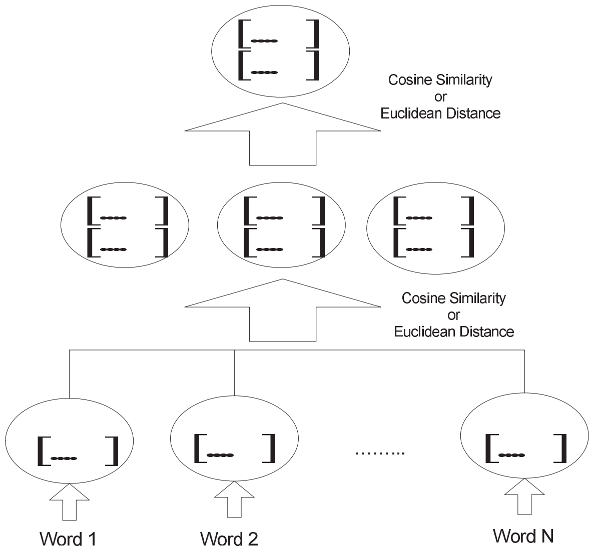

Fig. 2
THE TRADITIONAL VERSION OF AHC ALGORITHM

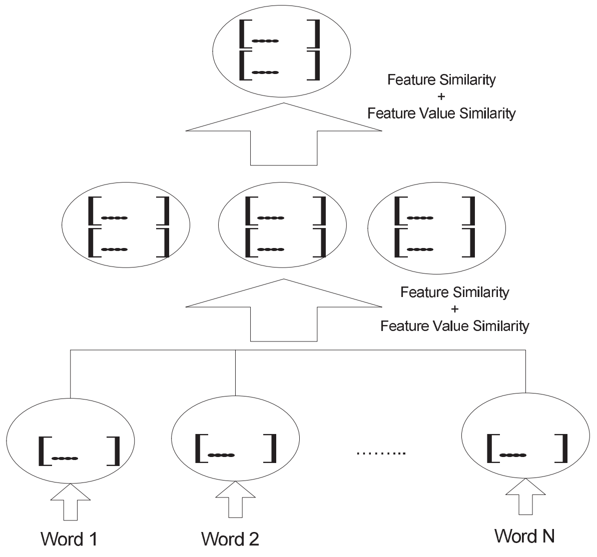

Fig. 3
THE PROPOSED VERSION OF AHC ALGORITHM

similarities of all possible pairs is set as the cluster similarity. In another scheme, we may select representative members of two clusters and the similarity between the selected members is regarded as the cluster similarity. In this research, we adopt the first scheme for computing the similarity between two clusters in using the AHC algorithm; other schemes will be considered in next research.

Let us compare the both AHC versions with each other. Both versions begin the clustering process with individual numerical vectors. In computing the similarity between two numerical vectors, the traditional version uses the Euclidean distance or cosine similarity mainly, whereas the proposed one uses the equation (3). Like the traditional version, the pair of clusters with its maximum similarity is merged into a single cluster in doing the clustering process. However, the proposed version is more tolerant to sparse numerical vectors in computing the similarities among them than the traditional version.

4. Conclusion

Let us mention the remaining tasks for doing the further research. We need to validate the proposed approach in specific domains such as medicine, engineering, and economics, as well as in generic domains such as ones of news articles. We may consider the computation of similarities among some main features rather than among all features for reducing the computation time. We try to modify other machine learning algorithms such as Naive Bayes, Perceptrons, and SVM (Support Vector Machine) based on both kinds of similarities. By adopting the proposed approach, we may implement the word clustering system as a real program.

References

[1] T. Mitchell, *Machine Learning*, McGraw-Hill, 1997.
[2] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval: The Concepts and Technology behind Search*, Addison-Wesley, 2011.
[3] T. Jo, “The Implementation of Dynamic Document Organization using Text Categorization and Text Clustering" PhD Dissertation, University of Ottawa, Ottawa, Canada, 2006.
[4] F. Sebastiani, “Machine Learning in Automated Text Categorization", *ACM Computing Survey*, Vol. 34, pp. 1-47, 2002.
[5] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text Classification with String Kernels", *Journal of Machine Learning Research*, Vol. 2, pp. 419-444, 2002.
[6] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, “Mismatch String Kernels for Discriminative Protein Classification", *Bioinformatics*, Vol. 20, pp. 467-476, 2004.
[7] R. J. Kate and R. J. Mooney, “Using String Kernels for Learning Semantic Parsers", in *Proc. ICCL ’06*, 2006, pp. 913-920.
[8] T. Jo and D. Cho, “Index based Approach for Text Categorization", *International Journal of Mathematics and Computers in Simulation*, Vol. 2, 2008, pp. 127-132.
[9] T. Jo, “Single Pass Algorithm for Text Clustering by Encoding Documents into Tables", *Journal of Korea Multimedia Society*, Vol. 11, 2008, pp. 1749-1757.
[10] T. Jo, “Device and Method for Categorizing Electronic Document Automatically", South Korean Patent 10-1071495, 2011.
[11] T. Jo, “Normalized Table Matching Algorithm as Approach to Text Categorization", *Soft Computing*, Vol. 19, 2015, pp. 849-849.
[12] T. Jo, “Inverted Index based Modified Version of K-Means Algorithm for Text Clustering", *Journal of Information Processing Systems*, Vol. 4, 2008, pp. 67-76.
[13] T. Jo, “Representationof Texts into String Vectors for Text Categorization", *Journal of Computing Science and Engineering*, Vol. 4, 2010, pp. 110-127.
[14] T. Jo, “NTSO (Neural Text Self Organizer): A New Neural Network for Text Clustering", *Journal of Network Technology*, Vol. 1, 2010, pp. 31-43.
[15] T. Jo, “NTC (Neural Text Categorizer): Neural Network for Text Categorization", *International Journal of Information Studies*, Vol 2, 2010, pp. 83-96.