Table based AHC for Text Clustering

Taeho Jo
Department of Computer and Information Communication Engineering, Hongik University, Sejong, South Korea

Abstract— *In this research, we propose the modified version of AHC (Agglomerated Hierarchical Clustering) algorithm as an approach to the text clustering. Encoding texts into numerical vectors for using the traditional versions causes the three main problems: the huge dimensionality, the sparse distribution, and the poor transparency; this fact motivated this research. As the solution to the three problems, the idea of this research is to modify the AHC algorithm which is known as the popular and practical one into the version where texts are encoded into tables, instead of the numerical vectors. The modified version which is proposed in this research is expected to cluster texts more reliably than the traditional version by solving the three problems. Hence, the goal of this research is to implement the text clustering system, using the proposed approach.*

Keywords: Text Clustering, Table Similarity , Agglomerative Hierarchical Clustering Algorithm

1. Introduction

The text clustering refers to the process of segmenting a group of content based various texts into subgroups of similar ones as an instance of pattern clustering. Even if various types of approaches are available, we assume that the unsupervised machine learning algorithms are mainly used as the approaches. Texts are encoded into structured forms and clustered based on their similarities among their structured forms rather than ones among their raw texts. The text clustering results in a list of unnamed clusters and the task of naming clusters relevantly is considered as another task. Note that the clustering is a very expensive computation whatever data items are.

Let us consider the three motivations which lead to this research. First, encoding texts into numerical vector for using a traditional approach may cause the three main problems: huge dimensionality, sparse distribution, and poor transparency [2][3][4][13][6]. Second, encoding texts into tables was very successful in another task of text mining: text categorization [3][4] [7]. Third, previously, we tried to encode texts into string vectors, but more mathematical definitions and characterizations were required for creating and modifying string vector based versions of machine learning algorithms [13][6]. Hence, the three agenda motivated us to carry out this research; we attempt to encode texts into tables for using the AHC (Agglomerative Hierarchical Clustering) algorithms. .

We present the agenda which are proposed in this research. In this research, texts are encoded into table, instead of numerical vectors, to avoid the three main problems. We define the similarity measure between tables which is always given as a normalized value and modify the AHC algorithm using the measure. The modified AHC algorithm will be used as the approach to the text clustering task. Note that each table which represents a text consists of its own entries of words and their weights.

Let us consider some benefits from this research. We avoid the three main problems in encoding texts into numerical vectors. We may expect the better performance and more stability than the traditional version of AHC algorithm. Since the table is more symbolic than the numerical vector as the representation of each text, it provides more transparency where we can guess the contents of texts only by their representations. However, since the table size is given as the external parameter of the proposed text clustering system, we need to be more careful for setting it to optimize the trade-off between the clustering reliability and the computation time.

This article is organized into the four sections. In Section 2, 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 [8]. 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 [9]. In 2004, Lesile et al. used the version of SVM which proposed by Lodhi et al. to the protein classification [10]. In 2004, Kate and Mooney used also the SVM version for classifying sentences by their meanings [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[15]. 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 [16]. In 2010, Jo proposed the unsupervised neural networks, called Neural Text Self Organizer, which receives the string vector as its input data [17]. 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 [18].

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 [3]. In 2008, Jo applied also his proposed approach to the text clustering, as well as the text categorization [15]. In 2011, Jo described as the technique of automatic text classification in his patent document [13]. In 2015, Jo improved the table matching algorithm into its more stable version [14].

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. In this research, we will modify the machine learning algorithm, AHC algorithm, into the version which processes tables instead of numerical vectors, and use it as the approach to the text clustering.

3. Proposed Approach

This section is concerned with the AHC (Agglomerative Hierarchical Clustering) algorithm as the approach to text categorization, and it consists of the three sections. In section 3.1, we do formally that of computing a similarity between tables into a normalized value between zero and one. In section 3.2, we mention the proposed version of AHC together with its traditional version. This section is intended to describe in detail the proposed version of AHC as the approach to the text clustering task.

3.1 Similarity between Two Tables

This section is concerned with the process of computing a similarity between tables. Texts are encoded into tables by the process which was described in section **??**. The two tables are viewed into the two sets of words and the set of shared words is retrieved by applying the intersection on the two sets. The similarity between the two tables is based on the ratio of the shared word weights to the total weights of the two tables. Therefore, we intend this section to describe in detail and formally the process of computing the similarity.

A table which represents a word may be formalized as a set of entries of words and its weights. The text, D_j is represented into a set of entries as follows:

$$D_j = \{(t_{j1}, w_{j1}), (t_{j2}, w_{j2}), ..., (t_{jn}, w_{jn})\}$$

where t_{ji} is a word included in the text, D_j, and w_{ji} is the weight of the word, t_{ji} in the text,D_j. The set of only words is as follows:

$$T(D_j) = \{t_{j1}, t_{j2}, ..., t_{jn}\}$$

The TF-IDF (Term Frequency - Inverse Document Frequency) weight, w_{ji} of the word, t_{ji} in the text, D_j is computed by equation (1)

$$w_{ji} = \begin{cases} log\frac{N}{DF_i}(1 + logTF_{ji}) & \text{if } TF_{ji} > 0, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

where N is the total number of texts in the corpus, DF_i is the number of texts including the word, t_{ji}, and TF_{ji} is the frequency of the word, t_{ji} in the given text, D_j. Therefore, the table is defined formally as unordered set of pairs of words and their weights.

Let us describe formally the process of computing the similarity between two tables indicating two texts. The two texts,D_1 and D_2 are encoded into the two tables as follows:

$$D_1 = \{(t_{11}, w_{11}), (t_{12}, w_{12}), ..., (t_{1n}, w_{1n})\}$$

$$D_2 = \{(t_{21}, w_{21}), (t_{22}, w_{22}), ..., (t_{2n}, w_{2n})\}$$

The two texts are represented into the two sets of words by applying the operator, $T(\cdot)$, as follows:

$$T(D_1) = \{t_{11}, t_{12}, ..., t_{1n}\}$$

$$T(D_2) = \{t_{21}, t_{22}, ..., t_{2n}\}$$

By applying the intersection to the two sets, a set of shared words is generated as follows:

$$T(D_1) \cap T(D_2) = \{st_1, st_2, ..., st_k\}$$

We construct the table of the shared words and their dual weights among which one is from D_1, and the other is from D_2 as follows:

$$ST = \{(st_1, w_{11}, w_{21}), (st_2, w_{12}, w_{22}), ..., (st_k, w_{k2}, w_{k2})\}$$

The similarity between the two tables is computed as the ratio of the total dual weights of the shared words to the total weights of the ones in both tables, by equation (2).

$$Sim(D_1, D_2) = \frac{\sum_{i=1}^{k}(w_{1i} + w_{2i})}{\sum_{i=1}^{m} w_{1i} + \sum_{i=1}^{m} w_{2i}} \quad (2)$$

It is always given as a normalized value between zero and one; if the two tables, D_1 and D_2 are same to each other, $D_1 = D_2$ the similarity becomes 1.0 as follows:

$$Sim(D_1, D_2) = \frac{\sum_{i=1}^{m}(w_{1i} + w_{2i})}{\sum_{i=1}^{m} w_{1i} + \sum_{i=1}^{m} w_{2i}}$$
$$= \frac{\sum_{i=1}^{m} w_{1i} + \sum_{i=1}^{m} w_{2i}}{\sum_{i=1}^{m} w_{1i} + \sum_{i=1}^{m} w_{2i}} = 1.0$$

If they are exclusive, $T(D_1) \cap T(D_2) = \emptyset$ the similarity becomes 0.0 as follows:

$$Sim(D_1, D_2) = \frac{0}{\sum_{i=1}^{m} w_{1i} + \sum_{i=1}^{m} w_{2i}} = 0.0$$

We demonstrate the process of computing the similarity between two tables using the simple example which is presented in Figure 1. The two texts are encoded into the two source tables as shown in Figure 1. In the example, the two words, 'artificial' and 'documents' are shared by the two tables, and each shared ones have their dual weights from the two input tables. The similarity between the two tables is computed to be 0.52 as a normalized value by equation (2). Therefore, the similarity is computed by lexical matching between the two tables.

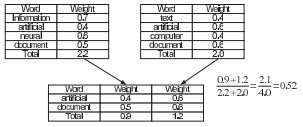

Fig. 1: Example of Two Tables

The similarity computation which is presented above is characterized mathematically. The commutative law applies to the computation as follows:

$$Sim(D_1, D_2) = \frac{\sum_{i=1}^{k}(w_{1i} + w_{2i})}{\sum_{i=1}^{m} w_{1i} + \sum_{i=1}^{m} w_{2i}}$$
$$= \frac{\sum_{i=1}^{k}(w_{2i} + w_{1i})}{\sum_{i=1}^{m} w_{2i} + \sum_{i=1}^{m} w_{1i}} = Sim(D_2, D_1).$$

The similarity is always given as a normalized value between zero and one as follows:

$$0 \leq Sim(D_1, D_2) \leq 1.$$

If the weights which are assigned to all words are identical, the similarity between two tables depends on the number of shared words as follows:

$$Sim(D_1, D_2) \leq Sim(D_1, D_3)$$
$$\rightarrow |T(D_1) \cap T(D_2)| \leq |T(D_1) \cap T(D_3)|.$$

The complexity of computing the similarity between two tables is $O(n \log n)$, since it takes $O(n \log n)$ for sorting the entries of two tables using the quick sort or the heap sort, and $O(n)$ for extracting shared elements by the consequential processing [1].

3.2 Proposed Version of AHC Algorithm

This section is concerned with the proposed AHC version as the approach to the text clustering. Raw texts are encoded into tables by the process which was described in section 3.1. In this section, we attempt to the traditional AHC into the version where a table is given as the input data. The version is intended to improve the clustering performance by avoiding problems from encoding texts into numerical vectors. Therefore, in this section, we describe the proposed AHC version in detail, together with the traditional version.

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.

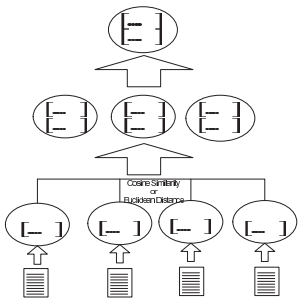

Fig. 2: The Traditional Version of AHC Algorithm

Separately from the traditional version, the clustering process by the proposed AHC version is illustrated in Figure 3. Texts are encoded into tables, and the algorithm begins with unit clusters each of which has a single table. The similarities of all possible pairs of clusters are computed and the pair with its maximum similarity is merged into a single cluster. The clustering proceeds by iterating the process of

computing the similarities and merging a pair. Because the sparse distribution in each table is never available inherently, the poor discriminations by sparse distributions are certainly overcome in this research.

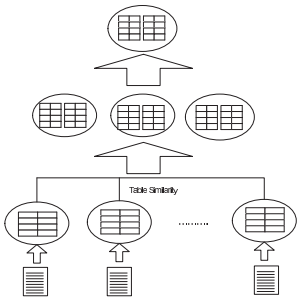

Fig. 3: The Proposed Version of AHC Algorithm

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 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.

Because the tables which represent texts are characterized more symbolically than numerical vectors, it is easier to trace results from clustering items. Let us trace why a particular item is arranged into the cluster, by comparing it with the clusters. From each cluster, we extract words which are shared by it and the cluster members. In each cluster, we present a list of shared ones and their weights, together with the total weight. Therefore, we present the evidence by highlighting the list which corresponds to the cluster and the total weight.

4. Conclusion

We need the remaining tasks for doing the further research. We may apply the proposed approach for clustering texts in the specific domains such as medicine, law, and engineering. We may consider the semantic relations among different words in the tables in compute their similarities, but it requires the similarity matrix or the word net for doing so. We may install the process of optimizing weights of words as the meta-learning tasks. We may implement the text clustering system, adopting the proposed approach.

5. Acknowledgement

This work was supported by 2017 Hongik University Research Fund.

References

[1] M.J. Folk, B. Zoellick, and G. Riccardi, File Structures: An Object Oriented with C++, Addison Wesley, 1998.
[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 and D. Cho, “Index Based Approach for Text Categorization", 127-132, International Journal of Mathematics and Computers in Simulation, No 2, 2008.
[4] T. Jo, “Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578", pp875-882, Journal of Korea Multimedia Society, No 11, 2008.
[5] T. Jo, “Topic Spotting to News Articles in 20NewsGroups with NTC, , Lecture Notes in Information Technology", pp50-56, No 7, 2011.
[6] T. Jo, “Definition of String Vector based Operations for Training NTSO using Inverted Index", pp57-63, Lecture Notes in Information Technology, No 7, 2011.
[7] T. Jo, “Definition of Table Similarity for News Article Classification", pp202-207, The Proceedings of The Fourth International Conference on Data Mining, 2012.
[8] F. Sebastiani, “Machine Learning in Automated Text Categorization", pp1-47, ACM Computing Survey, Vol 34, No 1, 2002.
[9] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text Classification with String Kernels", pp419-444, Journal of Machine Learning Research, Vol 2, No 2, 2002.
[10] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, “Mismatch String Kernels for Discriminative Protein Classification", pp467-476, Bioinformatics, Vol 20, No 4, 2004.
[11] R. J. Kate and R. J. Mooney, “Using String Kernels for Learning Semantic Parsers", pp913-920, Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, 2006.
[12] T. Jo, “Single Pass Algorithm for Text Clustering by Encoding Documents into Tables", pp1749-1757, Journal of Korea Multimedia Society, Vol 11, No 12, 2008.
[13] T. Jo, “Device and Method for Categorizing Electronic Document Automatically", Patent Document, 10-2009-0041272, 10-1071495, 2011.
[14] T. Jo, “Normalized Table Matching Algorithm as Approach to Text Categorization", pp839-849, Soft Computing, Vol 19, No 4, 2015.
[15] T. Jo, “Inverted Index based Modified Version of K-Means Algorithm for Text Clustering", pp67-76, Journal of Information Processing Systems, Vol 4, No 2, 2008.
[16] T. Jo, “Representationof Texts into String Vectors for Text Categorization", pp110-127, Journal of Computing Science and Engineering, Vol 4, No 2, 2010.
[17] T. Jo, “NTSO (Neural Text Self Organizer): A New Neural Network for Text Clustering", pp31-43, Journal of Network Technology, Vol 1, No 1, 2010.
[18] T. Jo, “NTC (Neural Text Categorizer): Neural Network for Text Categorization", pp83-96, International Journal of Information Studies, Vol 2, No 2, 2010.