Building a Concept Hierarchy by Hierarchical Clustering with Join/Merge Decision

Huang-Cheng Kuo[1,*] Tsung-Han Tsai[1] Jen-Peng Huang[2]

[1]Department of Computer Science and Information Engineering
National Chiayi University, Taiwan 600
*hckuo@mail.ncyu.edu.tw
[2]Department of Information Management
Southern Taiwan University of Technology, Taiwan 710

Abstract

Concept hierarchies are important for generalization in many data mining applications. We propose a method to automatically build a concept hierarchy from a provided distance matrix. The method is a modification of traditional agglomerative hierarchical clustering algorithm. When two closest clusters are selected for combining into a new cluster, the algorithm either creates a new cluster with the two original clusters as its sub-clusters, or let a cluster join the other without creating a new cluster at the higher level of the hierarchy. For the purpose of algorithm evaluation, a distance matrix is derived from the concept hierarchy built by algorithm. Root mean squared error between the provided distant matrix and the derived distance matrix is used as evaluation criterion. Empirical results show that the traditional algorithm under complete link strategy performs better than the other strategies, our algorithms perform almost the same under the three strategies, and our algorithms perform better than the traditional algorithms under various situations.

Keywords: Concept Hierarchy, Data Mining, Hierarchical Clustering

1. Introduction

Concept hierarchy, or called taxonomy, is usually in the form of tree. It is an important tool for capturing the generalization relationship among objects. Concept hierarchies exist in many data mining applications. For example, multiple level association rule mining [7, 8, 16, 11] is based on assuming the existence of concept hierarchy.

Concept hierarchies are usually built by domain experts. It is not practical in many applications. For example, in a supermarket data mining application, manually building the taxonomy of items is a very laboring job. Moreover, such taxonomies are hard to reflect the changing customer purchasing behaviors which subsequently affect the similarities between items. Therefore, it is obviously that automatically building a taxonomy based on the similarities between objects is desired.

With the similarities between objects, an intuitive way for building a concept hierarchy is hierarchical clustering. However, internal nodes in trees built by traditional agglomerative hierarchical clustering algorithm are all degree 2. This is not a regular form of concept hierarchy. We propose algorithms for automatically building concept hierarchies from a given distance matrix of the objects. In a concept hierarchy, it is very likely that more than two concepts form a common concept. In order to capture such characteristics of concept hierarchy, the clustering algorithms should allow more than two clusters merge into a cluster. We allow two clusters "join" by merging their children, i.e., sub-clusters, into a cluster. Fig. 1 shows an example of expected concept hierarchy on drink.

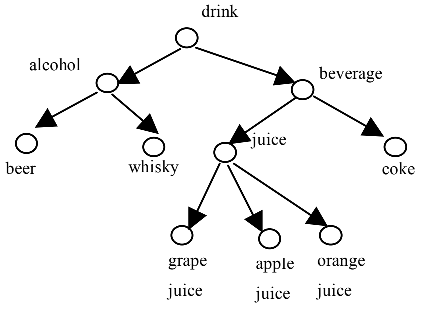

Fig. 1: An expected concept hierarchy

The rest of the paper is organized as follows. In section 2, the measurement for the algorithms is presented and the way to obtain the distance matrix is discussed. Section 3 discusses the algorithms to build a concept hierarchy from a given distance matrix among the objects. Experiment description and the result are in section 4. Conclusion is in section 5.

2. Measurement

There are some metrics for quality of clustering, such as intra-cluster distance, inter-cluster distance, and Dunn's validity index [3]. In information retrieval community [14, 15, 18], there are measurements for concept hierarchy on documents, such as the percentage of reserved taxonomic relations [4], top_k inclusion rate [20]. But, these are not suitable for our method since it is hard to obtain ideal concept hierarchies, and, our method does not concern clusters.

Input to the algorithms is a distance matrix, denoted as provided distance matrix. Since a correct concept hierarchy is usually not available, we propose an indirect measurement. In order to compare with the given distance matrix, the output concept hierarchy is converted into a distance matrix, denoted as derived distance matrix. The derived distance between two objects is inverse proportional to the level of their nearest common ancestor. Distance between two objects is 1 if their nearest common ancestor is the root. Root mean squared error (rmse) between the two distance matrices is served as the measurement of the output concept hierarchy quality.

As for obtaining the provided distance matrices, there are methods for different types of data. There are methods for different types of data to obtain distance matrices. For data in categorical attributes, we can adopt the similarity definition from CACTUS [5] with simplification. We use connectivity to define similarity between categories [12, 13]. Let dataset D = $\{d_1, d_2, \cdots, d_n\}$. D is subset of $D_1 * D_2 * \cdots * D_k$, where D_i is a categorical domain, for $1 \leqq i \leqq k$. Tuple $d_i = <c_{i1}, c_{i2}, \ldots, c_{ik}>$.

There is a link between categories *x* and *y* in domain D_i if there is a pair of tuples (d_u, d_v) having *x* and *y* as their i^{th} attribute values and have the same categorical value in an attribute j other than attribute i.

$$Links^{i}_{x,y}(D) = \{< d_u, d_v, j >| c_{ui} = x, c_{vi} = y,$$
$$c_{uj} = c_{vj}, 1 \le u \le n, 1 \le v \le n, i \ne j\}, x, y \in D_i, x \ne y$$

The similarity between two categories *x* and *y* in attribute *i* is defined as the number of links to total number of pairs having *x* and *y* as their i^{th} attribute values.

For market basket data, ROCK uses interconnectivity [9] to define the similarity between two items in the universal item set. For items x and y in a transaction database, the similarity can be defined by the support of 2-itemset {x, y} [1]. Other metrics for similarity between any two items can also be considered, such as Jaccard coefficient [10, 6].

There are two categories of measurements on the distance between two academic documents TF-IDF [17] in information retrieval and reference-based distance. Two academic documents are dissimilar if they share few or none citations. But if they share similar citations, they can be considered similar [2].

3. Building a Concept Hierarchy

It is intuitive to use traditional agglomerative hierarchical clustering for building a concept hierarchy. Hierarchical clustering treats each object as a singleton cluster, and then successively merges clusters until all objects have been merged into a single remaining cluster. The strategies for computing the distance between a pair of clusters used in this paper are single link [19], average link, and complete link [10].

In our modified hierarchical clustering, two clusters can either "merge" into a new cluster at the upper level of the tree or "join" together. When two clusters merge, they become the children of the newly created cluster. When two clusters join together, either (1) they form a new cluster whose children (sub-clusters) are the children of the two original clusters, or (2) a cluster becomes a child of the other cluster. However, two singleton clusters can only merge. It makes no sense for them to join together.

Let $dist_{clu}(X, Y)$ be the average distance between an object in cluster X and an object in Y. Let clusters A and B have sets of sub-clusters $\{A1, \cdots, Am\}$ and $\{B1, \cdots, Bn\}$, respectively. Then, $n_A = m(m+1)/2$ and $n_B = n(n+1)/2$ are the numbers of sub-cluster pairs in A and B, respectively. We denote $dist_{comp}(X)$ as the total distance of $dist_{clu}(Ai, Aj)$, and $dist_{comp\_avg}(A)$ as the average distance of $dist_{clu}(Ai, Aj)$, where Ai and Aj are two sub-clusters in A. The weighted average distance $dist_{comp\_avg}(A, B)$ of A and B is $(dist_{comp}(A) + dist_{comp}(B)) / (n_A + n_B)$. When the pair of cluster A and B is chosen by the hierarchical clustering algorithm for next iteration, the following conditions are used to determine the type of merge or join. The conditions are checked sequentially.

1. Two singleton clusters merge into a new cluster.
2. If $dist_{clu}(A, B) - dist_{comp\_avg}(A, B) > \sigma$, where σ is the standard deviation of the distance in the provided distance matrix, then A and B are merged. This means that though the pair of clusters A and B is chosen for next clustering iteration, their cluster-to-cluster distance is still quite large. So, they are not suitable to join together.
3. Given a small fraction β, say β = 0.05, if $|dist_{comp\_avg}(A)-dist_{comp\_avg}(B)| / dist_{comp\_avg}(A, B) < \beta$, then sub-clusters of A and B are sub-clusters of a new cluster.
4. If $dist_{comp\_avg}(A) > dist_{comp\_avg}(B)$, meaning objects in sub-clusters of A are more diverse than objects in sub-clusters of B, then B becomes a sub-cluster of A.

5. Else, A becomes a sub-cluster of B.

The total distance between sub-clusters of a resulting cluster X, $\text{dist}_{\text{comp}}(X)$, has to be updated after merging or joining clusters A and B. In Fig. 2(b), $\text{dist}_{\text{comp}}(C) \leftarrow \text{dist}_{\text{clu}}(A, B)$. The distance $\text{dist}_{\text{clu}}(A, B)$ has been computed when the pair (A, B) is chosen if average link is applied for computing cluster to cluster distance. In Fig. 2(c), $\text{dist}_{\text{comp}}(C) \leftarrow \text{dist}_{\text{comp}}(A) + \text{dist}_{\text{comp}}(B) + \text{dist}_{\text{clu}}(A, B)$. In fig. 2(d), $\text{dist}_{\text{comp}}(A) \leftarrow \text{dist}_{\text{comp}}(A) + \text{dist}_{\text{clu}}(A, B)$.

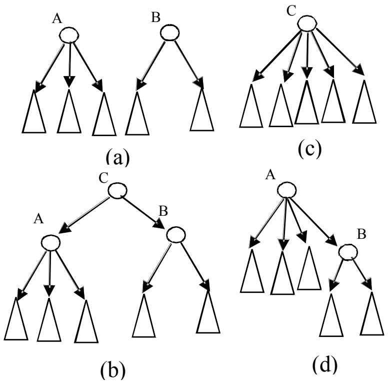

Fig. 2: The cases of merging or joining. (a) Clusters are to be processed in next clustering iteration. (b) They merge into a new cluster. (c) Their sub-clusters form a new cluster. (d) Cluster B becomes a sub-cluster of cluster A

4. Experiment Result

A provided distance matrix of *n* objects is generated with the assistance of a tree described in the following steps.

1. Start with a root.

2. The number of an internal node is uniformly distributed in a specified range, denoted as span. Generate a random number *s* in this range.

3. Randomly choose a leaf node. Create *s* children for the chosen node.

4. Repeat steps 2 and 3 until there are *n* leaf nodes.

5. Compute the distance between all pairs of categories.

6. Noise is applied on the distance matrix. Uniformly distributed numbers between (1-noise) and (1+noise) is multiplied to the distance values.

In the experiment, we illustrate the performance of the algorithms under three parameters, namely noise, span, and number of items. For each parameter combination, root mean squared error values of 100 tests are averaged. For generating provided distance matrices, we build trees with two intervals of span: [2, 6] and [2, 10].

We compare the root mean squared error (rmse) between the provided distance matrices and the distance matrices derived from the trees built by the algorithms. In figures 3 and 4, the lines MS, MA, and MC represent for our new modified algorithm under the three cluster-to-cluster distance strategies, namely single link (S), average link (A), and complete link (C), respectively. The lines TS, TA, and TC represent for traditional algorithm under the three strategies, respectively.

The spans of internal nodes are in two ranges: the narrower range [2, 6] and the wider range [2, 10]. Figure 3 shows the result of the narrower span range case. We observe that (1) our modified method is better than the traditional algorithm for both noise levels under all the three cluster to cluster distance strategies. (2) The gap between our method and the traditional method is larger when the noise is smaller. (3) Among the three strategies, single link is best for our modified method; average link is best for the traditional method. Complete link is worst for both methods. (4) As the number of objects increases, the root mean squared error decreases. But, the trend stops when the number of objects is more than around 400.

Figure 4 shows the result of the wider span range case. We have similar observations as in Fig. 4. Furthermore, there are something else to notice: (1) Both methods under the three strategies perform worse when the span of internal is wider. (2) The gap between the two methods is smaller as compared to fig. 4. (3) The root mean squared error decreases as the number of objects increases. But, the decreasing trend stops at around 600 objects.

5. Conclusion

Concept hierarchy is a useful mechanism for representing the generalization relationships among objects. So that, multiple level association rule mining can be conducted. In this paper, we build a concept hierarchy from a distance matrix with the goal that distance between any pair of objects is preserved as much as possible.

We adopt the traditional agglomerative hierarchical clustering. However, clusters do not only merge, but also join. Empirical results show that our modified algorithm has much better performance than the original algorithm.

This study can further be improved in some directions: (1) All the lengths, i.e., weights, of edges of the concept hierarchy are the same. If weights on edges of the concept hierarchy can be trained, the distance relationship between objects can be better preserved. (2) A better measurement for the methods might be devised, so that the resulting concept hierarchy can be directly evaluated. (3) In the information retrieval research field, document directories are often available. Definitions of distance between documents are also available. Experiments on automatically obtaining directories and comparing them to existing directories can be conducted to understand the algorithms on real data.

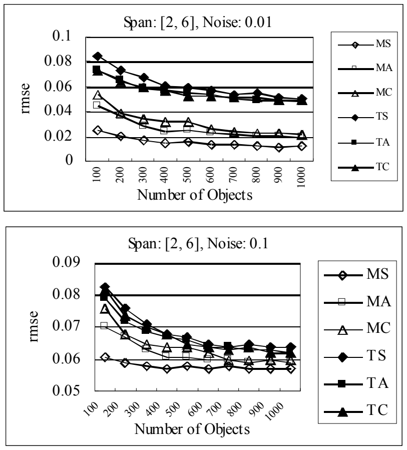

Fig. 3: Experiment set 1

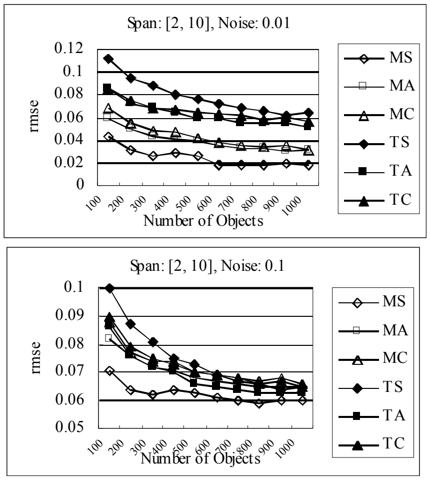

Fig. 4: Experiment set 2

6. References

[1] C. C. Aggarwal, J. L.Wolf, P. S. Yu, "A New Method for Similarity Indexing of Market Basket Data," ACM SIGMOD, pp. 407-418, 1999.

[2] S. Bani-Ahmad, A. Cakmak, G. Ozsoyoglu, "Evaluating Publication Similarity Measures," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, Vol. 28, No. 4, 2005, pp. 21-28.

[3] J. C. Dunn, "Well Separated Clusters and Optimal Fuzzy Partitions," Journal of Cyber, Vol. 4, pp. 95-104, 1974.

[4] K. M. Gupta, D. W. Aha, P. Moore, "Learning Feature Taxonomies for Case Indexing," ECCBR 2004, LNAI 3155, pp. 211-226.

[5] V. Ganti, J. Gehrke, and R. Ramakrishnan, "CACTUS-Clustering Categorical Data Using Summaries," ACM KDD, pp. 73-83, 1999.

[6] S. Guha, R. Rastogi, K. Shim, "ROCK: A Robust Clustering Algorithm for Categorical Attributes," IEEE ICDE Conference, pp. 512-521, 1999.

[7] J. Han and Y. Fu, "Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases," Workshop on Knowledge Discovery in Databases, pp. 157-168, 1994.

[8] J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases," VLDB Conference, pp. 420-431, 1995.

[9] Sudipto Guha, Rajeev Rastogi, Kyuseok Shim, "ROCK: A Robust Clustering Algorithm for Categorical Attributes," IEEE ICDE Conference, 1999, pp. 512 - 521.

[10] A. K. Jain, R. C. Dubes, Algorithms for Clustering Data, Prentice-Hall, 1988.

[11] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2001.

[12] Huang-Cheng Kuo, Yi-Sen Lin, Jen-Peng Huang, "Distance Preserving Mapping from Categories to Numbers for Indexing," International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Lecture Notes in Artificial Intelligence, Vol. 3214, 2004, pp. 1245-1251.

[13] H.-C. Kuo, J.-P. Huang, "Building a Concept Hierarchy form a Distance Matrix," International Conference on Intelligent Information Systems, 2005, pp. 87-95.

[14] D. J. Lawnie, W. B. Croft, A. Rosenberg, "Finding Topic Words for Hierarchical Summarization," ACM SIGIR, 2001, pp. 349-357.

[15] D. J. Lawnie, W. B. Croft, "Generating Hierarchical Summaries for Web Search," ACM SIGIR, 2003, pp. 457-458.

[16] R. Srikant and R. Agrawal, "Mining Generalized Association Rules," VLDB Conference, 1995, pp. 407-419.

[17] G, Salton, Automatic Text Processing, Addison Wesley, 1989.

[18] M. Sanderson, B. Croft, "Deriving Concept Hierarchies from Text," ACM SIGIR, 1999, pp. 206-213.

[19] R. Sibson, "SLINK: An Optimally Efficient Algorithm for the Single-Link Cluster Method," Computer Journal, Vol. 16, No. 1, pp. 30-34, 1972.

[20] J.-H. Wang, C.-C. Huang, J.-W. Teng, L.-F. Chien, "Generating Concept Hierarchies from Text for Intelligent Analysis," ISI Conference, LNCS 3073, 2004, pp. 100-133.