Mining Patterns from Case Base Analysis

Tao Li Shenghuo Zhu Mitsuonri Ogihara
Computer Science Department
University of Rochester
Rochester, NY 14627-0226
{taoli,zsh,ogihara}@cs.rochester.edu

Abstract

In this paper, we present our work on combining domain knowledge and data mining techniques for improve the service and realize cost reduction for a product company. We first extract domain knowledge from the database of call records and then incorporate the domain knowledge into the process of finding similar products and clustering. By finding similar products, we can use successes from one product and apply them to the similar products. By clustering, we can group products into clusters and design improvement strategy for each group. It is projected that our work would be very useful and beneficial to the company.

1 Introduction

Our work is carried out in a high-tech product company. The company manufactures many kinds of products. For many reasons, we are not able to disclose any private information about it. However we will present the core ideas of our work.

Our work was motivated by the cost reduction plan of the company's Help Center. The Help Center of the company is the recipient of the initial call of customers for service. It represents the first level of service provided by the company. A Case Base software is installed at the Help Center for remote diagnostic application. Help Center personnel take calls from customers and ask them questions prompted by the Case Base software. The customer responses are entered into the software and a diagnostic tree is traversed leading to either a solution to the customer's problem or to a call escalation process. In other words, the computational recipient examines the Case Base for the appropriate case number and determines the requisite actions.

The Case Base consists of many lines of text that are used as input to the Intuition software package which constructs a diagnostic tree and provides the prompts, questions, pictures, graphical displays, and instructions which are read by the Case Base worker in the Help Center as he/she engages the customer on conversation. This conversation is normally prompted by a customer call with some sort of problem. The problem may be a lack of information, need to instruction on using some machine feature, ordering parts or consumable supplies, or a machine failure. The case base as it stands today takes its only input from this conversation.

The Case Base consists of nearly 2300 possible cases. Most cases can be applied to each product. Generally, a case is a contextualized piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goals of reasoner [24]. To be more specific, in our Case Base software, a case refers to the path through a decision tree that ends up at a leaf node, the action that resolves or escalates the case. A path represents a specific context. Each of these cases has a number of elements. There are questions associated with each case. The Intuition software uses these questions to prune the list of possible causes for the customer call. Many cases have explanatory text and graphics that are displayed to the Help Center employee and explained to the customer. Finally, each case has an associated action that intends to resolve the customer problem. There are generally two kinds of actions: one is self-repair actions that the customers can repair by themselves according to some instructions, the other is placing service call which means to send request to the second help service tier-parts/consumables ordering or field service or send technicians. In the rest of our paper, we focus our attention on the cases associated with self-repair actions.

So for each customer call, the Case Base worker uses the Intuition software to identify the most promising case corresponding to the customer problem and tell the customer to take the associated action. However, this is not the end of the story. For the cases associated with self-repair actions, for many calls, even if we can correctly identify the corresponding cases and take the associated actions, the problems still cannot be resolved and we need to placing service calls (say, send technicians to the spot) anyway.

For a case C of a given product P, suppose there are A calls correspond to the case C, for B calls the customers successfully fix the problems by taking self-repair actions associated with the Case C, Then we have the following definition:

Definition 1 *We define the successful rate for case C of the given product P to be $Sr(C,P) = \frac{B}{A} \times 100\%$.*

A case generally has different successful rates for different products. We can also define the successful rate for product family (a collection of products).

Definition 2 *The successful rate for case C of a given product family $P = (P_1, P_2, \cdots)$ is defined to be $Sr(C,P) = \frac{B}{A} \times 100\%$, where A is the number of calls correspond to the case C of products in P and B is the number of calls the customers successfully fix the problems by taking self-repair actions.*

All our analysis for products can be easily extended to product families. In the rest of our paper, we assume that we are only deal with products. Moreover, generally most case-based reasoning systems have adaption/repair components, so the successful rates can be changed dynamically. Since our purpose is to discover the efficient scheme of improving strategies, we focus on the snapshot of the system in this paper.

The Help Center receives several millions calls every year and it also places thousands of service calls (to send technicians) annually. In order to realize cost reduction, we need to reduce the number of customer calls and improve the successful rates for each product. To reduce the number of customer calls, we should try to analyze the issue that how much of the Help Center call process could be eliminated by improving the design of the product (say, adding some self-diagnostic functions or some supply information). To improve our service and further realize cost reduction, we must first have an accurate estimate of our current performance and we must be able to assess our strengths and weaknesses. The idea is to learn what we do best and why and how. Then we must improve our weak areas. The end result is a stronger company with more value for our customers, employees and stockholders.

In this paper, we present our work of integrating the knowledge management and data mining techniques to fulfill the above goals. We first explore the case distribution and find what are the frequently retrieved cases so that we can focus our attention on those frequent cases. The case distribution is the domain knowledge. The domain knowledge gives us a general overview of the service requirement and it will help us to improve the service by itself (as we will discuss in Section 6). To be more efficient, we can utilize the domain knowledge and use data mining techniques to better achieve our goals. We compare products against each other to get insight of performance. We then find products that have similar behaviors so that we can use successes from one product and apply them to others. We can apply the successful adaption and repair strategies for one product to its similar products. Finally we cluster products into groups so that we can devise strategies for each group.

The rest of the paper is organized as follows: Section 2 reviews history and related work on case-based reasoning. Section 3 describes our Case Base Analysis and Section 4 discusses finding similar patterns among all the products. Section 5 presents the idea of clustering the products into groups and Section 6 shows the experiment results with some discussions. Finally Section 7 concludes our work.

2 Review on Case-based Reasoning

Over the last few years, case-based reasoning (CBR) has grown from a rather specific and isolated research area to a field of widespread interest [1]. Case-based reasoning tries to utilize the specific knowledge of previously experienced, concrete problem situations instead of relying solely on general knowledge of a problem domain. In other words, case-based reasoning remembers previous situations similar to the current one and uses them to help solve the new problem. The roots of case-based reasoning can be found in the works of Roger Schank on dynamic memory and the central role that a reminding of earlier situations and situation patterns has in problem solving and learning [35]. Case-based reasoning is applicable to a wide range of real-world situations, ranging from knowledge-rich situations in which construction of solutions is complex to knowledge-poor situations in which cases provide the only available knowledge. It has many advantages: allowing a reasoner to propose solutions to problem quickly, to reason in domains that are not well understood, to evaluate solutions when algorithmic methods are not available *etc.* [24]. A lot of CBR systems have been applied to Help Desk applications because problems often recur in the domain and when thy do so they utilize the same solutions [37, 25, 40]. Because no old situation is ever exactly the same as the new one, old solutions must usually be adapted to made applicable to new situations. Also in order to learn from experience, a case-based reasoning system should be able to **evaluate** and consequently **repair** the system based on feedback and simulation [24]. Various adaptation and repair approaches have been proposed [24, 38, 41, 27, 28, 17, 26].

Although our work is based on a case-based system, it is quite different from the application of CBR to Help Desk and those works on adaptation and repair strategies. The Help Center of the company already has a case-based reasoning system installed for remote diagnostics. Our work is try to first extract the domain knowledge from the case-base analysis, *i.e.*, the statistics of using case-based system. Then, we apply the domain knowledge to mine similar patterns from the products or cluster the products into groups so that we can design improvement strategies for product groups instead of individual products. In other words, we want to find similar products and cluster the products into groups so that we can efficiently design the adaptation/repair strategies.

3 Case Base Analysis

On one hand, although there are about 2300 possible cases, some of them are frequently retrieved and some of them are rarely retrieved. To efficiently save cost, we can focus on improving the successful rate for most frequent cases. So our first step is to try to find the most frequent cases.

For each product, we collect the call records in Help Center from May 2001 to July 2001. The call records contain the information for each call: the call time, the possible case number corresponding to the call (indicated by Case Base software), whether the self-repair actions worked or not, *etc.* We processing the call records and we find that generally for all products *about* 80% *of the customer calls corresponding to* 20% *of all the cases*. The 20% *cases account for about* 80% *customer calls* is the domain knowledge we get from the database of call records.

Then for each product, using the domain knowledge, we can focus on its top 20% cases and then find out what are the cases that have high successful rates and what cases do customers call us for the most. These findings would give us a clear view on the help service performance for each product. For example, we found that about 10% of the customer calls for product A are asking for some parts information. This leads the product development department to consider including the parts information into the product. This may immediately result in the cost reduction for Help center.

However each product may have different top 20% cases. To compare different products or product families, we then categorize the cases into several categories. The categorization is unique for all products and it provides a comparison basis. It is obtained from the suggestions of the experienced technicians and is based on the context description/situation of the cases. For example, the cases which are related to the failure of the same part are classified into one category and the cases that deal with the instructions on using some machine feature are in another category.

Definition 3 *The successful rate for a category H of product P is defined to be $Sr(H,P) = \frac{B}{A} \times 100\%$, where A is the number of calls correspond to the cases in category H and B is the number of calls the customers successfully fix the problems by taking self-repair actions.*

After the categorization, we can continue to process the call records and answer the questions like: What categories do customers call us for the most and which categories have the high successful rates, *etc.* These questions provide us a high-level view on the service performance.

4 Finding Similar Products

After we get enough information for each product, we want to compare products against each other to get insight of performance. We want find products that have similar behaviors so that we can use successes from one product and apply them to others.

As described in Section 3, using domain knowledge, we categorize the top 20% cases for each product and compute the successful rate for each category. So we can represent each product as an n-dimensional vector $p = (S_1, S_2, \cdots, S_n)$, where n is the number of categories and $S_1, S_2, \cdots, S_n$ are the successful rates for each categories respectively.

Suppose now we have m products $p_k = (S_1^k, S_2^k, \cdots, S_n^k), k = 1, 2, \cdots, m$, we compute the mean vector, M, the covariance matrix, Λ, and the deviation vector, d, as follows:

$$\begin{aligned} M &= (M_i) \quad i = 1, 2, \cdots, n \\ M_i &= \frac{1}{m}\sum_{k=1}^{m} S_i^k \\ \Lambda &= (\lambda_{ij}) \quad i, j = 1, 2, \cdots, n \\ \lambda_{ij} &= \frac{1}{m}\sum_{k=1}^{m}(S_k^i - M_k)(S_k^j - M_k) \\ d &= (d_i) \quad i = 1, 2, \cdots, n \\ d_i &= \sqrt{\lambda_{ii}} \end{aligned} \tag{1}$$

We call two products $p_k = (S_1^k, S_2^k, \cdots, S_n^k)$ and $p_l = (S_1^l, S_2^l, \cdots, S_n^l)$ similar if

$$\frac{|S_i^k - S_i^l|}{d_i} \leq c \quad i = 1, 2, \cdots, n, \tag{2}$$

where c is a given threshold (generally, $c = 1$). Then we can determine whether two products have similar behaviors.

Suppose we find that A and B are similar, which means that each category has approximately successful rate for A and B, if we have some adaptation/repair strategies for the cases of A, they might probably be suitable for product B too. For example, if we can promote the successful rates for some categories of product A by collecting more status information of some specific machine parts, we can also apply the same strategy to a similar product B.

5 Clustering Products into Groups

5.1 Clustering Introduction

First, we give a brief introduction to clustering problems. Clustering problems arise in many disciplines including engineering, business and social science, and have a wide range of applications, such as data compression, information retrieval, pattern recognition, trend analysis, customer segmentation and classification. The problem of clustering has been studied extensively in the database [43, 18, 33, 12, 14, 13], statistics [5, 4, 9, 8, 34, 32] and machine learning communities [7, 15, 16, 29, 30] with different approaches and different focuses.

The clustering problem can be described as follows: let V be a set of n multi-dimensional data points, we want to find a partition of V into clusters such that the points within each cluster are *similar* to each other. Various distance functions have been widely used to define the measure of similarity.

Clustering techniques can be broadly classified into partitional clustering, hierarchical clustering, density-based clustering and grid-based clustering [20]. Partitional clustering attempts to directly decompose the data set into K disjoint clusters such that the data points in a cluster are nearer to one another than the data points in other clusters. Several well-known algorithms such as K-means [31], PAM (Partitioning Around Medoids) [33], K-Medoids [23] and K-mode(K-prototypes for clustering categorical data) [22], *etc.*, fall into the category. Hierarchical clustering proceeds successively by building a tree of clusters. It can be viewed as a nested sequence of partitioning. The tree of clusters, often called *dendrogram*, shows the relationship of the clusters and a clustering of the data set can be obtained by cutting the dendrogram at a desired level. BIRCH [43, 44], CURE [18] and ROCK [19] are representative hierarchical clustering algorithms. Density-based clustering is to group the neighboring points of a data set into clusters based on density conditions. DBSCAN [11, 10], OPTICS [3] and DENCLUE [21] are well-known algorithms of this category. Grid-based clustering quantizes the object space into a finite number of cells that form a grid-structure and then performs clustering on the grid structure. Several representatives of this category are STING(Statistical Information Grid-based method) [39] and WaveCluster [36].

Detailed survey on clustering methods can be found in [42, 8].

5.2 Clustering Application: Grouping Products

In this subsection, we describe our work on clustering products into groups. By clustering, we can then try to explore the common features in each group and design improvement strategies for each group. First, we need to define a distance functions between products to measure their similarity.

As described in Section 4, we have all the products $p_k = (S_1^k, S_2^k, \cdots, S_n^k), k = 1, 2, \cdots, m$ and we also get the mean vector, M, and the covariance matrix, Λ.

Now we view the products as points in n-dimensional Euclidean space and define the distance between two products p_k and p_l as the weighted Euclidean distance in the Euclidean space:

$$dist(p_k, p_l) = \sqrt{(p_k - p_l)W^{-1}(p_k - p_l)^T} \tag{3}$$

where weighting matrix W^{-1} is a positive definite matrix. Different categories have usually different deviations. To avoid a certain category dominating the value of the distance, the inverse of covariance matrix Λ is considered as the optimal weighting matrix (see, e.g., [6]).

To simplify the model, it is assumed that categories are uncorrelated with each other. So the simplified version of distance is

$$dist(p_k, p_l) = \sqrt{\sum_{i=1}^{n} (\frac{s_i^k - s_i^l}{d_i})^2} \tag{4}$$

After we define the distance, we can now cluster the products p_k into groups. There are a multitude of clustering methods proposed in literature as we discussed in Section 5.1. We choose density-based clustering since we do not have a good way to decide the input parameter K for K-means and K-medoid algorithms and partitioning methods are unable to handle noise and outliers efficiently. Also hierarchical clustering is too time-consuming (usually quadratic algorithms) and we are not sure about the granularity of the lowest level of grid for grid-based approaches.

In density-based clustering, clusters are regions of space that have a high density of points and clusters can have arbitrary shapes. We use DBSCAN algorithm and for the details of the DBSCAN algorithm, please see [20, 11, 10]. Its main idea is as follows: it first checks the neighborhood of each point in the database. If the neighborhood of some point P contains a lot of points (more than a predefined number), then the neighborhood constitutes a cluster with P as the center of cluster. It then iteratively extends the clusters until no new points can be added to any cluster.

6 Experimental Results

As we mentioned above, we have found that *20% cases account for 80% of customer calls* and this holds for almost all products. For each product, we then focus on the top 20% cases. After we further investigate those cases, we noticed that many of the questions, which were previously asked by customers during the Help Center conversations, might be solved by using the data that are available in the device in question, or if we added some documents into the device. For example, we have found that for a product A, 2% of the calls for the product are asking about where to touch a function button and 5% the calls are asking about supply information of some parts. Thus this gives the company useful evidence and suggestion on cost reduction by adding more necessary information into the products.

In our work, we have more than 2000 products and the number of calls from May 2001 to July 2001 is more than one million. For each product, we have about 400 cases (20% out of all the cases). We categorize the cases into six categories: $C1, C2, C3, C4, C5, C6$.

For all the products we get the deviation vector $d = (0.072, 0.045, 0.026, 0.037, 0.17, 0.033)$. And we have found that lots of similar product pairs. Figure 1 and Figure 2 illustrate that product $P1$ is similar to product $P2$; product $P3$ is similar to product $P4$. As you can find from the figures, the differences of category values between similar products satisfy Equation 2.

Knowing similarity between products, we know what the values and elements of performance are and we then can use successes from one product and apply them to others.

For the clustering, we set the radius of the neighborhood to be $\epsilon = 0.015$, the minimum number of points required for a neighborhood to be a cluster is 30. Finally we cluster products into 16 groups.

By grouping the products in clusters, we can easily find the weakness and strength for each cluster and formulate intelligent improvement plan accordingly.

7 Conclusions

In this paper, we have presented a case study on combining domain knowledge and data mining techniques for improve the service and realize cost reduction for a product company. Clustering techniques have been employed to divide the products into similar groups based on the knowledge extracted from the database of call records.

In many application domains, knowledge management and data mining need to be fully

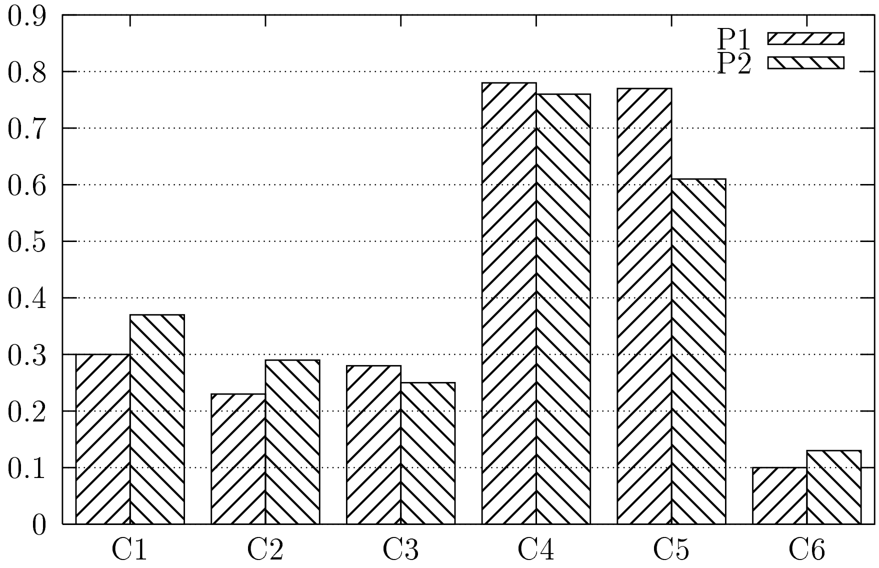

Figure 1: Similarity of $P1$ and $P2$. The vertical axis represents the category values S_i^k. The horizontal axis represents different categories.

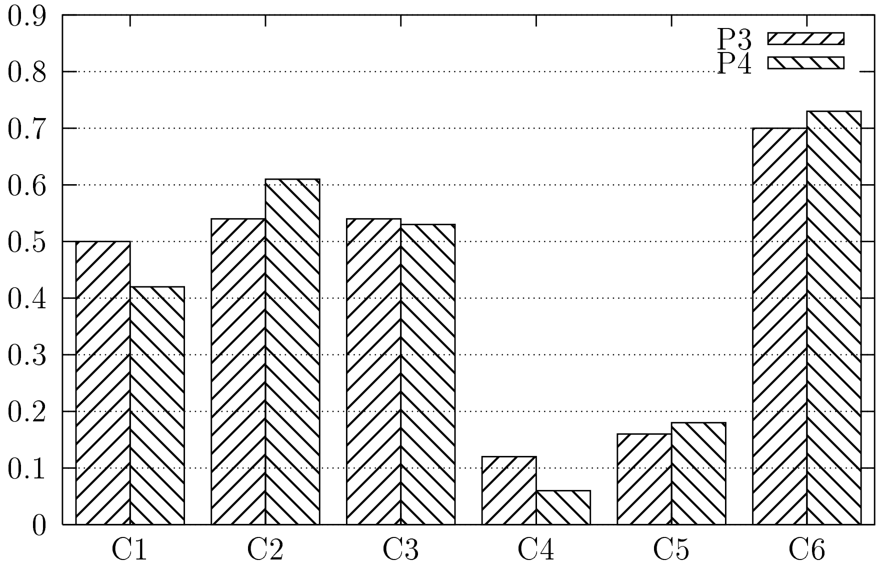

Figure 2: Similarity of $P3$ and $P4$. The vertical axis represents the category values S_i^k. The horizontal axis represents different categories.

integrated to provide more support and benefit to the organization. Domain knowledge can be captured or extracted through by direct or indirect knowledge acquisition. It is a useful way of constraining or pruning the search space for discovery process, enhancing the performance of data mining tasks[2]. Thus incorporating the knowledge management and data mining would improve the performance of the system and ensure data mining availability profitability.

Acknowledgment

We would like to thank anonymous reviewers for their helpful suggestions.

References

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.

[2] Sarabjot S. Anand, David A. Bell, and John G. Hughes. The role of domain knowledge in data mining. In *CIKM*, pages 37–43, 1995.

[3] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. In Alex Delis, Christos Faloutsos, and Shahram Ghandeharizadeh, editors, *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadephia, Pennsylvania, USA*, pages 49–60. ACM Press, 1999.

[4] M. Berger and I. Rigoutsos. An algorithm for poinr clustering and grid generation. *IEEE Transactions on Systems, Man and Cybernetics*, 21(5):1278–1286, 1991.

[5] M.R. Brito, E. Chavez, A. Quiroz, and J. Yukich. Connectivity of the mutual K-Nearest-Neighbor graph for clustering and outlier detection. *Statistics and Probability Letters*, 35:33–42, 1997.

[6] P. E. Caines and L. Ljung. Asymptotic normality of prediction error estimators. Control System Report 7602, Dept of Electrical Engineering, University of Toronto, Canada, 1976.

[7] Peter Cheeseman, James Kelly, and Matthew Self et al. AutoClass: A bayesian classification system. In *Proc. of the 5th Int'l Conf. on Machine Learning*. Morgan Kaufman, June 1988.

[8] R. Dubes and A.K. Jain. Clustering methodologies in exploratory data analysis. In M.C. Yovits, editor, *Advances in Computers*, volume 19. Academic Press, New York, 1980.

[9] Richard Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

[10] M. Ester, H. Kriegel, J. Sander, and X. Xu. Density-connected sets and their application for trend detection in spatial databases. In *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97)*, pages 10–15, Newport Beach, California, 1997.

[11] Martin Ester, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD 1996*, pages 226–231, 1996.

[12] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. A database interface for clustering in large spatial databases. In *Proc. of 1st Int'l Conf. on KDD*, 1995.

[13] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. A density based algorithm for discovering clusters in large spatial databases with nosies. In *Proc. of 1st Int'l Conf. on KDD*, 1995.

[14] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In *Proc. of 4th Int'l Symposium on Large Spatial Databases*, 1995.

[15] Douglas H. Fisher. Knowledge acquisition via incremental conceptal clustering. *Machine Learning*, 2(2), 1987.

[16] Douglas H. Fisher. Iterative optimization and simplification of hierarchical clusterings. Technical Report CS-95-01, Department of Computer Science, Vanderbilt University, Nashville, TN 37235, 1995.

[17] P. A. González-Calero, M. Gómez Albarrán, and B. Díaz Agudo. A substitution-based adaptation model. In *Challenges for Case-Based Reasoning - Proc. of the ICCBR'99 Workshops*, 1999.

[18] S. Guha, R. Rastogi, and K. Shim. CURE: An efficient clustering algorithm for large database. In *Proceeding s of the 1998 ACM SIGMOD Conference*, pages 73–84, 1998.

[19] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345–366, 2000.

[20] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2000.

[21] Alexander Hinneburg and Daniel A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[22] Zhexue Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *Research Issues on Data Mining and Knowledge Discovery*, 1997.

[23] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley & Sons, New York, 1990.

[24] Janet Kolodner. *Case-Based Reasoning*. Morgan Kaufmann, 1993.

[25] M. Kriegsman and R. Barletta. Building a case-based help desk application. *IEEE Expert*, 8, 1993.

[26] David B. Leake, Andrew Kinley, and David C. Wilson. Acquiring case adaptation knowledge: A hybrid approach. In *AAAI/IAAI, Vol. 1*, pages 684–689, 1996.

[27] David B. Leake, Andrew Kinley, and David C. Wilson. Linking adaptation and similarity learning. In *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*. Lawrence Erlbaum, 1996.

[28] David B. Leake and David C. Wilson. Combining CBR with interactive knowledge acquisition, manipulation and reuse. In *ICCBR*, pages 203–217, 1999.

[29] Michael Lebowitz. Experiments with incremental concept formation: UNIMEM. *Machine Learning*, 1987.

[30] Bing Liu, Yiyuan Xia, and Philip S. Yu. Clustering through decision tree construction. In *SIGMOD-00*, 2000.

[31] J. MacQueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathemtical Statistics and Probability*, pages 281–297, 1967.

[32] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 1983.

[33] Raymond T. Ng and Jiawei Han. Efficient and effective clustering methods for spatial data mining. In *Proc. of VLDB*, 1994.

[34] R.Lee. Clustering analysis and its applications. In J. Toum, editor, *Advances in Information Systems Science*, volume 8, pages 169–292. Plenum Press, New York, 1981.

[35] R. Schank. *Dynamic Memory: A Theory of Learning in Computers and People*. Cambridge University Press, 1982.

[36] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 428–439, 1998.

[37] Evangelos Simoudis and James S. Miller. The application of CBR to help desk applications. In *Proceedings of the DARPA Case-Based Reasoning Workshop*, pages 25–38, 1991.

[38] Katia Sycara. Using case-based reasoning for plan adaptation and repair. In *Proceedings Case-Based Reasoning Workshop*, Clearwater Beach, Florida, 1988.

[39] Wei Wang, Jiong Yang, and Richard R. Muntz. STING: A statistical information grid approach to spatial data mining. In *The VLDB Journal*, pages 186–195, 1997.

[40] I. Watson. *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. Morgan Kaufmann, San Mateo, CA, 1997.

[41] David C. Wilson. *Case-Base Maintenance: The Husbandry of Experience*. PhD thesis, Indiana University, 2001.

[42] M. Zait and H. Messatfa. A comparative study of clustering methods. *FGCS Journal, Special Issue on Data Mining*, 1997.

[43] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An efficient data clustering method for very large databases. In *ACM SIGMOD Conference*, 1996.

[44] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, 1997.