An Exploratory Study of Applying Case-Based Reasoning to Business Applications

Hajin Hwang
(Department of MIS, Catholic University of Taegu Hyosung)

ABSTRACT

As the effective use of information has gained greater attention over the decade, various conventional AI techniques have been applied to develop expert systems for business applications. Case-based reasoning(CBR) makes data more accessible by organizing it as a set of examples from past experience that can be generalized and applied to current problems. This paper illustrates basic concepts of CBR and addresses the opportunity of its application to business fields. CBR system discussed in this paper can provide a basis for building more flexible and adaptable expert systems for business applications.

I. Introduction

While the complexities and capabilities of information technology are rapidly increasing, making effective use of information has gained greater

attention over the years. With the current trends of data explosion stemming from the nature of the information age, distilling information overload into useful knowledge represents the major challenge in moving to a new knowledge age. In recent years, rule-based expert systems have been applied to many business fields to achieve competitive advantage. Utilizing the essential business knowledge, estimating and predicting the future courses of action for the corporate to take will be critical to efficient and effective use of corporate resources, and consequently to the business success.

Examining all relevant variables, the number of rules to generate to estimate and determine future courses of actions is usally quite large and time consuming to generate. Typically, this knowledge must include how to decompose the project into smaller tasks and accurately estimate each portion. Therefore, scratch is enormous. Case-based reasoning(CBR) avoids this knowledge acquisition bottleneck by using the wealth of existing knowledge embodided in past cases[4,8,15,17]. Since the business decision is not general from scratch but is adjusted from a previous experience, less specific and less accurate knowledge is required.

In addition, the use of rule-based expert systems to determine business courses of action is fraught with difficulties. Rules are required to analyze every possible factor's construction. This problem is exacerbated by the fact that in actual domains interaction among factors is common. The possible number of pair-wise interactions is the square of the number of

factors, and factors may interact in groups larger than two. CBR avoids these problems by retrieving previous examples with similar values for these factors, so their effects and interactions are included implicitly.[8,11]

Neural networks also make use of past cases in the form of learning, but they cannot easily justify the prediction they make.[7,17] CBR systems can point to the similar cases on which the prediction is based as justification. Any required adjustments from these cases are usally small and therefore credible. In addition, since the knowledge is in symbolic form, richer meaning can be conveyed. For example, an architectural design project might have gone over budget because of numerous change requests from the client. A text comment describing this reason can be stored along with the case. Later, when this project is retrieved as a similar case to estimate design costs for current projects with the same client, the architect is warned about the nature of the excessive project cost in addition to the higher than usual cost prediction. Therefore, the architect can explain to the client the reason for the higher cost or include a maximum number of change requests in the contract.

The evaluation of large amount of information essentially requires previous experience. Thus, focusing attention in the context of a large information base is challenging and difficult job. CBR provides a technique for capturing the experience in a large set of historical cases. It is a paradigm for knowledge-based systems development that solves new problems by retrieving and

adapting old solutions[5,8]. In addtion to problem solving, its case library can be used for training and additional research into the trends and anomalies in the data.

Furthermore, the case-based approach has intutive appeal over other artificial intelligence(AI) techniques because research in learning has established that, in most situations, it more closely resembles human reasoning. People typically solve problems by remembering and applying past experience. In that sense, CBR is a psychological theory of human cognition[8]. It addresses issues in memory, learning, planning, and problem solving. CBR also provides a foundation for a new technology of intelligent computer systems that can solve problems and adapt to new situations.

CBR has been applied to a wide variety of domains. Early applications focused on classification problems, but more recent work has applied CBR to planning and more sophisticated models for prediction. Successful recent application areas include message routing, computerized help desks, and militaray battle planning.[4,5,8,16] Case-based reasoning makes data more accessible by organizing it as a set of examples from past experience that can be generalized and applied to current problems. It employs a combination of artificial intelligence technologies such as advanced knowledge representation and inductive machine learning to structure the information and discover useful knowledge. CBR also integrates traditional approaches to data analysis into its methodology. The many CBR applications illustrate an

emerging pattern of using CBR to expand and apply corporate experience.

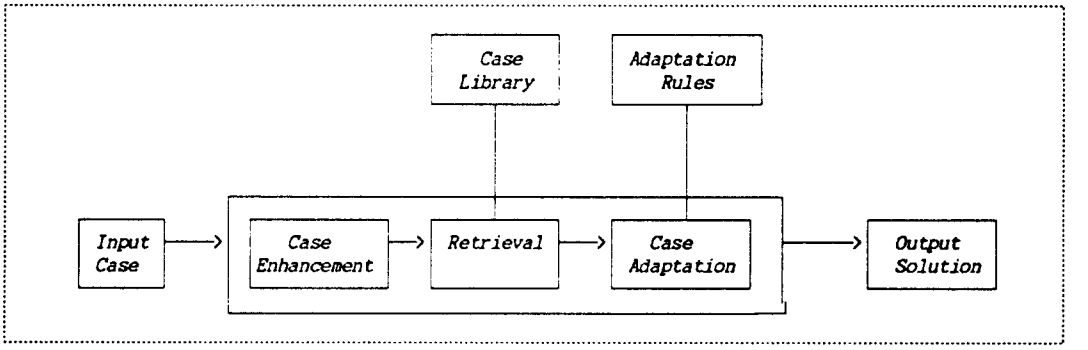
The main objective of the paper is to illustrate basic concepts of CBR and address the opportunity of its application to business fields. It also briefly describes development process of CBR systems, and tries to explore issues such as indexing and similarity assessment with an illustrative application. CBR system discussed in this paper can provide a basis for building more flexible and adaptable expert systems for business applications. Furthermore, a development project of this kind can provide insights into the more complex aspects of management decisions, because this field is one of the least defined and most general in the nature of the domain.

II. Overview of CBR system

Figure 1 illustrates the basic components of a CBR system[4,11,14]. Given an input specification of a problem, embodied in a case to be analyzed, a case-based system will search its memory for an existing case that matches it. If an exact match with the input exists, which is rare, the retrieved case can be used directly to solve the problem. If however, the retrieved case is only similar to the problem, it may not be appropriate to directly apply it as a solution. In this situation, the system or the user must alter the retrieved case to better match the input case. This adaptation process results in a completed solution. When appropriate, new,

unique cases should be added to the case library to expand its knowledge for future retrievals. In general, this process makes case-based systems easier to build and maintain. Once the case representation is defined, adding new cases is the most significant way to expand the system's knowledge.

Figure 1. IPO Concept of CBR System



In case based reasoning, a reasoner remembers previous situations similar to the current one and uses them to help solve the new problem. CBR can mean adapting old solutions to meet new demands; using old cases to explain new situations; using old cases to critique new solutions; or reasoning from precedents to interpret a new situation or create an equitable solution to a new problem[8].

1. Advantages of CBR systems

CBR systems provide various advantages. The most important is that it parallels the way people solve

problems in many situations.[6,8] When faced with a current problems, people tend to remember a previous similar problem and adapt that solution to fit the current circumstances. Remembering previous experiences is particularly useful in warning of the potential for problems that have occured in the past, warning to take actions to avoid repeating past mistakes.

Because CBR parallels the way people think, other benefits are realized. Past similar cases can be presented to users as justification of the current solution.[2,5] Since people use past cases as a basis, this type of justification provides strong intuitive appeal. When given the market value of your house, you intuitively ask the appraiser the selling prices of similar homes. Since making adjustments from past cases is entwined with the way people think, in many domains CBR systems can present the past similar cases to users and let them draw the appropriate conclusions.

CBR systems can improve their performance over time by the inclusion of more cases[4,8,14]. Innovative improvements are naturally and automatically incorporated. A smart travel agent might realize that by booking a pair of tickets to Paris it can appear as if a customer is staying over a Saturday night on both trips when he/she is actually not. This reduces the price of each ticket considerably. This case can be entered into the CBR travel planner, thus improving its ability. The planner can consider this strategy on the next out-of-town trip.

CBR systems handle ill-defined concepts and missing

data well[8]. If the current situation is missing the value of an important feature, that feature is simply not used similar cases will match well with the target case, except for the missing feature. The retrieved cases will possess a variety of values for the missing feature so its influence and importance can be determined. If cases are similar, the missing feature is unimportant and a prediction can be made with confidence. If the outcomes vary widely, the prediction should be delayed until the value of the missing feature is determined for the target case.

One of the important strengths of CBR that sets it apart from most AI techniques is that a CBR system is aware of its own limitations.[15] If no similar cases are retrieved, the CBR system cannot make a judgment or decision. This process is far superior to making a nonsensical judgment as most systems would. Also, a CBR system's performance degrades gracefully as the current situation moves out of its range of experience. Problems well within the case base's range will prompt the retrieval of several very similar cases, allowing accurate predictions and estimates of the prediction accuracy using standard statistical techniques such as the standard deviation. As the current problem moves outside the case range, fewer, less similar cases are retrieved, but the change is not abrupt. The system knows that less accurate predictions are being produced.

CBR systems also gives a means of evaluating solutions when no algorithmic method is available for evaluation.[8,16] Using cases to aid in evaluation is

particularly helpful when there are many unknowns, making any other kind of evaluation impossible or difficult. Instead, solutions are evaluated in the context of previous similar situations. Making decision based on similar past experiences is well-grounded. It starts with an answer known to be correct for similar circumstances. By starting from a similar case, a smaller adjustment can be made than started from scratch. This smaller adjustment is easier and introduces less error.

CBR can focus on important parts of a problem by pointing out what features of a problem are the important ones.[17] What was important in previous situations will tend to be important in new situations. Thus, CBR system can focus on those features that are implicated in a success of previous cases. After adjusting a past case to fit the current situation, the CBR application can show users other information relevant to adjustment, even if it cannot be accounted for automatically. Users can factor this information into the adjustment themselves or use it in assesing the estimate's accuracy. Comment or note fields are good examples of the technique because the English grammer provides great flexibility and understandability. They represent messages from the past from people involved with the old case to the present users solving a current problem.

Finally, the use of CBR entail less effort or cheaper than other AI techniques. The primary reason is that only the form of the knowledge must be designed by a knowledge engineer, not the knowledge itself. In other

words, developers must decide on the structure of the cases, but they need enter each case themselves. Depending on the domain, the cases can be read in from existing databases or even entered by the domain experts themselves. Rules cannot be entered by and automatically from data.

2. Weaknesses of CBR systems

Although CBR systems have many strengths over other techniques, it still has several limitations to overcome. The question that arises is whether to have new case entry happen automatically. Two main barriers exist[7,17]. First, at the time the target case is entered, similar cases retrieved, and an answer developed for the target case, the outcome is often not yet known. If a CBR system helps estimate the design cost of an architectured project, at the time the problem is entered and the system makes the estimate, the actual design cost is unknown. So this new case cannot be completely entered until the project is finished many months later.

Second, many managers are uncomfortable with the idea that users can enter cases and change the software's functioning, and for good reason.[7] The new cases should be checked for adaptation scheme used, even the entry of perfect cases may introduce unwanted statistical biases. CBR is fault tolerant, and these concerns are exaggerated. In any case, target cases are collected and put aside until the actual result is

available. The actual results are added to the case. The case is then reviewed for validity before it becomes part of the case base. Any indices also must be updated. A further refinement is to make the result predicated by the CBR application another feature. This feature can be used or allow adaptation by the users.

The biggest weakness of CBR is its requirement for sufficient cases[8,9]. Enough cases should be present in the case base so that a similar one is retrieved. The sparser the case base, the more effort must be invested into the adaptation strategy. In the extreme, making a decision from a case that is not very similar to the current situation is just as difficlut as making from scratch. It's not so much the absolute number of cases in the case base that is important as the density of cases are required to fill the case base to the required density. In other domains, especially those with many important features, a very large number may be required.

Another weakness to be noted is that CBR systems might be tempted to use cases blindly, relying on past experience without validating it in the new situation.[8] CBR systems might allow cases to bias users or system itself too much in solving a new problem.

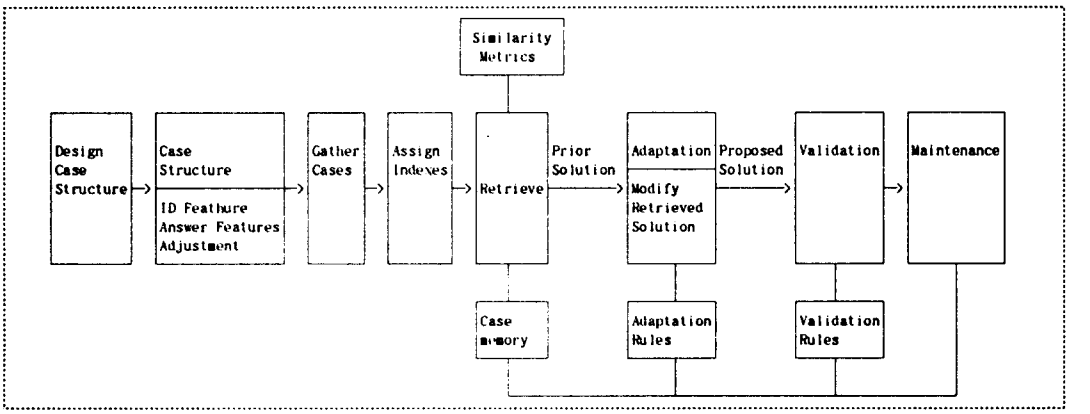
III. CBR Development Methodology

1. CBR Development Process

CBR development process consists of four major components. The major issues in developing a case-based

system: case representation, indexing and retrieval, adaptation and application, and maintenance are illustrated in Figure 2 [7,8,17].

Figure 2. CBR Development Process



The first step of the CBR development process is to design the structure of the case. The second is to collect the cases. The third is to prototype the similarity retrieval. Finally, the prototype undergoses successive refinement, Each step is discussed in more detail, and an illustrative application is given.

1) Case Representation: Design and Collect Case

One of the major advantages of CBR is that it is possible to build and implement a system with a small library of seed cases and allow the knowledge base to be expanded and refined over time. Initial cases can be taken directly from the expert's notebook or past experiences. A case represents a problem's context that are used to determine if a case is similar to a new

problem as well as the correct solution to the problem.

In case representation, data must be organized into well-specified units that form the basis for reasoning. Cases are typically slot-filler or frame-based representations that are commonplace in AI. Conceptually, this idea is simple. For instance, a system designed to approve bank loan applications would consider each application as a different case. However, the best repersentation of the attributes of a case may be difficult to determine. Estimating the appropriate depth of knowledge for an application requires a good understanding of the domain. In a loan application, the possible values for a customer's occupation in a loan application may be related conceptually, in a conceptual hierarchy, doctor and lawyer would be chidlren of "professional" jobs. Simply representing the occupation with a string or a code as is common in databases would omit valuable knowledge that may be useful in case indexing and adaptation. Many CBR systems use sophisticated knowledge representations to represent case knowledge. Finally, nested case structures, where case attributes refer to other cases, may be required for sophisticated problems, such as subtasks in a planning problem.

2) Case Indexing and Retrieval: Similarity Retrieval

Case-based systems derive their power from their ability to retrieve relevant cases from a case library efficiently. How should a large library be indexed so that this retrieval process is most accurate and

appropriately focused? An expert user in a well-understood domain may be able to use a case library just as a databases, issuing standard queries to retrieve specific cases. However, for most applications, more sophisticated approaches are required. Three major types of indexing have been applied, often in combination. They include:[3,8,12,14,18]

(1) Weighed-sum approach: This strategy retrieves cases basesd on a weighed sum of features in the input case. The cases with the "closest" overall match according to some similarity metric are returned from the match process. This approach is best if the retrieval needs are not focused tightly on solving a specific problem, such as finding a "dream house" in a real-estate database or retrieving a list of similar problems from a customer-support database.

(2) Inductive retrieval approaches: These methods are best when the retrieval goal is well-defined, such as the bond-rating problem. Cases are indexed based on the most importatnt features affecting the outcome as induced from the data itself. The resulting decision tree provides for considerably faster retrieval times than weighed-sum approach. Popular induction techniques used for case indexing include variants ID3 and CART modified to utilize more complex knowledge representations[7,17].

(3) Knowledge-based retrieval: This technique applies existing domain knowledge to locate relevant cases. This approach is similar to rule-based expert systems, in which an expert determines the features used

to classify cases. The knowledge need not be complete, and frequently systems combine a partial model of the domain with other indexing methods to retrieve accurate solutions.

Consider the indexing retrieval problem in two phases. First, the relative importance of case attributes must be determined for the current problem. In inductive indexing approaches, this process generally is automatic and based on the data itself. Knowledge-based and weighed sum approach require an expert to determine the relevant features for indexing. Then, the input case must be matched or stores in the case library using these attributes and their specified importance. As noted earlier, the technique applied varies based on the problem being solved and the amount of information available to solve it.

Many types of indexing can be used in similarity retrieval. The indexing must work in concert with the retrieval algorithm and the similarity definition.[18] For most applications, indices can be kept simple. One or two hierarchical decompositions of the case base and hashing on one or two particular features are two examples. In very large case bases, with greater than 10,000 cases, more intricate schemes may be required.

3) Adaptation and Validation

If an exact matching case can't be found, the system presents the closest matching case. The user then describes how he/she wants to try to modify the case. After the user makes a modification, the system tries to

validate the new configuration. Validation is done by comparison with similar valid and invalid cases. If the case is predicted to be valid, the system then proceeds to generate the description of how to apply the case to the current situation. If the system predicts that the case might be incompatible, for instance, because of similarity to an invalid case, it suggests alternative configurations that are similar but valid.[8,16]

Then, how should retrieved cases be applied to the current situation? The answer to this question is application-dependent. For instance, a system designed to rate bonds simply may use the rating on the input bond. Frequently, however, a case library does not contain an exact match for the input case, or a conflict in the solution from the retrieved cases may exist. In these situations, adaptation rules may need to modify the retrieved cases to ensure their relevance to the current case. Adaptation is still an area of active research in CBR, and it is often one of the most difficult problems in developing a case-based system, particularly when data or domain knowledge is limited[5,8,16]. As a case library grows, the need for adaptation frequently diminishes. Consequently, many deployed applications require little or no adaptation.

4) Maintenance

While maintenance can be performed on the library on a case-by-case basis, updates can be done by importing new data and expanding the library with the reclustered new information. The maintenance process is also

application dependent. Important issues for maintenance include: Who should be reponsible for maintaining the case library and how frequently is updating needed? While these questions reflect managerial concerns, they also are vital in domains where the knowledge is dynamic and require frequent maintenance.

Many factors must be considered here regarding the integrity and consistency of the case library over time. Verification and validation are somewhat more straightforward than expert systems, becauase inconsistencies in a library are usually easier to identify. Also the process of generalizing from the dynamic data into applicable rules is performed automatically by the system, further ensuring consistency of results.

2. An Illustrative Application

To design the case structure, the case's relevant features must be discovered. Three kinds of features exist. Descriptive features, typically names, identification numbers, descriptions, and so on, help identify and describe the case. The second set of features represents the solution. In a prediction application, this would be the number to be predicted, such as the sales volume on a given day. The third set of features affects the adjustment. A diffrence in the value of one of these features should lead to a diffrence, even if slight, in the solution. A useful way to discover these adjustment features is to have the

expert compare and contrast diffrent cases. The features they describe as contributing to diffrences in answers are the adjustment features. For example, in a design cost prediction system, the architects might explain that the design cost of one house is more than another because the first has two stories and the second has one. This situation means that Number Of Stories should be one of the adjustment features will be used for similarity retrieval and adjustment of the retrieved case. Adjustment features also include comment-type fields displayed to users, who may further adjust the solution themselves based on the comment content.

Once the set of features that define a case has been decided, the cases themselves can be collected. This entry of the knowledge is the easiest step in CBR application development. Much of the data may reside in electronic form, which must be translated into the appropriate format. Often, some protion of case information only resides on paper or within domain experts. This information can be entered by clerical staff or the domain experts, since it usally consists of filling in simple values for selected features of the cases. For example, in a manufactured parts application, one feature not in an existing data could be Shape. The general shape of a part could be classified as Round, Complex, Vane, Strut, and so on. Shape, size, complexity, and material describe the physical attributes of a part. Assembly lines can be described by a numerical identifier as well as a brief description.

During entry of case information, some care should

be taken not only to avoid introducing new data errors but also to find errors in existing data.[4,5,16] One way to reveal errors is to use each case in the case base as a target for similarity retrieval. If the target is diffrent from the cases to which it is supposedly similar, it may contain errors. Alternatively, the definition of similarity used for retrieval may need adjustment.

Prototyping the similarity retireval is the most important step in a CBR application. Two separate, related portions exist: the definition of similarity and the retrieval algorithm that uses it. Keeping them conceptually seperate and explicit is important. Several types of similarity definitions and their accompanying retrieval algorithms exist. Many CBR researchers prefer similarity based on a weighted sum of the feature matches.[5,7,8,17,18] The feature matching is performed by functions returning a number representing the degree of matching, zero to one. These returned values are combined with a weighted sum and divided by the maximum possible, the sum of the weights, for a similarity score from zero to 100%. During this stage, the Rretireval algorithm examines every case to compute its similarity score. More efficient methods can be used later. Maintaining the freedom to change the matching functions arbitrarily and dynamically adjust the weights is important. Stottler[17,18] recommend the CBR tool Esteem, by Esteem Software Inc., for this prototyping stage.

An illustration of a simple example is given to

demonstrate the similarity assessment technique. Table 1 shows values of three attributes in hypothetical target case, the turbo engine, and two previous cases. The assembly lines feature is a list of values. Table 2 demonstrate the weights that were decided to be used in the assessment of similarity. It also shows how similarity is computed. Each attribute is compared to the corresponding attribute of the target case, then resulted in a matching score from zero to one.

Table 1. Values for 3 Attributes for the Turbo Engine

	Assembly lines	Shape	Part description
Target case	1132	Round	Turbo engine
Previous case 1	0123 1296 1132 4501	Complex	High performance fuel-injected engine
Previous case 2	8031 8431 4163 1184 0123	Round	Compressor

Table 2. Weighted To Be Used In The Similarity Assessment

	Weight	Previous case 1	Previous case 2
Assembly lines	10	1.0	0.6
Shape	10	0.0	1.0
Part description	5	1.0	0.0
Weighted sum		15	16
% of maximum		60	64

Previous case 1 is given a matching score of zero for 'shape' because 'shape' must match exactly 'complex' isn't quite the same as 'round'. On the other hand, previous case 2 received a matching score of one since its attribute matches exactly. Previous case 1 received a matching score of one for part description because 'turbo engine' is a subclass of 'high performance fuel-injected engine'. Previous case 2 receives a matching score of zero for part description because 'turbo engine' is not a subclass of 'compressor'.

The comparison process for assembly lines is more complicated because its attribute is a list of values. Previous case 1 gets amatching score of one for assembly lines since its best match from the list, 1132, matches exactly. Previous case 2's best match is 1184, which matches on the first two digits. It receives a matching score of 0.6.

Next, a weighted sum is calculated for each previous case by summing the product of the weight and score for each attribute. For previous case 2: 10*0.6+10*1.0+5*0.0=16. This weighted sum is divided by the maximum possible weighted sum, in this case, 25. The normalized similarity can then be computed with a simple division. For previous case 2, 16/25=0.64. These cases would be presented to the users for examination.

Sometimes the weighed sum approach is referred to as a nearest-neighbor search, although it is a variant, depending on whether the weights change and the complexity of the matching functions. Many times a matching function will consider combinations of features

or use derived features. For example, two building site locations would be similar if their latitudes and longitudes were close in value, but not if either was close separately. Derived features for the target case must be calculated at the time it is entered, but for cases in the case base this calculation should be done in advance.

Prototyping the similarity retireval is an iterative process, requiring successive refinements. The similarity definition should be validated by performing retrievals with actual current cases of problems, showing domain experts the results and adjusting the similarity definition according to their feedback.[2,6,13,17] Depending on how the retrieved cases will be used, the beauty of CBR is that the retrieval doesn't have to be perfect. The most similar case doesn't always have to be at the top of the list, just in the first three to five. If, in the final application, knowledgeable users select the retrieved cases to adjust, some nonsimilar cases are acceptable in the retrieved list as well.

In developing a definition of similarity, keep in mind the purpose for which the retrieved cases will be used. Diffrent applications of the cases will be used. Diffrent applications of the case base may require diffrent definitions of similarity. The retrieved cases must be adjusted to fit the current situation more correctly. This adjustment will be based on differences in feature values among the retrieved and target cases. Retrieval is based on feature similarity; adaptation is

based on feature differences. This adjustment is more difficult for the differences in some features than in others. The more difficult features should be given more importance(weight) in the similarity definition. This strategy causes cases to be retrieved with smaller differences in these difficult features, thus avoiding or minimizing the problematic adjustment.

Once the similarity prototyping is complete, by adding a target entry form and presentation of retrieved cases capability, a rudimentary CBR prototype is created. Domain experts can enter their current problem(target case), retrieve and examine similar cases, select one or more as as basis, and make adjustment process closely to gain knowledge for the next stage. The final step is to refine the rudimentary prototype developed in the previous three steps(design case structure, gather additional cases, and prototype similarity definition), other considerations are important at this stage: automatic cases adjustment, case indexing and new case entry.

In CBR, the adjustment problem can be tackled with AI and statistical techniques[4,8,10,18]. This adjustment problem can be considered another easier AI problem. The use of a previous case gives us a much closer starting point than developing the solution from scratch and bounds the resulting error estimate much more tightly as well. Rules are the most obvious adjustment technology. Fuzzy logic, neural networks, and object-oriented programming, when cases are represented as object, should be considered alone or in combination.

One useful technique is to apply the adaptation to several retrieved cases and average the results for a more accurate estimate. A standard deviation of the results gives a rough idea of the probable error bounds as well. During adjustment, filtering bad or unusual data is important.

Any CBR systems involving more than a couple hundred cases requires a more efficient method of retrieval than comparing every case to the target. Indexing and multistage retrieval should be investigated separately or in combination. In multistage retrieval, a set of cases containing the most similar ones is retrieved through some efficient means. This set is much smaller than the original case base. Each case in the set is then examined. The efficient retrieval method might involve database queries or some indexing method. Typically, the first-stage retrieval takes advantage of some aspect of the domain or similarity definition. One or two features might be so important that all similar cases must have values for those features that are close to the corresponding values of the target case.

IV.Summary and Conclusion

CBR is a strong AI technique that is applicable to make rule based expert system more flexible and intelligent. It offers several benefits over other techniques such as less development efforts, similarity to human thought processes, justifications, automatic performance improvement, awareness of knowledge

boundaries, applicability where other techniques fail, incomplete domain theory, and rich representation schemes. In addition, it combines well with other techniques and can easily be added as an additional capability[6,8,11].

Neural networks currently are very popular for anayzing large databases. However, they suffer from many of the same inadequacies as the statistical approaches. They generally are limited in their ability to work with non-numeric data, and they can't explain their results without substantially more development effort. In the training process, there is no clear way to know what a neural network has learned. This information is only implicit in the network's performance on a specified test set. Additionally, the network builder must devote considerable effort to designing the network's structure, such as the number of nodes and layers and their connectivity.

Rule-based expert systems have been applied successfully to many domains with large amounts of data. They provide a high level language with which to represent knowledge. However, the process of acquiring that knowledge has been identified clearly as the bottleneck in the developing these systems. Generalizing rules from a large database is a difficult experts. Once developed, rule-based expert systems can perform many useful tasks, but developing and maintaining them frequently can eliminate their cost-effectiveness. Automated systems for inducing rules from data suffer from the inherent limitation in using rules as a basis

for representation. While rules are good for situations where the decisions are clear-cut, such as "Don't give a loan to someone without an income," they frequently break down when subtle interacting factors must be considered in reaching a decision.[4,8,17]

CBR offers a number of benefits over these conventional analysis technique. These benefits arise primarily from CBR's integration of useful aspects from each. The CBR methodology can integrate advanced techniques for frame-based and rule-based knowledge representation with statisticial tools for analyzing data. Cases are the foundation for knowledge, not just for inducing rules, but also in explaining and enhancing a system's behavior. This approach provides for a combination of features and continuity that unavailable from other technique. In sunmmary, the case-based methodology for developing intelligent expert systems provides the following important features:[5,8,11,15]

First, case-based methodology has the ability to explain results based on the criteria for indexing the cases, relevant cases from the case library, and optional information from a domain mode. Second, incomplete data results in an easily understood degradation in the performance of the system. Case retrievals based on an incomplete input case produce broader results that can let the access whatever knowledge might be applicable in the case library in reaching a decision. Third, case-based systems are maintained and expanded more easily. If performance is inadequte for a set of cases, these cases can be added to

case library. This additional knowledge is generalized and applied to future cases.

Finally, case-based systems provide a parpadigm for interacting with expert systems that are useful for both novices and expert. Novices can use a case library to learn more about the domain and the relevant genenalizations that can be applied in different situations. Experts can use the knowledge in the case library not only to automate simple decisions, but also as a foundation for the subjective comparision of similar cases whose outcome is not understood easily. Given a new case, a well-designed case-based system can present the relevant cases in given situation, explain why they are relevant and describe how they may applicable.

References

1. Barr, A., and E. A. Feigenbaum. The Handbook of Artificial Intelligence, Vol. I and II, Addison Wesley Publishing Co., 1981
2. J. Carbonell. Learning by Analogy: Formulating and Generalizing Plans from Past Experience In Machine Learning, Palo Alto, CA. Tioga Press, 1983.
3. J. Carbonell. Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition in Machine Learning, Vol. 1, Morgan Kaufmann, 1988.
4. A.R.Golding & P.S.Rosenbloom. Impoving Rule-Based Systems through Case-Based Reasoning. In Proc.of

IJCAI-89, Detroit, 1989.

5. David Hinkle & C. Toomey. "Applying Case-Based Reasoning to Manufacturing," AI Magazine, Spring 1995, pp. 65-73.
6. K. Hammond. Case-Based Planning: Viewing Planning as a Memory Task. Academic Press:New York, 1989.
7. Johnson,P. "A Cas-Based Reasoning Paradigm for Mining Financial Database", in Proceedings of the Second International Conference on Artificial Intelligence Applications on Wall Street. R.Freedman,ed. Gaithersburg,M.D. : Software Engineering Press, 1993.
8. J.Kolodner. "An Introduction to Case-Based Reasoning," A.I. Review, Vol.6, No.1, 1992. pp. 3-34.
9. J.Kolodner & C.Riesbeck. Case-Based Reasoning Tutorial. IJCAi-89, Detroit, Michigan, 1989.
10. Madhavan, R.K., "Goal-Based Reasoning for Securities Analysis", AI Expert, February, 1994.
11. C.Riesbeck & R. Schank. Inside Case-Based Reasoning, Lawrence Erlbaum Assoc., Hillsdale,NJ, 1989.
12. E.Rissland & K. Ashley. HYPO : A Case-Based Reasoning System.
    Proceedings. IJCAI-87, Milan, Italy. 1987.
13. H. Shinn. A Unified Appoach to Analogical Reasoning. Ph.D. Thesis, Georgia Institute of Technology, Atlanta, GA, 1989.
14. H. Shinn. A Case-Based Approach to Means-Ends Analysis. In Proceedings of PRICAI-92. 1992.

15. S. Slade. Case-Based Reasoning: A Reasearch Paradigm. AI Magazine, Vol. 12, No. 1, Spring 1991. pp. 42-55.

16. Robert A. Small, & Bryan Yoshimoto. "The VLS Tech-Assist Expert System," AI Magazine, Spring 1995, pp. 41-50.

17. Stottler, R. H. "CBR for Bid Preparation", AI Expert, Vol. 7, No. 3, March 1992.

18. Stottler, R. H., A. L. Henke and J. A. King, "Rapid Retrieval Algorithms for CBR:, in Proceedings of the I.J.C. on Artificial Intelligence, 1989.