International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 6, ISSN No. 2455-2143, Pages 225-229
Published Online October 2019 in IJEAST (http://www.ijeast.com)

EXPERT SYSTEM USING CASE-BASED REASONING TECHNIQUE TO IDENTIFY CLASS OF AN OBJECT BELONGS TO VARIOUS DOMAINS OF KNOWLEDGE BASED SYSTEM

Nihar Ranjan Hota,
Associate Professor,
Department of CSE
EATM, Bhubaneswar, Odisha, India

Sushree Sanibigrah,
Research Scholar,
Department of CSE
EATM, Bhubaneswar, Odisha, India

ABSTRACT - On the building up of a general Expert System utilizing Case-Based Reasoning techniques which is able to predict the class information for various sets of data and finding the efficacy of the system by comparing it with classifier models built on the basis of Decision Tree Induction and Statistical principles. The aim of this thesis work has been to build up an Expert System which can identify the class of an object belonging to various types of data domain, initially with the help of a domain expert. The class information along with the data describing the object form an *instance* or *case*, which is added as such in a knowledge base, and helps to increase the expertise threshold of the system. Once the threshold attains a predefined level, the system continues to add new knowledge (if it indeed is new when compared with the existing repository) or discard the same if it matches any part of the stored knowledge exactly. The comparison criteria and the addition of knowledge required much research attention the performance of the system has been compared with that of two standard classifiers: viz. Decision Tree based Classifier and Naïve Bayesian Classifier. In the process of Classification and the basics of Case-Based Reasoning (CBR) technique which is used for building up the Knowledge base System. Finally researcher focus is to implement such expert system using case based reasoning technique to identify class of an object belongs to various domains.

KEYWORDS: Expert System (ES), Case-Based Reasoning (CBR), Decision Tree based Classifier (DTBC), Naive Bayesian Classifier (NBC).

I. INTRODUCTION

The aim of this thesis work has been to build up an Expert System which can identify the class of an object belonging to various types of data domain, initially with the help of a domain expert. The class information along with the data tuple describing the object form an *instance* or *case*, which is added as such in a knowledge base, and helps to increase the expertise threshold of the system. Once the threshold attains a predefined level, the system continues to add new knowledge (if it indeed is new when compared with the existing repository) or discard the same if it matches any part of the stored knowledge exactly. The comparison criteria and the addition of knowledge required much research which has been described in some detail in the subsequent chapters. The performance of the system has been compared with that of two standard classifiers: viz. Decision Tree based Classifier and Naïve Bayesian Classifier. The rest of the current chapter is devoted to brief discussions regarding the underlying theory of an Expert System, the process of Classification and the basics of Case-Based Reasoning (CBR) technique which is used for building up the Knowledge base.

A. *Expert System*

An Expert System may variously be defined as follows-

- An expert system is an application in the domain of Artificial Intelligence (AI) that uses a knowledge base of human expertise to aid in solving problems. The degree of success achieved in solving the problem is based on the quality of the data and rules obtained from the human expert. A well designed Expert systems aims to perform at a rate comparable to a human expert.

- An expert system is an interactive computer-based decision tool that uses both facts and heuristics to solve decision making problems, based on knowledge acquired from an expert.
- An expert system is a model and associated procedure that exhibits, within a specific domain, a degree of expertise in problem solving that is comparable to that of a human expert.

1.2 Basics of Case Based Reasoning

In a nutshell, as the name implies, CBR is Reasoning Based on Cases. A problem with its solution (problem-solution pair) is called a Case. Case-Based Reasoning is the act of developing solutions to unsolved problems based on pre-existing solutions of a similar nature. This is analogous to being presented with a problem that we have to solve. While tackling the current problem, we start to formulate a solution, by reviewing our experience of other problems and determine to what extent they relate to the current problem. If a previous problem (of a problem/solution pair recorded in our memory) is fairly close to the current problem, then the solution to the previous problem is applied to the current problem. As the current problem and previous solution are tested for functional compatibility and operational feasibility, the problem solver is actually determining how well the retrieved case matches the current needs.

II. TYPES OF CLASSIFIERS

A Classifier is a tool which has the ability to assign a class value to each data tuple presented to it once it has learnt how to do so. The process of learning is known as training or model building. Thus a Classifier may also be designated as a Learner. Here follows a discussion on two types of Classifiers: the *Eager Learners* and the *Lazy Learners*.

2.1 Eager Learners

These are classifiers built on the basis of a set of data for which class information is already provided. Since this data set is used to train up the generalized model for accurately classifying any unseen or new data tuple, this data set is known as the set of training tuples. Once trained up, the performance of the classifier needs to be tested with a fresh set of data consisting of the test tuples. These are also provided with class information, but the class information is used for comparison against the class generated by the classifier. The rating of the classifier depends largely on the percentage of matching between the two class information. A high-performance Classifier will now be *eager* and ready to classify previously unseen tuples. Hence the name *Eager Learner*.

Decision Tree and ***Naïve Bayesian*** are the Eager Learner techniques explored in the present context, mainly to compare their performance against that of the Expert System. The next two subsections elaborate their working principle.

2.1.1 Decision Tree

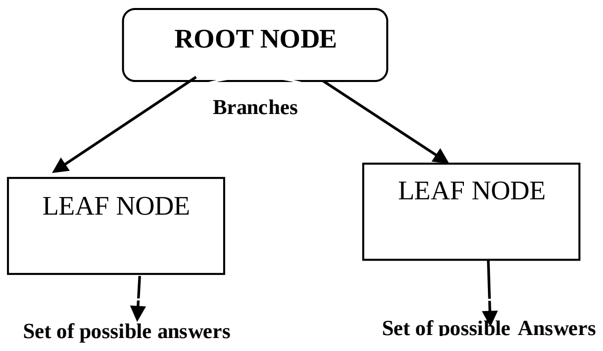
The goal is to create a model that predicts the value of a target variable based on several input variables. Decision tree learning, used in statistics, data mining and machine learning, uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. More descriptive names for such tree models are classification trees or regression trees. In these tree structures, leaves are identified by class labels and branches are identified by conjunctions of feature / attribute values that lead to those class labels.

A decision tree is an arrangement of tests that provides an appropriate classification at every step in an analysis. "In general, decision trees represent a disjunction of conjunctions of constraints on the attribute-values of instances. Each path from the tree root to a leaf corresponds to a conjunction of attribute tests, and the tree itself to a disjunction of these conjunctions". More specifically, decision trees classify instances by sorting them down the tree from the root node to some leaf node, which provides the classification of the instance. Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute.

An instance is classified by starting at the root node of the decision tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute. This process is then repeated at the node on this branch and so on until a leaf node is reached.

In a Decision Tree continuous (real-valued) features can be handled by allowing nodes to split a real valued feature into two
ranges based on a threshold (e.g. length < 3 and length ≥ 3). Algorithms for finding consistent trees are efficient for processing large amounts of training data for data mining tasks. Methods may be developed for handling noisy training data (both class and feature noise).

A simple diagram of a Decision tree is given below:-



(Figure 2.1 A simple Diagram of Decision Tree)

2.2 Lazy Learner

This type of learner waits until the last minute before doing any model construction in order to classify a given test tuple. That is when given a training tuple a lazy learner simply stores it(or does a little minor processing) and waits until it is given a test tuple. Thus, a Lazy Learner takes less time in training but more time at the time of classification.

Although computationally expensive, it has a natural advantage – it supports incremental learning. This is a learning mechanism adopted by human beings. Since all computerized learning falls under the realm of Artificial Intelligence and AI techniques mimic human activity as far as possible, it is perfectly suited to those classification problems where human intelligence plays a vital role. In computer parlance, it is able to model complex decision spaces having hyperpolygonal shapes that may not be as easily describable by other learning algorithms.

Two most popular examples of Lazy Learning paradigm are K-Nearest-Neighbour (KNN) technique and Case-Based-Reasoning (CBR). After touching lightly on the KNN classifier in the next subsection, a detailed description of CBR classification mechanism follows on the subsequent subsection.

2.2.1 KNN Classifier

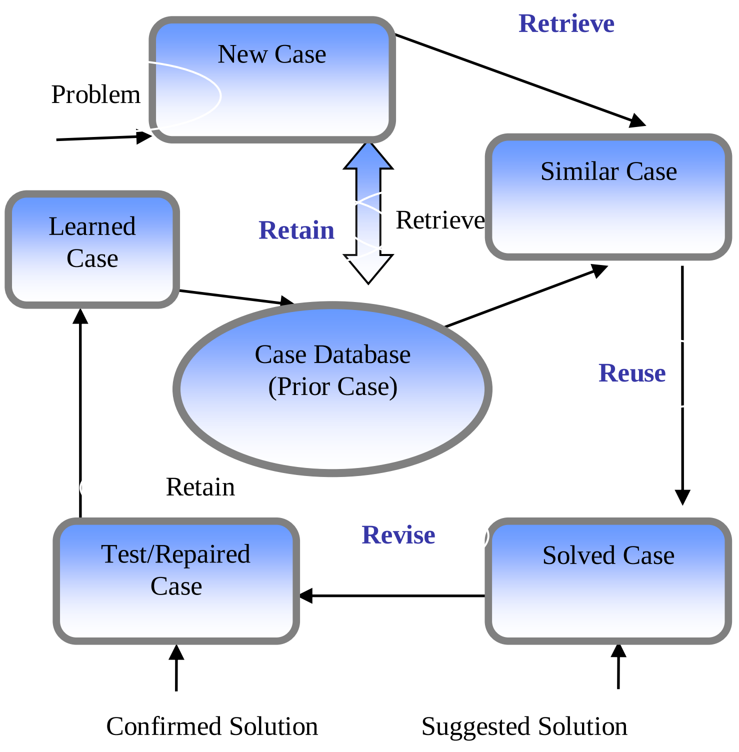
KNN is a simple Lazy learning algorithm that stores all available cases and classifies new cases based on a similarity measure. As the name suggests, whenever a new object of unknown class is presented to this type of classifier, it starts to search for K of the objects nearest neighbors and then learn the most common class label amongst the k class labels and assign it to the new object. The objects are stored as a point in n-dimensional space, each dimension corresponding to an attribute describing the object. Similarity can be measured in terms of some distance metric, such as Euclidean or Manhattan, between the new object point and its neighbor. The value of K is usually determined experimentally.

Its very simplicity renders KNN inadequate in finding solutions to problems with complex symbolic descriptions. Since most real-life problems have this innate property, such problem-solution pairs are stored as “cases” or *instances* in a problem database designated as a ‘case-base’ for a CBR Classifier. But unlike KNN, similarity with neighbors cannot be measured with distance metrics using CBR techniques, as explained in the next subsection.

2.2.2 CBR Classifier

Roots of CBR are found in the works of Roger Shank on dynamic memory. The first system that might be called a case-based reasoner was the CYRUS system, developed by Janet Kolodner at Yale University (Schank's group). CYRUS was based on Schank's dynamic memory model and MOP theory of problem solving and learning. It was basically a question-answering system with knowledge of the various travels and meetings of former US Secretary of State Cyrus Vance. The case memory model developed for this system has later served as basis for several other case-based reasoning systems. Another basis for CBR, and another set of models, was developed by Bruce Porter and his group at the University of Texas, Austin. They initially addressed the machine learning problem of concept learning for classification tasks. This lead to the development of the PROTOS system, which emphasized on integrating general domain knowledge and specific case knowledge into a unified representation structure. The combination of cases with general domain knowledge was pushed further in GREBE, an application in the domain of law. Another early significant contribution to CBR was the work by Edwina Rissland and her group at the University of Massachusetts, Amhearst. Currently, the CBR activities in the United States as well as in Europe are spreading out. Germany seems to have taken a leading position in terms of number of active researchers, and several groups of significant size and activity level have been established recently. From Japan and other Asian countries, there are also activity points. In Japan, the interest is to a large extent focused towards the parallel computation approach to CBR.

Despite the many different appearances of CBR systems, the essentials of CBR are captured in a surprisingly simple and uniform process model. It is known by the name of the CBR cycle as proposed by Aamodt and Plaza. The CBR cycle consists of 4 sequential steps around the knowledge of the CBR system as shown figure below.



(Figure 2.1 CBR CYCLE)

III. METHODOLOGY FOR DEVELOPING THE EXPERT SYSTEM

3.1 Algorithm

The Expert System has been generated on the principles of Case-Based Reasoning, as already mentioned. The algorithmic representation of the logic followed is given below:

```
function ExpertSystem
Input: (R: a set of non-target attributes,
        C: the target attribute,
        S: a data set
        P: set of prime attributes);
begin
If S is empty, return a value with Failure;
Read the threshold value from the user;
For data tuples within threshold limit,
Read entire tuple and generate its binary key value,
Take expert's advice for the class value;
Assign the tuple to a position generated by a hash function;
In case of collision, compare the present tuple with the tuple at that position;
If match occurs then tuple is already present in the hash table;
Else generate new positions to assign the tuple,
    for every new position generated,
        Check if the position is empty,
        If empty then assign the tuple to that position,
        else
            compare it with the tuple present at that position
For each tuple above threshold limit,
Read entire tuple and generate its binary key value;
Then assign the tuple to a position generated by a hash function;
In case of collision, compare the present tuple with the tuple at that position;
If match occurs then tuple is already present and hence discarded.
Else generate new positions to assign the tuple,
    for every new position generated,
        Check if the position is empty,
        If empty then assign the tuple to that position,
        else
            compare it with the tuple present at that position
For the key value generated, select class value using majority voting and P.
End
```

IV. PERFORMANCE ANALYSIS

4.1 Performance Tables

For Decision Tree Algorithm

Data Set	NO. OF TUPLES/ ATTRIBUTES	DECISION TREE	
		PERFORMANCE	DURATION OF RUN
		(Accuracy in %)	(Time in Sec.)
1.Tic Tac Toe	958/10	85.8%	2.0000
2. Car evaluation	1728/7	70.1%	3.0000
3. Nursery	12960/9	60.2%	5.0000

For Naïve Bayesian Algorithm

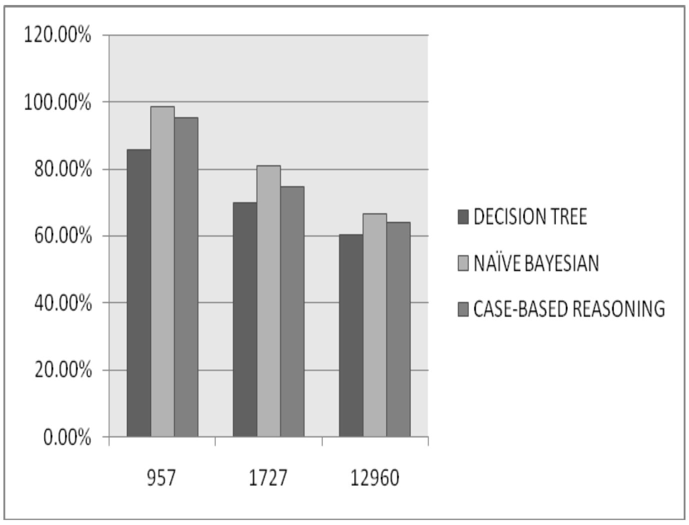
Data Set	NO. OF TUPLES/ ATTRIBUTES	NAÏVE BAYESIAN	
		PERFORMANCE	DURATION OF RUN
		(Accuracy in %)	(Time in Sec.)
1.Tic Tac Toe	958/10	98.7%	1.0000
2. Car evaluation	1728/7	80.9%	1.0000

3. Nursery	12960/9	66.78%	3.0000

For Case-Based Reasoning Algorithm

Data Set	NO. OF TUPLES/ ATTRIBUTES	CASE-BASED REASONING	
		PERFORMANCE	DURATION OF RUN
		(Accuracy in %)	(Time in Sec.)
1.Tic Tac Toe	958/10	95.4%	12420.0000
2. Car evaluation	1728/7	74.6%	5.0000
3. Nursery	12960/9	64.2%	18700.00

4.2 Graph



4.3 Time complexity

1. Naïve Bayesian = O(n*p) where n=number of tuple and p = number of attributes
2. Decision Tree = O(p*n*log|n|) where p=number of attributes and n=Number of tuples
3. Case Based Reasoning = O(n^2*b) where n=Number of tuples and b=size of the bit string formed.

V. CONCLUSION AND FUTURE SCOPE

Designing an Expert System has got its advantages as well as an equal share of disheartening drawbacks as the above Experimental Run Results reflect. At first glance the other two classifiers seem to steal away the shields so far as run-time rates are concerned. But performance-wise accuracy percentage is more than the DT induction method. Although Naïve Bayesian wins the race both ways, we have to keep in mind the limitation of the Class conditional independence approximation utilized by this technique. Besides, the Naïve Bayesian will definitely fail in case of most complex data types. In case of real life data which are obtained piecemeal and contain both categorical as well as continuous type variations, even the DT model may not be produced properly due to inadequate training phase.

It has been hinted in most of the related research papers and texts that CBR benefits to a large extent by processing on parallel systems. Although it has not been possible to test out the efficacy of such claims in the present work, it is definitely an area which needs to be explored.

The memory requirement is another area where there is scope for improvement. Although there is no need to store a model, the indexing technique needed for direct access within the hash table may consume extra time and space. The scope of associative memory mapping may play a vital role in attaining this end economically.

So, we conclude that successful Expert Systems may be safely built on the principles of Case-based Reasoning, provided some parallel hardware is made available to enhance processing speed and memory.

VI. REFERENCES

1. Jiawei Han and Micheline Kamber, “Data Mining: Concepts and Techniques”, 2nd ed, Morgan Kauffman Publishers, 2006.
2. Tom M. Mitchel, Carne Mellon University, Machine learning, The McGraw-Hill Publications.
3. JL Koldoner(1993),chapters 1-3 of Case Based Reasoning, Morgan Kaufmann publishers, San Mateo, CA
4. http://archive.ics.uci.edu/ml/machine-learning-databases/nursery/nursery.names, June,1997
5. http://archive.ics.uci.edu/ml/machine-learning-databases/nursery/nursery.data ,June,1997
6. http://archive.ics.uci.edu/ml/machine-learning-databases/car/car.names ,June,1997
7. http://archive.ics.uci.edu/ml/machine-learning-databases/car/car.data ,June, 1997
8. http://archive.ics.uci.edu/ml/machine-learning-databases/tic-tac-toe/tic-tac-toe.names, Aug, 19,1991
9. http://archive.ics.uci.edu/ml/machine-learning-databases/tic-tac-toe/tic-tac-toe.data, Aug, 19,1991
10. http://dms.irb.hr/tutorial/tut_dtrees.php

Web address http//www.ai-cbr.org/classroom/cbr review.html