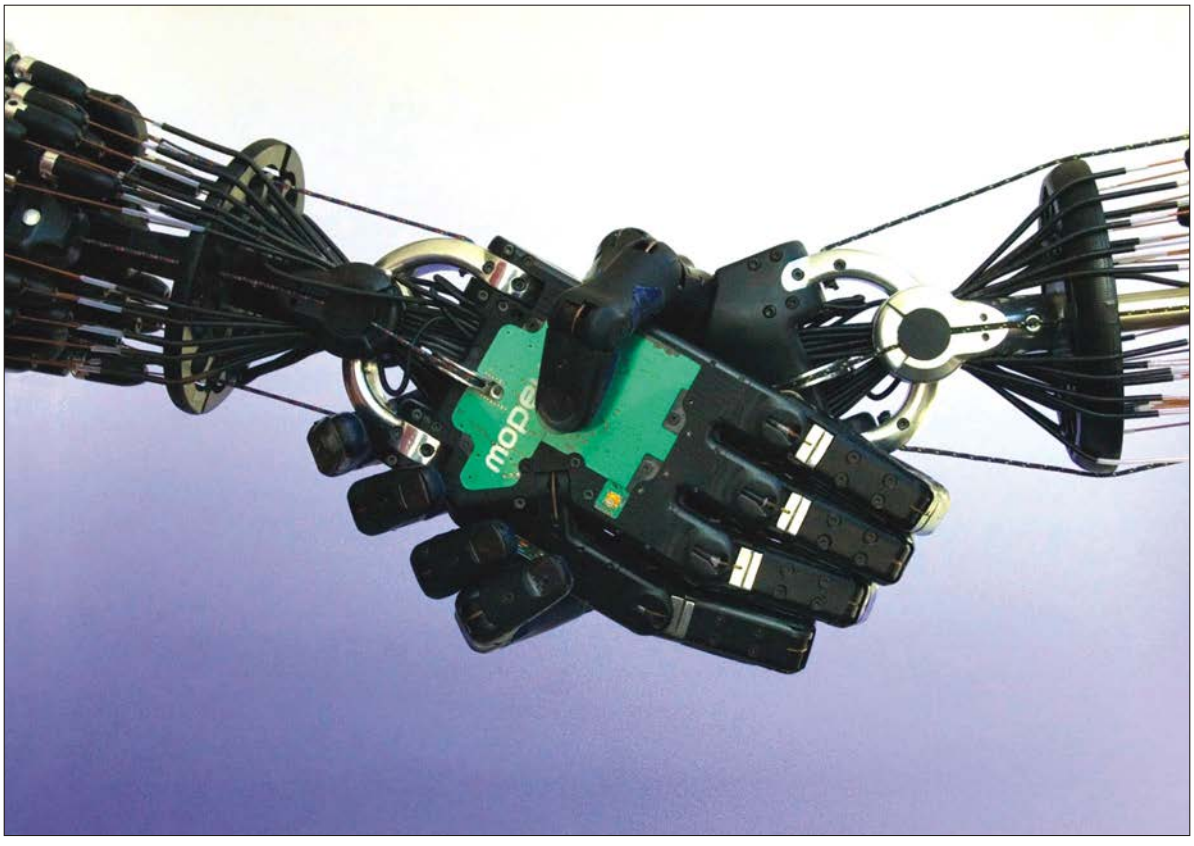

Journal of Artificial Intelligence

ISSN 1994-5450

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Artificial Intelligence 2 (2): 73-80, 2009
ISSN 1994-5450

A Hybrid Architecture for a Decision Making System

Nabil M. Hewahi
Department of Computer Science, Islamic University of Gaza, Palestine

Abstract: In this study, we propose a theoretical general architecture for a decision making system. This architecture is domain independent and based on three main operations. The first operation is obtaining the decision tree from the given datasets. The second operation is injecting the extracted rules from the decision tree into a neural network using an algorithm which we call semi-KBANN algorithm and the final operation is using a heuristic function technique to help in making the final decision of a certain situation. The architecture mechanism is based on feeding each stage operation from the output of the previous stage. The operations are executed serially, stage 1, stage 2 and then stage 3. In stage 1, we extract classification rules from the induced decision trees. In the second stage, we incorporate the extracted rules into a neural network to be able to have more generalization that might be tied by decision trees and finally we use heuristic mechanism to help the system to take a proper decision. We assume that the dataset is cleaned from the outliers using any of the many well known algorithms. Outliers is beyond our study.

Key words: Decision tree, neural networks, heuristic function, KBANN algorithm

INTRODUCTION

Decision making and support systems are very useful in simple as well as complex systems to help decision makers to take the optimum decision. The more is the complexity and sensitivity of the system, the more is the complexed techniques to be used. Decision systems are used in systems such as management of organizational operations, engineering, medicine, investment portfolios, industrial process and control, military operations, business, marketing and many others. There is a substantial amount of empirical evidence that human intuitive judgment and decision making can be far from optimal and it deteriorates even further with complexity and stress. In decision support systems, many techniques are used from statistics, operation research and economics to help in getting rational decision, but due to the developments in systems and their complexities such techniques are not so helpful. Currently, many new techniques inspired from artificial intelligence and cognitive science are used (Druzdzel and Flynn, 2002). The used technique for a certain system is usually based on the decision to be taken. One of the main problems is that each technique to be used is having unique characteristics, but it may lead to some drawbacks. Recently in the market, there are various tools in the market to help the customers in taking their decisions. The customer should be aware to choose the appropriate tool that serve his/her target. Many important decisions routinely made are dynamic in nature, which require number of decisions instead of one decision. Those decisions are independent, which means that the environment concerned with the system could be change. In this case the decision is changing based on the changes happening in the environment. Some examples of dynamic decision making are those that can be considered under computer simulation-based Interactive Learning Environment (ILE). Some of the ILE systems are microworlds, management flight simulator, teaching and tutoring systems. Several decision making systems are listed categorized based on the number of decisions to be produced by each system (Karakul and Qudrat-Ullah, 2007). In this study, we propose a general architecture that can be used in several decision making systems to get more accurate

results. Many decision making systems use only one approach to take a decision. Some of those approaches are decision trees, neural networks, genetic algorithms and fuzzy logic. Other systems use hybrid approaches, but most of those systems are domain dependent. We consider our architecture as domain independent because it uses many approaches in one, which means if one approach is not adequate for a certain domain, others will help. Moreover, our general proposed structure can be good in many dynamic environment decision systems.

In our proposed system, we propose a hybrid system that can be domain independent and uses three main approaches together. The system architecture tries to get advantage of each approach alone and the output of a certain approach is an input for the next approach. The produced results from each stage (approach) is expressed and incorporated somehow in the final decision of the system.

THE PROPOSED SYSTEM

As shown in Fig. 1, our proposed decision making system architecture consists of three consecutive approaches (stages), decision trees, Artificial Neural Networks (ANN) and heuristic function. We shall at the beginning define each of the three approaches. A decision tree is being developed based on the given dataset. Figure 2 shows a simple decision tree. A decision tree is a flow-chart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test and leaf nodes represent classes or class distributions. The top most in a tree is the root node. Decision trees are mainly used for classification and one of its advantages is that we can induce classification rules out of it. There are many algorithms used for inducing classification rules from decision trees. We shall state some of those algorithms in stage I section. Neural networks is a mathematical model or computational model that tries to simulate the structure and/or functional aspects of biological neural networks. It consists of an interconnected group of artificial neurons and

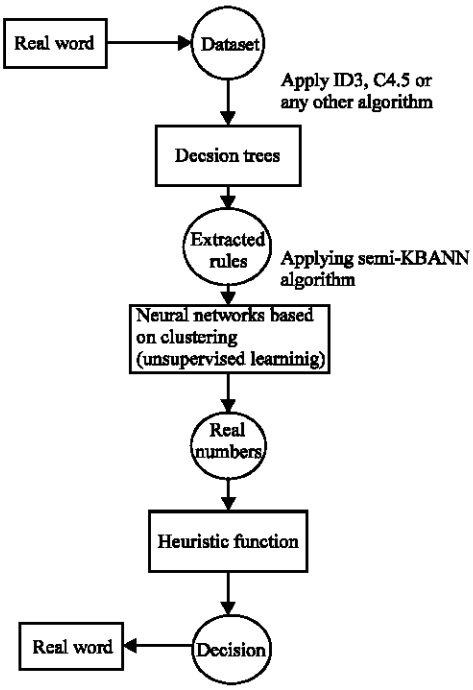

Fig. 1: The proposed general architecture for a decision making system

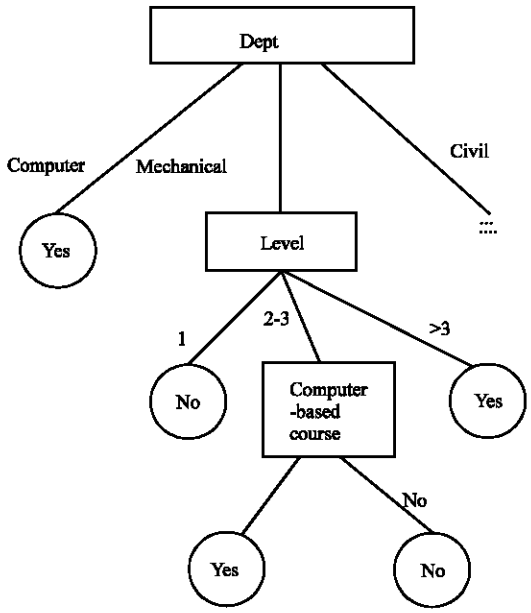

Fig. 2: Decision Tree for the class using computer in the study (Yes/No)

processes information using a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. In more practical terms neural networks are non-linear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. Usually NN contains three main layers, input layer, hidden layers and output layer. There are two main types of neural networks, either based on supervised learning or based on unsupervised learning. In supervised learning inputs and outputs are given and the system is trained by examples. Supervised learning is commonly used for classification. In unsupervised learning inputs are only given and the system is trained only based on the inputs. Unsupervised leaning is commonly used for clustering. Heuristic function is a function that ranks alternatives in various search algorithms at each branching step based on the available information in order to make a decision about which branch to follow during a search. A heuristic function is used to rank the alternatives in each step according to available information. Such a function is usually based on experience and intuition.

In our proposed system the input of the decision tree is the dataset and its output is the extracted classification rules. The input of neural networks is the extracted classification rules and the output of this stage is real numbers between 0 and 1 representing the clustering. In the final stage, the obtained numbers are entered into heuristic function and a final decision is given. The advantage of having these three approaches together is to solve as many as possible of various problems. If the domain of the problem is having enough and accurate dataset, the decision trees alone or neural networks alone could be used to help in decision making. Problems like this could be marketing and sales prediction and medical diagnosis. If the dataset is not accurate and enough, decision three alone will face a problem of getting good classification. In this case neural network might slightly help in pruning the results that can be obtained by the decision trees. Problems like this can be in control and manufacturing. If the domain which we have is still having enough and accurate dataset but it is really difficult to give a clear classification because the classes have very close results, then heuristic function will help to give a better and more accurate result. In many systems, in such a case fuzzy logic also can

be used. Also, in cases where the dataset is enough and accurate and the system situation does not have absolute true or absolute false, fuzzy logic can be used instead of a heuristic function. Such kind of problems could be related to weather prediction and system control. In this case what so called soft computing (neurofuzzy) is used.

Stage I (Decision Trees)

In the first stage of the system architecture. A very well known algorithm for decision tree induction is ID3 (Quinlan, 1986, 1987). The basic decision tree induction algorithm requires all attributes to be categorical. Many enhancements to the algorithm have been done and incorporated into C4.5 algorithm (Quinlan, 2003). ID3 and C4.5 algorithms have been well established for relatively small data sets. Efficiency and scalability become issues of concern when these algorithms are applied to the mining of very large real-world databases. Since the training millions of samples should reside in the memory, this restriction limits the scalability of such algorithms, where the decision tree construction can become inefficient due to swapping of the training samples in and out of main and cache memories. A new version of C4.5 algorithm was developed and called C5.0. C5.0 algorithm proved to be much faster, less memory usage and more accurate in generating rulesets than C4.5 algorithm. Murthy (1997) gave a comprehensive survey of many of the issues concerned with decision tree induction. Gaines (1994) addressed the problem of transforming the knowledge bases of expert systems using induced rules or decision trees into comprehensible knowledge structures. A knowledge structure is developed that generalizes and subsumes production rules, decision trees and rules with exceptions (rules with exceptions based on ripple down rules suggested by Compton and Jansen (1990) and Compton and Richards (1998). The structure is a directed acyclic graph. The nodes are premises, some of which have attached conclusions and the arcs are inheritance links with disjunctive multiple inheritance. Zhou *et al*. (2001) proposed a belief networks method for rule mining to overcome drawbacks of some of the existing data mining methods, such as classification trees, neural networks and association rules. According to Zhou (2001), using those methods, the user's prior knowledge can not be easily specified and incorporated into the knowledge discovery process and the rules mined from databases lack quantitative analyses. Those drawbacks have been solved by taking the advantage of belief networks as the directed acyclic graph language and their function for numerical representation of probabilistic dependencies among the variables in the databases. Various issues and new algorithms on extracting various kinds of rules have been explored (Zhang and Zhang, 2002).

Once we get the decision tree from the first stage, we extract the classification rules that can help in decision making. In our case we shall not depend only on those rules for making a decision, but we shall consider the extracted rules as a base for the next steps. Referring to Fig. 2, some of the extracted rules are:

- IF Dept = Computer THEN Class = yes
- IF Dept = Mechanical and Level = 1 THEN Class = no
- IF Dept = Mechanical and level = 2 or 3 and computer-based course = yes THEN Class = yes
- IF Dept = Mechanical and level = 2 or 3 and computer-based course = no THEN Class = no
- IF Dept = Mechanical and level >3 THEN class = yes

Some of the drawbacks of decision tree are:

- Whenever the dataset is small, the results might not be accurate and whenever it is very large the induced tree might be confusing
- Some times many attributes are neglected due to lack of situations, which will create a problem in making a decision for cases like them in the future

- The extracted rules are general as much as possible, which will not give any impression about many special cases

These drawbacks will not make the decision trees a good approach for certain domains. This means decision tree alone is not a proper approach in domain independent. To recover from those drawbacks, we use neural network in the next stage.

Stage II (Neural Networks)

In this stage we take the output of the first stage as its input. This means we have to formulate the extracted rules in the form of neural networks. A well known algorithm called KBANN algorithm (Towell and Shavlik, 1991, 1994) is one algorithm that can be used to incorporate the obtained rules in a neural network. The advantages of this step is to make a generalization that can give better decisions about situations which are not known in the first stage. Because KBANN algorithm at its final step uses backpropagation algorithm, it will not be a useful step for us because we might not have extra examples (or otherwise we could have used them for inducing the decision tree). To get usefulness of the KBANN algorithm, we use its rule incorporation step into the neural network and then we use unsupervised learning of kohonen model to cluster the cases. Below is the KBANN algorithm:

- For each instance attribute, create a network input
- For each horn clause in domain theory, create a network unit
  - Connect inputs to attributes tested by antecedents
  - Each non-negated antecedent gets a weight W
  - Each negated antecedent gets a weight –W
  - Threshold weight is –(n-0.5)W, where n is the number of non-negated antecedents.
- Make all other connections between layers, giving these very low weights
- Apply BP using training examples

It is to be noted that if the difference between the sum and the threshold is $\geq$=0.5, it means the output is true, otherwise it is false.

The KBANN algorithm will not benefit us because when it has been proposed it was meant for incorporating rules (prolog forms) into neural networks with existence of positive and negative examples (which is not available in our case) and train the neural using backpropagation algorithm. In our situation, our proposed approach will use a clustering approach with continuous adaptation, which means the decision making system will change its behavior from time to time based on its experience. The more it is used, the better is its decision. To make it clear, given a situation X in time t, the system may give a result of yes class, whereas at t+n might produce no class result for the same situation X. This is due to continuous learning and the gained experience. To get benefit of KBANN algorithm, we propose a semi- KBANN algorithm. The proposed algorithm which we call semi-KBANN is as below:

- From the obtained rules, specify those that fit in class A and those that fit in class B (assume we have two classes)
- For each instance attribute, create a network input
- For each horn clause in domain theory, create a network unit
  - Connect inputs to attributes tested by antecedents
  - Each non-negated antecedent gets a weight W
  - Each negated antecedent gets a weight –W
  - Threshold weight is –(n-0.5)W, where n is the number of non-negated antecedents
  - Make all other connections between layers, giving these very low weights

- Whenever there is a new case, apply kohonen model (competitive learning) to find out to which cluster does it belong. Knowing the cluster, the system can decide to which class it should be classified based on step 1. To clarify this point, if the new case is with the cluster in which the rules of class A reside, then it belongs to class A, otherwise, it belongs to class B. The result of a certain case might change from time to time. The more experience it gets, the better is its current decision
- To stop increasing the size of the list of inputs, we allow only maximum of four repeats for the same input in the list. If all have the same cluster, then keep one and remove the other three from the list. If they are not the same keep all the four in list
- Every n inputs train the complete input list with competitive learning.

The decision of this stage is not the overall system decision. To make a system final decision, we go to stage III, which is the heuristic function.

Stage III (Heuristic Function)

To decide about the system's decision, we build a heuristic function that uses the system experience. Because the decision for a certain case to which class does it belong is based on Ecludian distance, the smaller distance is an indication to the chosen cluster. We normalize the obtained distance between 0-1. Assume we have two classes and since we allow up to four different outputs for the same input case (e.g., outputs could be class A, class B, class A, class B). This means we shall have four values for the classes). Our proposed heuristic function is as below:

For those assigned to class A(cluster A), do the following:

- Compute the absolute value of the difference between Ecludian distances of the cluster A and B
- Sum the differences between the distances, say sum-1

Do the same procedure for Class B and get sum-2. If sum-1 > sum-2, then the system is making a decision of class A, otherwise, the system is making a decision of class B.

For example, we assume that we have four inputs for our neural network, let us consider Table 1.

Computations related to class A = $|0.54-0.73|+|0.71-0.8| = 0.28$
Computations related to class B = $|0.6-0.54|+|0.45-0.6| = 0.21$

The system will make a decision of class A.

It is to be noted that if we have only one case, the system will take a decision in favor of its corresponding assigned class. If we have two cases with the same classification, the system will also have their corresponding class. In case each of the two has a different class, our formula will help in the decision and in case of three repeats for the input and all have the same classification, the system will make a decision of the same class and if two are classified of class A, where the third is classified of class B, we compare the distance difference of class B (Val-1) with the average of the distance differences (Val-2). The system then chooses the class with the maximum of the two values Val-1 and Val-2. Assume we have values as shown in Table 1.

We compute the difference in the distances for class A:

$|0.54-0.73| = 0.19$
$|0.71-0.8| = 0.09$

Table 1: An example of four results of the same data input

Class assigned for (1,1,0,1) input	Ecludian distance from class A	Ecludian distance from class B
A	0.54	0.73
B	0.60	0.54
A	0.71	0.80
B	0.45	0.60

Table 2: An example of three results of the same data input

Class assigned for (1,1,0,1) input	Ecludian distance from class A	Ecludian distance from class B
A	0.54	0.73
B	0.60	0.54
A	0.71	0.80

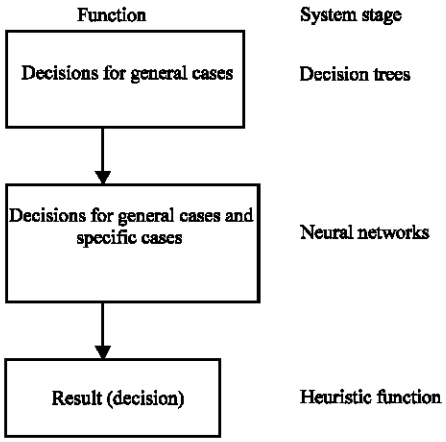

Fig. 3: The decision making system stages and each stage's function

We take the average distance = (0.19+0.09)/2 = 0.14

We compute the difference in distance for class B:

$$|0.6-0.54| = 0.06$$

We compare the average difference of distance of class A with the average difference of distance of class B. Since, 0.14>0.06, the system makes a decision of class A. We use the same as the above method if we have four outputs for the same input case with three similar outputs and one is different.

CONCLUSION

In this study we have presented a theoretical general architecture for a decision making system. The proposed system is domain independent because it uses various approaches incorporated into each other and the drawback of a certain approach can be recovered by another approach. The used approaches are three and categorized into three sequential stages. The overview of the stages and the function of each is presented in Fig. 3. In the first stage, a decision tree is induced from the given dataset and then classification rules are extracted. In the second stage the obtained extracted classification rules are incorporated into a neural network using a proposed algorithm called semi-KBANN algorithm. The second stage makes the system decisions more general where generalization can be obtained from the neural networks training which is based on kohonen model of unsupervised

learning. This means the system will be able to guide us in cases of no previous experience in the first stage. The more is the system 's experience (more cases), the better is the result of this stage. In the third stage, maximum of four repeats are allowed and saved in the training list along with their Ecludian distances. A heuristic function is used to take a final decision about the given case, it depends on the sum of the differences of the Ecludian distances for certain cluster (class). The class with max sum is chosen as the system 's decision. The future directions could be implementing the proposed system and apply it on various domains and compare the results with unique approaches.

REFERENCES

Compton, R. and A. Jansen, 1990. Philosophical basis for knowledge acquisition. Knowledge Acquisition, 2: 241-257.

Compton, P. and D. Richards, 1998. Taking up the situated cognition challenge with ripple down rule. Int. J. Hum. Comp. Studies, Special Issue Situated Cogni., 49: 895-926.

Druzdzel, M. and R. Flynn, 2002. Decision Support Systems. In: Encyclopedia of Library and Information Science, Kent, A. (Ed.), Marcel Dekker, Inc., New York.

Gaines, B., 1994. Transforming rules and trees into comprehensible knowledge structures. http://ksi.cpsc.ucalary.ca/articles/induct/edag94/

Karakul, M. and H. Qudrat-Ullah, 2007. How to Improve Dynamic Decision Making? Practice and Promise. In: Complex Decision Making Theory and Practice, Qudrat–Uallah, H. *et al*. (Ed.). Springer, Berline, pp: 3-24.

Murthy, S., 1997. Automatic construction of decision trees from data: A multi disciplinary survey. Data Mining Knowledge Discovery, 2: 259-389.

Quinlan, J.R., 1986. Induction of decision trees. Machine Learn., 1: 81-106.

Quinlan, J., 1987. Simplifying decision trees. Int. J. Man-Machine Studies, 27: 221-234.

Quinlan, J., 2003. C4.5: Programs for Machine Learning. 5th Edn., Morgan Kaufmann, San Mateo, CA.

Towell, G. and J. Shavlik, 1991. Interpretation of artificial neural networks: Mapping knowledge base neural networks into rules. Proceedings of Advances in Neural Information Processing Systems, 1991, Denver, Co., Morgan Kaufmann, pp: 977-984.

Towel, G. and S. Shavlik, 1994. Knowledge base artificial neural networks. Artificial Intelli., 70: 119-165.

Zhang, C. and S. Zhang, 2002. Association Rule Mining : Models and Algorithms. Lecture Notes Artificial Intelligence. Vol. 2307, Springer, Berlin, ISBN 978-3-540-43533-4, pp: 238.

Zhou, Z., L. Huan, Z.L. Stan and C.H. Chin, 2001. Rule mining prior knowledge: A belief network approach. Intell. Data Anal., 5: 95-110.