A CONNEXIONIST APPROACH FOR CASE BASED REASONING

Antonio B. Bailón Miguel Delgado Eva Gibaja
Dpto. Ciencias de la Computación e Inteligencia Artificial
E.T.S. Ingeniería Informática, Universidad de Granada
c/ Daniel Saucedo Aranda s/n, 18071 Granada (Spain)

José María de la Torre
Dpto. Economía Financiera y Contabilidad
Facultad de Ciencias Económicas, Universidad de Granada
Campus Universitario de La Cartuja, 18071 Granada (Spain)

Keywords: Case Based Reasoning, Associative Memories, Connexionist Model

Abstract: Case Based Learning is an approach to automatic learning and reasoning based on the use of the knowledge gained in past experiences to solve new problems. To suggest a solution for a new problem it is necessary to search for similar problems in the base of problems for which we know their solutions. After selecting one or more similar problems their solutions are used to elaborate a suggested solution for the new problem. Associative memories recover patterns based on their similarity with a new input pattern. This behaviour made them useful to store the base of cases of a Case Based Reasoning system. In this paper we analyze the use of a special model of associative memory named CCLAM (Bailón et al., 2002) with this objective. To test the potentiality of the tool we will discuss its use in a particular application: the detection of the "health" of a company.

1 INTRODUCTION

In real life, when we have to deal with a problem our memory evokes past similar situations. Those similar problems were solved in the past and now we remember the actions that helped us to solve them (or at least we remember the actions that we tried but didn't solve the problems). If a new problem is similar to a problem that was previously solved we can reuse its solution (possibly after revising it) to solve the current problem. If our actions are successful we will remember them if we need to solve a similar problem.

Case based reasoning relies on the existence of a base of cases that stores our experience in a particular problem. When a new problem arrives, the base is searched for a similar case to reuse its solution. New cases are stored in the base of cases when they are solved (Aamodt and Plaza, 1994).

The role of the memory is very important because the process relies on its behaviour storing cases and retrieving cases based on their similarity with a new one. Associative memories receive an input patern and retrieve the most similar stored pattern. This is the expected behaviour of the base of cases in case based reasoning and that's why we use associative memories.

2 CONTINUOUS CLASSIFYING ASSOCIATIVE MEMORY

The Continuous Classifying Associative Memory CCLAM is a memory model which allows the storage of arbitrary analogical patterns which have continuous components in the interval $[0, 1]$.

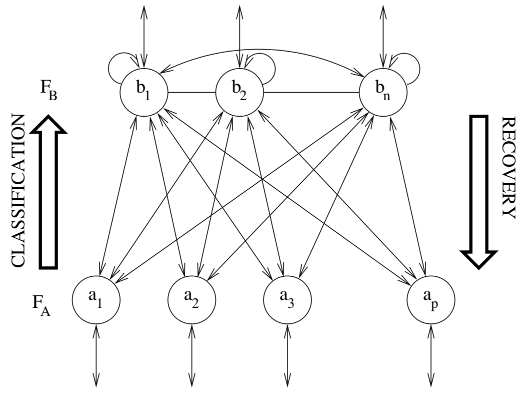

Figure 1: Continuous Classifying Associative Memory (CCLAM).

In the operation of the CCLAM we can distinguish two processes: classification and recovery. The clas-

B. Bailón A., Delgado M., Gibaja E. and María de la Torre J. (2004).
A CONNEXIONIST APPROACH FOR CASE BASED REASONING.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 369-374
DOI: 10.5220/0002641403690374
Copyright © SciTePress

sification process receives an input pattern in the F_A layer (Fig.1) and obtains a value in each processing element (PE) of F_B which indicates the degree to which the input pattern is similar to the stored pattern represented by this PE. The recovery process receives in F_B the extent to which we refer to each of the stored patterns and recovers a combination of these patterns in F_A.

The F_B layer is competitive. The weights of the competition signals between the PEs of the F_B layer are represented in the matrix $\beta_{n\times n}$, with n being the number of stored patterns and β_{ij} being the weight of the connection which exists from b_i to b_j. The winning element is the one whose activation state Y_i verifies $Y_i \geq Y_j\beta_{ji} \quad \forall j$. For the competition we can establish whether the winning PE will retain its activation state or take the value 1 as its new state.

The behavior of the CCLAM depends on:

- The weigths of the connections made between the layers F_A and F_B, represented in the matrix M_{np}.
- The functions Υ and Ψ that control the propagation of the information between the layers.
- The weigths of the competition connections made in the F_B layer.
- Whether the winning PE in F_B will retain its value or take value 1.

This behavior of the classification and recovery processes may be different. To distinguish the weights and functions used in each process we will name them using the superscripts C and R appropiately.

2.1 Learning

In the learning process, the memorization of a pattern entails the creation of a new PE in F_B and the appropriate adjustment of the weights of the connections which are carried out towards it. Learning has the following steps:

1. Arrival at F_A of a pattern to be memorized.
2. Verification that the pattern was not previously stored.
3. Creation of a new PE in F_B associated to the pattern.
4. Adjustment of the competition weights of the F_B layer.
5. Adjustment of the weights of the connections made between F_A and the new PE.

2.2 Classification

In the classification process the PEs send activation signals from the F_A layer towards the F_B layer. Given an input pattern presented in the F_A layer activation states are obtained in the F_B layer which represent the similarity which exists between the input pattern and each of the stored patterns. The classification process has the following steps:

1. A pattern $X = (x_1, \ldots, x_p)$ arrives at the F_A layer.
2. The signal propagates towards the F_B layer so that the PE b_i receives the activation signal:
   (a) $y_i = \Psi^C \left\{ \Upsilon^C \left(x_i, m_{ij}^C \right) \right\}_{j=1\ldots p}$
   (b) Υ^C and Ψ^C are functions bounded to the interval $[0, 1]$. The choice of these functions depends on the behavior which we would like the memory to present.
3. The PEs of the F_B layer compete amongst themselves with the intensity reflected in the weights matrix β^C.
4. We obtain the result of the classification process in the F_B layer. The activation state obtained in the PEs of the F_B layer indicates the degree to which the input pattern is classified by each of the classes represented by the stored patterns.

2.3 Recovery

In the recovery process, the information flows from F_B to F_A. In the F_B layer is presented an input pattern which represents the degree to which recovery involves each of the stored patterns. The recovery has the following steps:

1. A pattern $Y = (y_1, \ldots, y_n)$ arrives at the F_B layer.
2. The PEs of the F_B layer compete amongst themselves with the weights matrix β^R. The new activation states form the pattern $Y' = (y_1', \ldots, y_n')$.
3. The signal propagates from the F_B layer towards the F_A layer so that the PE a_i receives the activation signal:
   (a) $x_i = \Psi^R \left\{ \Upsilon^R \left(y_j', m_{ji}^R \right) \right\}_{j=1\ldots n}$
   (b) Υ^R and Ψ^R are functions bound to the interval $[0, 1]$.The choice of these functions depends on the behavior which we would like the memory to present.
4. We obtain the pattern resulting from the recovery process in the F_A layer. This might be one of the stored patterns or a combination of them.

3 CASE BASED REASONING USING CCLAM

Our goal is to use the CCLAM in case-based reasoning to allow to:

- Store the patterns included in the set of known cases.
- Compute the similarity that exists between a new case and each of the stored cases.
- Select the cases that will take part in the elaboration of the solution proposed for a new case.
- Obtain the proposed solution as the solution or combination of solutions of several known cases.
- Store the new cases.
- Forget the cases that don't provide a solution different from those provided by a large number of the nearest known cases.

The CCLAM can store arbitrary continuous patterns while other associative memories present spurious states and have a strong dependency on the kind of patterns that they can store. For that reason the CCLAM can be used in case based reasoning allowing the storage and correct retrieval of known cases.

In each case $X = (x_1, x_2, \ldots, x_n)$ we can distinguish the attributes that define the problem from those attributes that define the solution. Each case X is formed by two patterns (P, S) such as $P = (p_1, p_2, \ldots, p_{n_p})$, $p_i \in \{x_j\}_{j=1\ldots n}$, is the pattern that contains the attributes of the problem and $S = (s_1, s_2, \ldots, s_{n_s})$,$s_i \in \{x_j\}_{j=1\ldots n}$, contains the attributes of the solution of case X.

Let's suppose that each problem has only one solution. If we find the same problem with two different solutions it is possible that:

- One of the solutions embeds the other because it is more general. We shoul decide if it is better to store the more general or the more particular solution.
- The solutions are incompatible. The solution of a problem can't be used to solve the other one althougt the two problems are expressed using the same attributes. This conflict arises when the attributes that define the problems aren't enough to distinguish them. To solve the conflict we must augment the attributes used to represent a problem to include those in what the actual cases differ.

To store the cases we will use three CCLAM. One of them will store the patterns that encode the attributes of the problems and will be named P-Memory. Other memory will store the attributes of the solutions and will be named S-Memory. The two memories will be connected by means of a third CCLAM that selects the solution that is recovered based on the similarities found between the stored cases and a new problem (Fig. 2). This CCLAM that links the P-Memory and the S-Memory will be named L-Memory.

3.1 P-Memory

The P-Memory has as many processing elements (PE) in the F_A layer as attributes of the problem. In the F_B layer has as many PE as stored problems.

The weight of the connection that links the a_i PE in the F_A layer to the b_j PE of layer F_B represents the value of the i-th attribute in the j-th stored problem.

In the classification process the memory receives in F_A a pattern that represents a new problem and will provide in the F_B layer the degree of similarity that exists between the new problem and each of the stored problems.

3.1.1 Similarity between problems

In case-based reasoning it's neccessary to compare a new problem with all the stored problems to obtain the similarity that exists between them. We need a function that computes the similarity that exists between the problems. We use a function that measures the distance between the patterns (the greater the distance between the patterns, the lower the simiarity between the problems). The most commonly used functions are euclidean and manhattan distance.

We can configure the P-Memory to compute the distance between the new problem and each known problem. Depending on the configuration we can compute any particular instance of the Minkowski distance (1). When $p = 1$ it computes the manhattan distance and when $p = 2$ it computes the euclidean distance.

$$D_p(A, B) = \left(\sum_{i=1}^{N} |a_i - b_i|^p \right)^{\frac{1}{p}} \tag{1}$$

To compute the similarity that exists between the patterns we must take into account that the attributes can have different relative importance. For example lets represent the problem of selecting people to form a basketball team based on their height and weight. Candidates will be represented with two attributes that measure their height in meters, weight in kilograms and classification as good or bad candidate. We store the following cases:

$$\begin{array}{l} \text{Case 1}: \ (1.9m, 70kg, good) \\ \text{Case 2}: \ (1.4m, 65kg, bad) \end{array} \tag{2}$$

We recognize in case 1 a tall slim person and in case 2 a short fat person. When a new candidate arrives we have to decide wether to select him for the team. The new case $(1.89m, 67kg)$ will be compared with cases 1 and 2 to decide its classification. The manhattan and euclidean distances show that the nearest stored pattern is case 2. But intuitively we recognize in the new case a tall slim person that is more similar to case 1.

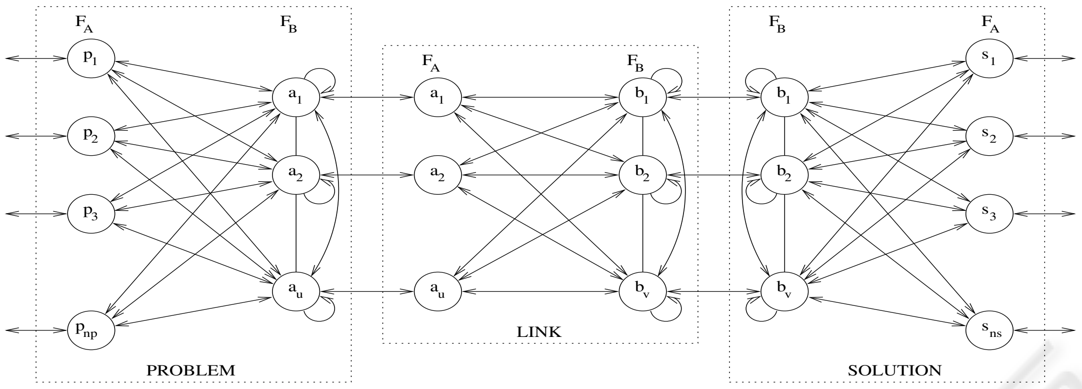

Figure 2: Case Based Reasoning using CCLAM.

This happens because we have assigned the same importance to the height and the weight of the person but in this particular problem of selecting people to play basketball the height of a person is more important than his weigth.

$$
\begin{array}{l}
D_1\left((1.89, 67), (1.9, 70)\right) = 3.01 \\
D_2\left((1.89, 67), (1.9, 70)\right) \approx 3 \\
D_1\left((1.89, 67), (1.4, 65)\right) = 2.49 \\
D_2\left((1.89, 67), (1.4, 65)\right) \approx 2.06
\end{array}
\tag{3}
$$

We need to assign a relative importance to each attribute so that the distance between patterns reflects it. If we assign wheight w_i to attribute p_i we must configure the CCLAM to compute the similarity between patterns:

$$
\begin{array}{l}
\Upsilon^C\left(x_i, m_{ij}\right) = |x_i - m_{ij}| \\
\Psi^C\left(a_1, \ldots, a_n\right) = 1 - \sqrt[p]{\frac{\sum_{i=1}^{n}(w_i a_i)^p}{\sum_{i=1}^{n} w_i^p}}
\end{array}
\tag{4}
$$

The attributes of the problems are normalized to give patterns that can be stored in the memory. Each attribute takes its value in $[0, 1]$ and then the Minkowski distance $\sqrt[p]{\frac{\sum_{i=1}^{n}(w_i a_i)^p}{\sum_{i=1}^{n} w_i^p}}$ takes its value in $[0, 1]$ too. The value computed by the Ψ^C function is the similarity between the patterns measured in the interval $[0, 1]$.

The weights used to represent the relative importance of the attributes can be established by an expert or can be learned from the known cases. We can use cross validation to adjust the weights. If the number of cases is high enough, when case C is removed from the memory the rest of the cases can give the correct solution to problem C. The goal is a weight assignment that maximizes the number of cases that are correctly solved if they are removed from memory. To learn the weights we can use several methods such as simulated annealing or genetic algorithms. The weights can be revised periodically to reflect the knowledge adquired with the new cases learned after the previous assignment of weights.

3.2 L-Memory

The CCLAM that links the P-Memory and the S-Memory has in the F_A layer one PE for each stored problem and one PE in the F_B layer for each stored solution. There is a connection with weight 1 between the PE in F_A that represents a problem and the PE in F_B that represents the solution associated to that problem. The rest of connections have weight 0.

The CCLAM receives in the F_A layer the degree of similarity that exists between a new problem and each stored problem (this information comes from the P-Memory). After the classification process the only active element in F_B represents the proposed solution.

In some situations we can obtain the solution as a combination of several stored solutions of similar problems. Then we can obtain in F_B the degree with which the solutions can be combined. This can be useful in problems where the solution is a real number that represents a physical magnitude.

3.2.1 Wheight of the environment of the problem

Let's suppose that a new problem P arrives. The P-Memory finds one case with similarity 0.6 that suggests solution A. It also finds 100 cases with similarity 0.59 and all of them suggest solution B. Which solution should be proposed?

The answer depends on the problem. In general we can use a voting system. Each known problem suggests its own solution with a degree that depends on its similarity to the new problem. Then the proposed solution will be that with the highest number of votes.

To compute the weight of a vote from the similarity of the problem we use a non decreasing function Δ :

$[0,1] \rightarrow [0,1]$ with the properties:

$$\begin{array}{l}\Delta(0) = 0 \\ \Delta(1) = 1\end{array} \quad (5)$$

The L-Memory computes the sum of the degrees with which the stored problems vote for each solution. Let's suppose that a new case arrives and the CCLAM that stored the problems computes the similarities $(s_1, s_2, \ldots, s_n)$. The L-Memory receives the similarities in the layer F_A. The weight of the connection made between the i-th PE in F_A and the j-th PE in F_B is m_{ij}. The memory is configured with:

$$\begin{array}{l}\Upsilon^C(s_i, m_{ij}) = \Delta(s_i)\, m_{ij} \\ \Psi^C(x_1, x_2, \ldots, x_n) = \frac{1}{n}\sum_{i=1}^{n} x_i\end{array} \quad (6)$$

After the classification process we obtain in F_B layer the degree with which each solution is sugested by the stored cases.

If we can't combine several solutions to obtain a new one then we select the PE that represents the most voted solution to be the only PE that will remain active.

If we will combine several solutions we must adjust the degrees with which each one is voted so that the sum of all of them equals one to ensure that we obtain the wheighted mean of the solutions. The result is sent to the S-Memory.

3.3 S-Memory

The S-Memory has in the F_A layer a PE for each attribute of the solutions and one PE in the F_B layer for each stored solution.

The weight of the connection made between element i in F_B and element j in F_A represents the value of the j-th attribute of the i-th solution.

The recovery process obtains in F_A layer the solution sugested by the activation degrees received in F_B.

3.3.1 Recovery of the solution

To obtain in F_A the solution that will be suggested to solve the new problem we must configure the CCLAM with:

$$\begin{array}{l}\Upsilon^R(s_i, m_{ij}) = s_i m_{ij} \\ \Psi^R(x_1, x_2, \ldots, x_n) = \frac{1}{n}\sum_{i=1}^{n} x_i\end{array} \quad (7)$$

This configuration computes the weighted average of the stored solutions. The weight of each solution depends on the similarity that exists between its problem and the new case. If only one PE in F_B is active with activation 1 and the rest remain inactive the CCLAM recovers the most voted solution.

With the recovered solution the CCLAM can show the degree with which it was recalled. This can be interpreted as a confidence measure.

3.4 Case Pruning

The set of problems that share the same solution can be found grouped in clusters. Those problems lying in the perimeter of the cluster are more important because they give more information than the others. Then if we need to reduce the number of stored cases we can remove the cases that give less information.

To store only the new cases that give new information we must study the solutions suggested by our CCLAM-based CBR system. Let the solution of the new case be A and the suggested solutions A,B,C... suggested with degrees $s(A), s(B), s(C), \ldots$. The new case will be stored if the difference between the degree of the correct solution and the sum of the degrees of the incorrect solutions is lower that a established threshold.

A pattern located in the interior of a cluster makes the correct solution be suggested with a high degree because there are many near patterns with the correct solution. If the pattern is near the perimeter of the cluster then the influnce of patterns with wrong solutions is high.

Setting the threshold to a convenient value will forbid the store of patterns that are clearly in the interior of a cluster of patterns with which it shares the same solution.

The prune of cases must be done very carefully because we lose information when we don't store a pattern.

4 APPLICATION: A CASE STUDY

One problem that is receiving an increasing attention in the literature is that of prediction of a company health (behaviour).

The idea is to represent any company by a vector $(x_1, \ldots, x_n)$ of attribute values and try from that to detect the company health.

Let us consider this health is represented in this turn by a new boolean variable, h, such that $h = 0$ represents a bad health and $h = 1$ represents a good helth.

Clasical prediction (classification) models try to identify an input-output relation like $h = H(x_1, \ldots, x_n)$, where function H is learned from a set of examples by some inductive procedure.

This approach has shown some inefficiency because it's very difficult to capture the knowledge about the health of a company as contained in a set $\left\{\left(x_1^i, x_2^i, \ldots, x_n^i, h^i\right)\right\}_{i=1\ldots T}$ of training examples. Semantics, scales, degree of importance, etc. of this variables troubles us with this task. For this reason some different approaches have raised in last years (artificial neural networks, Kohonen maps, ...).

Here we will investigate the use of our CBR approach by CCLAM to classify enterprises in the above described sense. In the following we describe a summary of our experiment and its conclusions, being impossible to include here a detailed description of them because the size and the amount of data and algorithmic steps.

Te interested reader may consult http://decsai.ugr.es/ to obtain the whole set of data and the complete discussion about the experiment.

The training set contains information about 1500 companies each described by 53 attributes. That is in our case $n = 53$ and $T = 1500$.

The variables have very different semantics and according to that different domains and scales. In fact we have:

- 36 continuous attributes (ex. debt level)
- 9 integer attributes (ex. number of employees)
- 6 boolean attributes (ex. whether the company has been audited)
- 2 categorial attributes (ex. economic activity code)

To take into account the different distance functions (or similarity functions) needed to compare different kinds of attributes we must configure the P-Memory so that there will be different Ψ^C and Υ^C functions depending on the kind of attribute that represents each processing element.

We don't know the degree of influence of each attribute in the health of a company. To estimate the relative importance of the attributes we measure the goodness of a set of weights as the amount of correct classifications of cases in a cross validation process.

We assume that similar companies have the same classification (this assumption is the key of case based reasoning). We also assume that the number of known cases is high enough. If the weights were right then when we extract a case from the memory and use it as a new problem there will be similar problems that will classify it correctly. We repeat this operation with all the cases and the percentage of correct classifications is the value that measures the goodness of the set of weights.

To find an appropiate set of weights for the training set of cases we have used genetic algorithms to find initial good solutions that were then refined using simulated annealing.

Once configured our CCLAM based CBR system with the appropiate weights it was tested with a set of 764 new cases to measure the number of correct classifications obtaining a 73% of success that represents a good result that justifies the use of the system.

5 CONCLUSIONS

- The CCLAM is a good choice for case based reasoning because it can store any number of arbitrary cases and it don't recover cases not stored previously because of the absence of spurious states.
- The use of two CCLAM to store the problems and the solutions in different memories allows the correct computation ofthe similarity of the problems and the correct retrieval of the suggested solution.
- The L-Memory allows the use of a voting system to select the most voted of the solutions proposed by the stored cases.
- The similarity between problems is computed by means of the generalized minkowsky distance with the correct configuration of the P-Memory.
- When the solution is expressed with attributes that can be aggregated we can obtain a lineal combination of the sugested solutions.

REFERENCES

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

Aha, D. W. (1998). The omnipresence of case-based reasoning in science and application. *Knowledge-Based Systems*, 11(5-6):261–273.

Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, (6):37–66.

Bailón, A., Delgado, M., and Fajardo, W. (2002). Continuous classifying associative memory. *International Journal of Intelligent Systems*, 17(4):391–407.

Bartsch-Sprl, B., Lenz, M., and Hbner, A. Case-based reasoning – survey and future directions.

Globig, C. and Wess, S. (1995). Learning in case-based classification algorithms. In Jantke, K. P. and Lange, S., editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961, pages 340–362. Springer-Verlag.

Leake, D. B. (1996). *Case-Based Reasoning: Experiences, Lessons, and Future Directions*. MIT Press.

Richter, M. M. (1992). Classification and learning of similarity measures. Technical Report SR-92-18.

Wilson, D. R. and Martinez, T. R. (1997). Instance pruning techniques. In *Proc. 14th International Conference on Machine Learning*, pages 403–411. Morgan Kaufmann.