# Integrating Hybrid Rule-Based with Case-Based Reasoning

Jim Prentzas and Ioannis Hatzilygeroudis

University of Patras, School of Engineering
Dept of Computer Engin. & Informatics, 26500 Patras, Hellas (Greece)
and
Computer Technology Institute, P.O. Box 1122, 26110 Patras, Hellas (Greece)
{prentzas, ihatz}@ceid.upatras.gr, ihatz@cti.gr

**Abstract.** In this paper, we present an approach integrating neurule-based and case-based reasoning. Neurules are a kind of hybrid rules that combine a symbolic (production rules) and a connectionist representation (adaline unit). Each neurule is represented as an adaline unit. One way that the neurules can be produced is from symbolic rules by merging the symbolic rules having the same conclusion. In this way, the number of rules in the rule base is decreased. If the symbolic rules, acting as source knowledge of the neurules, do not cover the full complexities of the domain, accuracy of the produced neurules is affected as well. To improve accuracy, neurules can be integrated with cases representing their exceptions. The integration approach enhances a previous method integrating symbolic rules with cases. The use of neurules instead of symbolic rules improves the efficiency of the inference mechanism and allows for drawing conclusions even if some of the inputs are unknown.

## 1 Introduction

Symbolic rules constitute a popular knowledge representation scheme used in the development of expert systems. Rules represent general knowledge of the domain. They exhibit a number of attractive features such as naturalness, modularity and ease of explanation. One of their major drawbacks is the difficulty in acquiring them. The traditional process of eliciting rules through the interaction with the expert may turn out to be a bottleneck causing delays in the system's overall development. Furthermore, the acquired rules may be imperfect and not covering the full complexities of the domain. Rule induction methods deal with many of these disadvantages but may still be unable to recognize exceptions in small, low frequency sections of the domain [7].

Case-based reasoning offers some advantages compared to symbolic rules and other knowledge representation formalisms. Cases represent specific knowledge of the domain. Cases are natural and usually easy to obtain [1], [14], [17]. New cases can be inserted into a knowledge base without making changes to the preexisting knowledge. Incremental learning comes natural to case-based reasoning. The more cases are available the better the domain knowledge will be represented. Therefore, the accuracy of a case-based system can be enhanced throughout its operation, as new cases become available. A negative aspect of cases compared to symbolic rules is that they do not provide concise representations of the incorporated knowledge. Furthermore, the time-performance of the retrieval operations is not always the desirable.

Approaches integrating rule-based and case-based reasoning have given interesting and effective knowledge representation schemes [2], [6], [7], [8], [16], [18], [21]. The goal of these efforts is to derive hybrid representations that augment the positive aspects of the integrated formalisms and simultaneously minimize their negative aspects. In [11] the approaches integrating rule-based and case-based reasoning are distinguished into two basic categories: efficiency-improving and accuracy-improving methods. The former concern integration methods in which rules and cases are dependent, meaning that one representation scheme was derived from the other (i.e., rules derived from cases or vice versa), and the efficiency of the integrated scheme exceeds the efficiency that could have been achieved with rules or cases alone (e.g. [3], [15], [22]). The latter involves approaches in which the two representation schemes are independent and their integration results in improved accuracy compared to each one representation scheme working individually (e.g. [5], [11], [20]).

A popular integration method resulting in accuracy improvement is described in [11]. This approach involves integration of independent rules and cases in an innovative way. The purpose of this integration is to use cases in order to enhance a set of symbolic rules that is only approximately correct. On the one hand, cases are used as exceptions to the symbolic rules filling their gaps in representing domain knowledge. On the other hand, symbolic rules perform indexing of the cases facilitating their retrieval. Experimental results have demonstrated the effectiveness of the integration.

However, the performance of this method can be further enhanced if neurules [12] are used instead of pure symbolic rules. Neurules are a type of hybrid rules integrating symbolic rules with neurocomputing. Their main characteristic is that they retain the modularity of production rules and also their naturalness in a great degree.

One way that the neurules can be produced is from symbolic rules [12]. However, if the symbolic rules, acting as source knowledge to the neurule base, do not cover the full complexities of the domain, the accuracy of the resulting neurule base will be affected as well. Integrating neurules with cases in an approach similar to the one described in [11] can create an effective scheme combining three types of knowledge representation formalisms: symbolic rules, neural networks and cases. Such integration is justified by the fact that the approach described in [11] requires efficient symbolic rule-based reasoning acting as the indexing component of case-based reasoning. Neurules improve the performance of symbolic rules since neurule-based reasoning is more efficient than symbolic rule-based reasoning [12].

In this paper, we present an approach for effectively combining neurule-base and case-based reasoning. This approach improves on the one hand the accuracy of the neurules and on the other hand the performance of the approach presented in [11]. The rest of the paper is organized as following. Section 2 presents neurules, whereas

Section 3 presents the architecture for integrating neurule-based and case-based reasoning. Section 4 presents methods for constructing the indexing scheme of the case library. Section 5 describes the hybrid inference mechanism. Section 6 presents experimental results regarding the performance of the inference process. Finally, Section 7 concludes.

## 2 Neurules

Neurules are a type of hybrid rules integrating symbolic rules with neurocomputing giving pre-eminence to the symbolic component. Neurocomputing is used within the symbolic framework to improve the performance of symbolic rules [12]. In contrast to other hybrid approaches (e.g. [9], [10]), the constructed knowledge base retains the modularity of production rules, since it consists of autonomous units (neurules), and also retains their naturalness in a great degree, since neurules look much like symbolic rules. Also, the inference mechanism is a tightly integrated process, which results in more efficient inferences than those of symbolic rules. Explanations in the form of if-then rules can be produced [13].

### 2.1 Syntax and Semantics

The form of a neurule is depicted in Fig.1a. Each condition $C_i$ is assigned a number $sf_i$, called its *significance factor*. Moreover, each rule itself is assigned a number $sf_0$, called its *bias factor*. Internally, each neurule is considered as an adaline unit (Fig.1b). The *inputs* $C_i$ ($i$=1,...,$n$) of the unit are the *conditions* of the rule. The weights of the unit are the significance factors of the neurule and its bias is the bias factor of the neurule. Each input takes a value from the following set of discrete values: [1 (true), -1 (false), 0 (unknown)]. This gives the opportunity to easily distinguish between the falsity and the absence of a condition in contrast to symbolic rules. The *output D*, which represents the *conclusion* (decision) of the rule, is calculated via the standard formulas:

$$D = f(\mathbf{a})\,, \quad \mathbf{a} = sf_0 + \sum_{i=1}^{n} sf_i \; C_i$$

$$f(\mathbf{a}) = \begin{cases} 1 & if\ \mathbf{a} \geq 0 \\ -1 & otherwise \end{cases}$$

where **a** is the *activation value* and *f(x)* the *activation function*, a threshold function. Hence, the output can take one of two values ('-1', '1') representing failure and success of the rule respectively.

The general syntax of a condition $C_i$ and the conclusion $D$ is:

<condition>::= <variable> <l-predicate> <value>
<conclusion>::= <variable> <r-predicate> <value>

where <variable> denotes a *variable*, that is a symbol representing a concept in the domain, e.g. 'sex', 'pain' etc, in a medical domain. <l-predicate> denotes a symbolic or a numeric predicate. The symbolic predicates are {is, isnot} whereas the numeric predicates are {<, >, =}. <r-predicate> can only be a symbolic predicate. <value> denotes a value. It can be a *symbol* or a *number*. Corresponding symbolic rules have the same syntax as that in Fig.1a, without the significance factors and with ',' denoting conjunction. The significance factor of a condition represents the significance (weight) of the condition in drawing the conclusion(s). Table 1 presents two example rules, a symbolic (R1) and a neurule (N1), from a medical diagnosis domain.

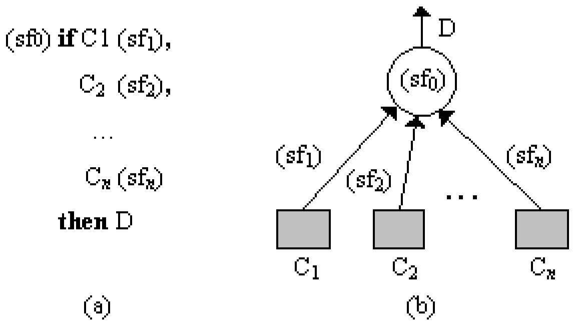

**Fig. 1.** (a) Form of a neurule (b) a neurule as an adaline unit

A variable in a condition can be either an input variable or an intermediate variable or even an output variable, whereas a variable in a conclusion can be either an intermediate or an output variable. An input variable takes values from the user (input data), whereas intermediate or output variables take values through inference since they represent intermediate and final conclusions respectively. We distinguish between intermediate and output neurules. An intermediate neurule is a neurule having at least one intermediate variable in its conditions and intermediate variables in its conclusions. An output neurule is one having an output variable in its conclusions.

**Table 1.** An example (a) symbolic rule and (b) neurule

| | |
|---|---|
| R1<br>**if** sex is man,<br>age > 20,<br>age < 36<br>**then** patient-class is man21-35 | N1<br>(-4.2) **if** pain is continuous (3.0),<br>patient-class isnot man36-55 (2.8),<br>fever is medium (2.7),<br>fever is high (2.7)<br>**then** disease-type is inflammation |
| (a) | (b) |

Neurules can be constructed from symbolic rules thus exploiting existing symbolic rule bases. This process is fully described in [12]. According to the process, symbolic rules having the same conclusion are organized into merger sets and an adaline unit is initially assigned to each one of them. Each unit is individually trained via the Least Mean Square (LMS) algorithm. Its training set is based on the combined logical function of the rules (i.e., the disjunction of the conjunctions of their conditions) in the merger set. When the training set is inseparable, special techniques are used. In that case, more than one neurule having the same conclusion are produced. Section 4.2 contains an example merger set (Table 3) and the corresponding produced neurule (Table 5).

## 2.2 The Neurule-Based Inference Engine

The neurule-based inference engine performs a task of classification: based on the values of the condition variables and the weighted sums of the conditions, conclusions are reached. It gives pre-eminence to symbolic reasoning, based on a backward chaining strategy [12]. As soon as the initial input data is given and put in the working memory, the output neurules are considered for evaluation. One of them is selected for evaluation. Selection is based on textual order. A neurule fires if the output of the corresponding adaline unit is computed to be '1' after evaluation of its conditions. A neurule is said to be 'blocked' if the output of the corresponding adaline unit is computed to be '-1' after evaluation of its conditions.

A condition evaluates to 'true', if it matches a fact in the working memory, that is there is a fact with the same variable, predicate and value. A condition evaluates to 'unknown', if there is a fact with the same variable, predicate and 'unknown' as its value. A condition cannot be evaluated if there is no fact in the working memory with the same variable. In this case, either a question is made to the user to provide data for the variable, in case of an input variable, or an intermediate neurule with a conclusion containing the variable is examined, in case of an intermediate variable. A condition with an input variable evaluates to 'false', if there is a fact in the working memory with the same variable, predicate and different value. A condition with an intermediate variable evaluates to 'false' if additionally to the latter there is no unevaluated intermediate neurule that has a conclusion with the same variable. Inference stops either when one or more output neurules are fired (success) or there is no further action (failure).

Conditions of neurules are organized according to the descending order of their significance factors. This facilitates inference. When a neurule is examined in the inference process, not all of its conditions need to be evaluated. Evaluation of a neurule's conditions proceeds until their weighted sum exceeds the remaining sum (i.e., sum of the absolute values of the unevaluated conditions' significance factors).

When a neurule base is produced from a symbolic rule base, experimental results have shown that neurule-based inference is more efficient than the corresponding symbolic rule-based inference [12]. The main reason for this is the fact that a neurule is a merger of usually more than one symbolic rule having the same conclusion and thus the total number of rules in the rule base is reduced. In this way, the number of rules participating in the inference process is reduced and so is the number of the evaluated conditions. Another advantage of neurule-based reasoning compared to symbolic rule-based reasoning is the ability to reach conclusions from neurules even if some of the conditions are unknown. This is not possible in symbolic rule-based reasoning. A symbolic rule needs all its conditions to be known in order to produce a conclusion.

## 3 The Hybrid Architecture

In Fig. 2, the architecture of the hybrid system implementing the method for integrating neurule-based and case-based reasoning is presented. The run-time system (in the dashed shape) consists of the following modules: the *working memory*, the *hybrid inference mechanism*, the *explanation mechanism*, the *neurule base* and the *indexed case library*.

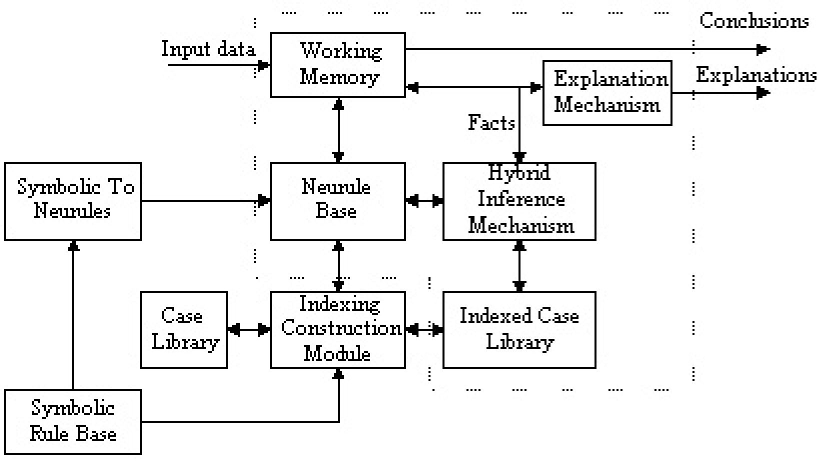

**Fig. 2.** The hybrid architecture

The neurule base contains neurules. These neurules may be produced by conversion from a *symbolic rule base*. This process is described in detail in [12] and is performed by the *symbolic to neurules* module.

Each case is formalized as a set of attribute values. A few of these attributes are used for descriptive purposes, but most of them in performing inferences. Most of the attributes used in making inferences correspond to input, intermediate and output variables of symbolic rules. The neurule base is used to index a *case library*. In this way, an indexed case library is derived. Special care is required when converting an symbolic rule-base (SRB) that already indexes a case library to a neurule base. The process of acquiring an indexing scheme is performed offline by the *indexing construction module* and is described in Section 4.

The hybrid inference mechanism makes inferences combining neurule-based and case-based reasoning. It takes into account the facts contained in the working memory, the neurules in the neurule base and the cases in the indexed case library. The hybrid inference mechanism is described in Section 5.

## 4 Indexing

Indexing concerns the organization of the available cases so that combined neurule-based and case-based reasoning can be performed. The neurules contained in the neurule base are used to index cases representing their exceptions. A case constitutes an exception to a neurule if its attribute values satisfy sufficient conditions of the neurule but the neurule's conclusion contradicts the corresponding attribute value of the case. Exception cases to neurules are considered of most importance as they fill gaps in the knowledge represented by neurules. During inference, exceptions may assist in reaching the right conclusion.

The indexing process may take as input the following two types of knowledge:
(a) Available neurules and cases.
(b) Available symbolic rules and exception cases. This type of knowledge concerns an available formalism of symbolic rules and indexed exception cases as the one presented in [11].

The availability of data determines which type of knowledge is provided as input to the indexing module. The indexing process for each one of these types is briefly presented in the following.

### 4.1 Indexing Process for Available Neurules and Cases

The available neurules must be associated with the cases constituting their exceptions. For each case, this information can be easily acquired as following:

Until all the intermediate and output attribute values of the case have been considered:

1. Perform neurule-based reasoning for the neurules based on the attribute values of the case.
2. If a neurule fires, check whether the value of its conclusion variable matches the corresponding attribute value of the case. If it doesn't, mark the case as an exception to this neurule.

As an example, to demonstrate how the indexing process works, we use neurule N1 presented in Table 1 and the two example cases in Table 2. The cases possess other attributes as well. However, only their most important attributes are shown in Table 2.

**Table 2.** Example cases

| *patient-class* | *pain* | *fever* | *ant-reaction* | *joints-pain* | *disease-type* |
|---|---|---|---|---|---|
| human0-20 | continuous | medium | high | yes | special-arthritis |
| human0-20 | continuous | high | high | no | inflammation |

Disease-type is the attribute of the cases corresponding to the neurule's conclusion. Let $ws_1$ and $rs_1$ be the weighted and remaining sum respectively for the first case. Also let $ws_2$ and $rs_2$ be the weighted and remaining sum respectively for the second case. Evaluation of conditions for the first case (second case) continues until $|ws_1| > |rs_1|$ ($|ws_2| > |rs_2|$). The final values of these sums are the following:

$ws_1 = (-4.2) + 3.0 + 2.8 + 2.7 = 4.3 > 0$,

$rs_1=2.7$,

$ws_2 = (-4.2) + 3.0 + 2.8 - 2.7 + 2.7 = 1.6 > 0$,

$rs_2=0$.

Therefore, the attribute values of both cases give a positive weighted sum for the neurule's conditions. Notice that only the first three conditions of the neurule need to be taken into account for the first case since the weighted sum (4.3) exceeds the remaining sum (i.e., the absolute value of the significance factor of the fourth condition), which equals to 2.7. The fact that the weighted sums were positive means that both cases will be classified in the 'inflammation' disease-type. However, only the disease-type observed for the second case complies with the neurule's conclusion. The first case contradicts the neurule. Therefore, it will be indexed as an exception to the neurule.

### 4.2 Indexing Process for Available Symbolic Rules and Exception Cases

The symbolic rules are converted to neurules using the symbolic to neurules module. The produced neurules are associated with the exception cases of the symbolic rules belonging to their merger sets.

A potential disadvantage of this conversion is the fact that in average a produced neurule will be associated with more exception cases than its merged symbolic rules. This may affect negatively the case-based reasoning part of the inference process. However, the explanation rule produced from the neurule can be used to surpass this deficiency by limiting the exception cases considered by case-based reasoning (see Section 5).

**Table 3.** Symbolic rules of a merger set

| | |
|---|---|
| R2<br>**if** patient-class is human0-20,<br>pain is night,<br>fever is no-fever,<br>ant-reaction is none<br>**then** disease-type is primary-malignant | R3<br>**if** patient-class is human21-35,<br>pain is night,<br>ant-reaction is none<br>**then** disease-type is primary-malignant |
| R4<br>**if** patient-class is human21-35,<br>pain is continuous,<br>fever is no-fever,<br>ant-reaction is none<br>**then** disease-type is primary-malignant | |

As an example, consider the three symbolic rules (R2, R3, R4) presented in Table 3. Table 4 presents some of their exception cases. The symbolic rule for which each of these example cases is an exception, is shown in the 'disease-type' column of the table in parenthesis. Merging these three symbolic rules produces the neurule presented in Table 5. The cases in Table 4 are now indexed as exceptions to this neurule. This example demonstrates how the average number of exception cases indexed by a neurule is increased if it is produced from a merger set of symbolic rules indexing exception cases.

**Table 4.** Exception cases indexed by the symbolic rules in Table 3

| *patient-class* | *pain* | *Fever* | *ant-reaction* | *joints-pain* | *disease-type* |
|---|---|---|---|---|---|
| human0-20 | night | no-fever | none | yes | inflammation (R2) |
| human21-35 | continuous | no-fever | none | no | primary-benign (R4) |
| human21-35 | continuous | no-fever | none | yes | chronic-inflammation (R4) |
| human21-35 | night | no-fever | none | no | arthritis (R3) |

**Table 5.** Neurule produced by merging the symbolic rules in Table 3

```
NR2-R3-R4
(-7.8) if patient-class is human21-35 (6.9),
          pain is night (6.4),
          ant-reaction is none (6.3),
          patient-class is human0-20 (3.0),
          fever is no-fever (2.7),
          pain is continuous (2.6)
       then disease-type is primary-malignant
```

## 5 The Hybrid Inference Mechanism

The inference mechanism combines neurule-based reasoning with case-based reasoning. The combined inference process mainly focuses on the neurules. The exception cases are considered only when sufficient conditions of a neurule are fulfilled so that it can fire. If this is so, firing of the neurule is suspended and case-based reasoning is performed for its exception cases. The results produced by case-based reasoning are evaluated in order to assess whether the neurule will fire or whether the conclusion proposed by the exception case will be considered valid.

Case-based reasoning and evaluation of its results is performed as in [11]. Reasoning tries to find similarities between known data and exception cases. If such similarity is found, an analogical rule encompassing the similar attribute values between the indexed case and the input case is produced. The analogy rules produced by case-based reasoning are evaluated. Evaluation is based on two factors: the similarity degree between the input case and the exception cases and how well the analogy rule works (generalizes) on other exception cases. If the analogy between the input case and an exception case proves to be compelling, the conclusion supported by the case is considered valid, whereas the associated neurule becomes 'blocked'.

The basic steps of the inference process combining neurule-based and case-based reasoning are the following:

1. Perform neurule-based reasoning for the neurules.
2. If enough of the conditions of a neurule are fulfilled so that it can fire, do the following:
   2.1. If the neurule has no associated exception cases, it fires and its conclusion is inserted into the working memory.
   2.2. If the neurule is associated with exception cases suspend its firing. Produce an explanation rule for the neurule and perform case-based reasoning for the neurule's associated exception cases matching the explanation rule.
   2.3. If the analogy proposed by case-based reasoning is compelling, insert the conclusion supported by the exception case into the working memory and mark the neurule as 'blocked'. Otherwise, mark the neurule as 'fired' and insert its conclusion into the working memory.

To explain how conclusions are reached, the explanation mechanism described in [13] is used. If a neurule becomes 'blocked', due to an exception case, the explanation rule produced is the analogy rule yielded by the operation of case-based reasoning for the neurule's exception cases. If this is so, the explanation mechanism informs that an exception was triggered and the exception conclusion was produced instead of the normal one.

We now present a simple example. Suppose that the case contained in Table 6 is given as input to neurule NR2-R3-R4 (see Table 5).

**Table 6.** An input case to neurule NR2-R3-R4

| *patient-class* | *pain* | *fever* | *ant-reaction* | *joints-pain* |
|---|---|---|---|---|
| human21-35 | night | high | none | no |

Only the first three conditions need to be examined. Their weighted sum is (-7.8) + 6.9 + 6.4 + 6.3 = 11.8. This number is greater than the remaining sum of the rest three conditions that equals to 3.0 + 2.7 + 2.6 = 8.3.

The explanation rule (EXR) produced is shown in Table 7. Firing of the neurule is suspended and the exception cases matching the explanation rule are considered in the case-based reasoning process. From the exception cases shown in Table 4, only the fourth one matches the explanation rule.

**Table 7.** Explanation rule produced from neurule NR2-R3-R4

```
EXR
if    patient-class is human21-35,
      pain is night,
      ant-reaction is none
then disease-type is primary-malignant
```

Suppose that the same input case was given to the three symbolic rules shown in Table 3. In the symbolic rule-based reasoning part of the inference, the order that the rules will be considered is R2, R3, and R4. The conditions of rule R2 are not satisfied. However, all conditions of rule R3 are satisfied. Four rule conditions will have to be examined (one condition for R2 and three conditions for R3) instead of the three conditions examined in neurule-based reasoning. Firing of rule R3 is suspended and its indexed exception cases (i.e., the fourth case in Table 4) will be considered by case-based reasoning.

Therefore, both types of inference mechanisms produce the same results. However, the inference mechanism combining neurule-based and case-based reasoning requires the examination of fewer conditions. This feature is intensified in longer inference chains when much more rules have to be examined.

## 6 Experimental Results

To test the effectiveness of our approach we used a symbolic rule base concerning a medical application domain. The symbolic rule base contained fifty-eight (58) symbolic rules acquired by interviewing an expert. The symbolic rules were indexing available exception cases in order to improve their accuracy. This combined symbolic rule base and indexed case library will be referred to as SRCL. By using the symbolic-to-neurules module, the symbolic rules were converted to neurules. In total, thirty-four (34) neurules were produced. The exception cases of the merged symbolic rules were indexed accordingly by the produced neurules. The combined neurule base and indexed case library will be referred to as NRCL.

Inferences were run for SRCL and NRCL. Inferences from SRCL were performed using the inference mechanism combining rule-based and case-based reasoning as described in [11]. Inferences from NRCL were performed according to the inference mechanism integrating neurule-based and case-based reasoning. As expected, inferences produced the same conclusions in both SRCL and NRCL for the same variable-value data. However, inferences from NRCL required the evaluation of fewer conditions than the corresponding inferences from SRCL.

Table 8 presents such experimental results regarding inferences from SRCL and NRCL. It presents results regarding the number of visited rules as well as the number of evaluated conditions. The table also presents if the conclusion was derived as an exception or not (column 'Exception Occurred').

As can be seen from the table, there is an average 45% reduction in the rules visited in NRCL. Furthermore, about 25% fewer conditions were evaluated in inferences from NRCL. Finally, the integration with case-based reasoning improved the accuracy of the inference mechanism in about 30% of the inferences.

## 7 Conclusions

In this paper, we present an approach that integrates neurule-based and case-based reasoning. Neurules are a type of hybrid rules integrating symbolic rules with neurocomputing. In contrast to other neuro-symbolic approaches, neurules retain the naturalness and modularity of symbolic rules. Integration of neurules and cases is done in order to improve the accuracy of the inference mechanism. Neurules are used to index cases representing their exceptions.

The use of neurules instead of symbolic rules as in the approach described in [11] offers a number of advantages. Conclusions from neurules can be reached more efficiently. In addition, neurule-based inference can be performed even if some of the inputs are unknown. This is not possible in symbolic rule-based reasoning. Even an existing formalism of symbolic rules and indexed exception cases can be converted to a formalism of neurules and indexed exception cases.

The presented approach integrates three types of knowledge representation schemes: symbolic rules, neural networks and case-based reasoning. Most hybrid intelligent systems implemented in the past usually integrate two intelligent technologies e.g. neural networks and expert systems [4], neural and fuzzy logic [19], genetic algorithms and neural networks, etc. A new development that should receive interest in the future is the integration of more than two intelligent technologies that can facilitate the solution of complex problems and exploit multiple types of available data sources.

**Table 8.** Experimental Results

| *Inference No* | *Rules Visited* | *Conditions Evaluated* | *Exception Occurred* |
|---|---|---|---|
| | SRCL / NRCL | SRCL / NRCL | |
| 1 | 8 / 4 | 17 / 10 | No |
| 2 | 11 / 6 | 26 / 17 | Yes |
| 3 | 13 / 8 | 26 / 21 | No |
| 4 | 17 / 10 | 33 / 27 | No |
| 5 | 21 / 10 | 32 / 22 | No |
| 6 | 24 / 11 | 43 / 32 | No |
| 7 | 26 / 13 | 48 / 39 | Yes |
| 8 | 29 / 15 | 57 / 44 | Yes |
| 9 | 32 / 16 | 58 / 43 | Yes |
| 10 | 40 / 22 | 59 / 51 | No |
| 11 | 42 / 23 | 77 / 59 | No |
| 12 | 44 / 24 | 81 / 63 | No |
| 13 | 47 / 26 | 86 / 65 | No |
| **Total** | **354 / 188** | **643 / 493** | |

**Acknowledgements**

This work is supported by the Research Committee of the University of Patras, Program Caratheodoris-2001, project No 2788.

## References

1. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches. Artificial Intelligence Communications 7 (1994) 39-59.
2. Aha, D., Daniels, J.J. (eds.): Case-Based Reasoning Integrations: Papers from the 1998 AAAI Workshop. Technical Report WS-98-15. AAAI Press (1998).
3. Althoff, K., Wess, S., Traphoner, R.: INRECA-A Seamless Integration of Induction and Case-Based Reasoning for Decision Support Tasks. Proceedings of the Eighth Workshop German SIG on Machine Learning (1995).
4. Boutsinas, B., Vrahatis, M., N.: Artificial nonmonotonic neural networks. AI 132 (2001) 1-38.
5. Branting, L.K.: Building Explanations from Rules and Structured Cases. International Journal of Man-Machine Studies 34 (1991) 797-837.
6. Branting, L.K.: Reasoning with Rules and Precedents. Kluwer Academic Publishers. Dordrecht (1999).
7. Cercone, N., An, A., Chan, C.: Rule-Induction and Case-Based Reasoning: Hybrid Architectures Appear Advantageous. IEEE Transactions on Knowledge and Data Engineering 11 (1999) 164-174.
8. Domingos, P.: Unifying Instance-Based and Rule-Based Induction. Machine Learning 24 (1996) 144-168.
9. Gallant, S. I.: Neural Network Learning and Expert Systems. MIT Press (1993).
10. Ghalwash, A. Z.: A Recency Inference Engine for Connectionist Knowledge Bases: Applied Intelligence 9 (1998) 201-215.
11. Golding, A.R., Rosenbloom, P.S.: Improving accuracy by combining rule-based and case-based reasoning. Artificial Intelligence 87 (1996) 215-254.
12. Hatzilygeroudis, I., Prentzas, J.: Neurules: Improving the Performance of Symbolic Rules. International Journal on AI Tools 9 (2000) 113-130.
13. Hatzilygeroudis, I., Prentzas, J.: An Efficient Hybrid Rule-Based Inference Engine with Explanation Capability. Proceedings of the 14th International FLAIRS Conference. AAAI Press (2001) 227-231.
14. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann Publishers, San Mateo, CA (1993).
15. Koton, P.: Reasoning about Evidence in Causal Explanations. Proceedings of the AAAI-88. AAAI Press (1988).
16. Leake, D.B.: Combining Rules and Cases to Learn Case Adaptation. Proceedings of the 17th Annual Conference of the Cognitive Science Society (1995).
17. Leake, D.B. (ed.): Case-Based Reasoning: Experiences, Lessons & Future Directions. AAAI Press/MIT Press (1996).
18. Marling, C.R., Petot, G.J., Sterling, L.S.: Integrating Case-Based and Rule-Based Reasoning to Meet Multiple Design Constraints. Computational Intelligence 15 (1999) 308-332.

19. Neagu, C.-D., Palade, V.: Modular Neuro-Fuzzy Networks Used in Explicit and Implicit Knowledge Integration, Proceedings of the 15th International FLAIRS Conference (FLAIRS-02). AAAI Press (2002) 277-281.
20. Rissland, E.L., Skalak, D.B.: CABARET: Rule Interpretation in a Hybrid Architecture. International Journal of Man-Machine Studies 34 (1991) 839-887.
21. Surma, J., Vanhoof, K.: An Empirical Study on Combining Instance-Based and Rule-Based Classifiers. Case-Based Reasoning Research and Development, First International Conference, ICCBR-95, Proceedings. Lecture Notes in Computer Science, Vol. 1010. Springer-Verlag, Berlin Heidelberg New York (1995).
22. Veloso, M.M.: Learning by Analogical Reasoning in General Problem Solving. PhD Thesis, Technical Report CMU-CS-92-174, Carnegie Mellon University, Pittsburg, PA (1992).