Published in the International Journal of Hybrid Intelligent Systems 1(3-4) (2004) 111-126

Neuro-Symbolic Approaches for Knowledge Representation in Expert Systems

Ioannis Hatzilygeroudis and Jim Prentzas*

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

Abstract. In this paper, we first present and compare existing categorization schemes for neuro-symbolic approaches. We then stress the point that not all hybrid neuro-symbolic approaches can be accommodated by existing categories. Such a case is rule-based neuro-symbolic approaches that propose a unified knowledge representation scheme suitable for use in expert systems. That kind of integrated schemes have the two component approaches tightly and indistinguishably integrated, offer an interactive inference engine and can provide explanations. Therefore, we introduce a new category of neuro-symbolic integrations, namely 'representational integrations'. Furthermore, two sub-categories of representational integrations are distinguished, based on which of the two component approaches of the integrations is given pre-eminence. Representative approaches as well as advantages and disadvantages of both sub-categories are discussed.

Keywords: Neuro-symbolic integrations, Rule-based expert systems, Connectionist expert systems

1 Introduction

The integration of different knowledge representation methods is a very active research area in Artificial Intelligence. The aim is to create hybrid formalisms that benefit from each of their components. It is generally believed that complex problems can be easier solved with hybrid systems.

One of the most popular types of integration is the combination of symbolic and connectionist approaches. For example, efforts to combine symbolic rules and neural networks have yielded advanced knowledge representation formalisms (Bookman and Sun 1993, Fu 1994, Medsker 1994 and 1995; Hilario 1997, Sun and Alexandre 1997, McGarry et al. 1999; Wermter and Sun 2000, Cloete and Zurada 2000, Garcez et al. 2002). The success of these hybrid methods is based on the fact that the two integrated formalisms have complementary advantages and disadvantages.

Symbolic rules offer a number of advantages for knowledge representation such as, naturalness, modularity and ease of explanation. Expert systems are the most popular rule-based applications. They provide an interactive inference mechanism, which guides the user in supplying input values, and an explanation mechanism, which justifies the reached conclusions. However, symbolic rules have also some deficiencies. The most important disadvantage is the difficulty in acquiring rules, a problem known as the knowledge acquisition bottleneck (Gonzalez and Dankel 1993).

Neural networks represent a totally different approach to problem solving, known as connectionism (e.g. Gallant 1993). Some advantages of neural networks are their ability to obtain their knowledge from training examples (reducing the interaction with the experts), their high level of efficiency and their ability to represent complex and imprecise knowledge. Main drawbacks, compared to symbolic rules, are the lack of naturalness and modularity and the difficulty (if not impossibility) in providing explanations.

The integration of symbolic rules and neural networks can result into various neuro-symbolic representations. Most of them give pre-eminence to the connectionist approach, hence do not provide the functionalities required by an expert system, like interactive inference and generation of explanations.

* The order is alphabetical

Various categorization schemes for neuro-symbolic approaches have been recently presented (Medsker 1994, Hilario 1997, McGarry et al. 1999). Due to the richness and the variety of integration methods, not all hybrid approaches can be fully accommodated by existing categorization schemes. Such a case involves certain hybrid approaches that offer a unified neuro-symbolic knowledge representation scheme, which provides the basic functions of expert systems. This paper focuses on these approaches. Two categories of such approaches are distinguished: one giving pre-eminence to connectionist and one giving pre-eminence to the symbolic framework. The systems of the second category are proven to be more advantageous than the systems of the first category, as far as expert systems functionalities are concerned. Those two categories constitute a new more general category of integrated systems, called 'representational integrations'.

This paper is organized as follows. Section 2 discusses background knowledge focusing on the advantages and disadvantages of symbolic rules and neural networks. In Section 3, a critical overview of existing categorization schemes is made. Section 4 presents rule-based neuro-symbolic integrations for knowledge representation in expert systems, which do not exactly fit into the existing categories, and introduces a new category. Finally, section 5 concludes the paper.

2 Knowledge Representation and Expert Systems

2.1 Characteristics of Expert Systems

A knowledge representation (KR) scheme (or formalism or language) is the basis of any expert system (ES). There is no ES without a KR scheme, which is used to represent the knowledge involved in the ES and perform inferences. We can distinguish, from a technical point of view, two main aspects of any KR scheme, its *syntax* and its *inference mechanism* (Reichgelt 1991). The syntax (or notation) of a KR scheme refers to the explicit way it expresses knowledge (or information). There are various forms of syntax, ranging from symbol or text based forms (e.g. logic-based formalisms) to diagrammatic forms (e.g. semantic nets). Of course, the syntax of a KR scheme is accompanied by some *semantics*, which gives meaning to the expressions of the scheme. The explicitly expressed knowledge constitutes the *knowledge base* (KB) of the ES. The inference mechanism of a KR scheme refers to the way it derives knowledge, i.e. makes explicit knowledge which is implicit in the KB.

ESs are typically used for problem solving, by imitating the way a human expert does it (Jackson 1999). The main parts of a typical expert system are illustrated in Figure 1. The *inference engine* (IE), which implements the inference mechanism of the employed KR scheme, uses the knowledge contained in the *knowledge base* (KB) as well as any known data (e.g. facts) concerning the problem at hand. The known data, which is either initial data supplied to the system by the user at the beginning of an inference process or intermediate/final conclusions reached by the system, is stored into the *working memory*. The inference engine initially takes a few (or even none) input values from the user. When known data does not suffice for drawing conclusions, IE focuses on unknown inputs that seem to be most relevant (or important) to the inference process at hand and queries the user to supply further input data. The *explanation mechanism* provides explanations regarding the conclusions reached by the IE.

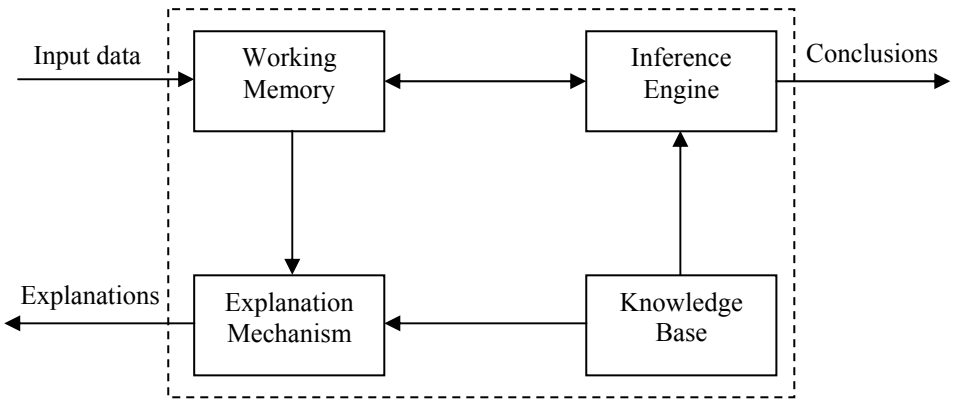


Figure 1. The basic structure of an expert system

An expert system has a number of characteristics that distinguish it from other intelligent systems used for problem solving:

- *Separation of knowledge from its use*. KB represents the domain knowledge. IE is separate from KB. IE implements the way knowledge is used and is domain independent. This independency leads to a *modular structure* (see Fig. 1).
- *Interactive inferences*. The system halts at certain points during inferences to ask for input data from the user. This means that not all available input data may be needed during an inference. This also leads to an interactive and efficient way of acquiring data from the user.
- *Provision of explanations*. An expert system should be able to explain at least how its conclusions have been reached.

These characteristics could be considered as requirements for an intelligent system to be an expert system.

2.2 Symbolic Rules

The most popular KR scheme used in ESs is *production* or *symbolic rules* (Buchanan and Shortliffe 1984). The popularity of symbolic rules mainly stems from their naturalness, which facilitates comprehension of the encompassed knowledge. The basic syntax of a rule is the following:

if <conditions>
then <conclusion>

where <conditions> represents a number of conditions and <conclusion> represents the conclusion that will be derived when the conditions are satisfied. The conditions are combined with one or more of the logical operators 'and', 'or' and 'not'. The conclusion of a rule can be reached when the logical function connecting its conditions results to true. When it happens, the rule is said to fire.

Reasoning with rules is based on handling symbols, which represent concepts. Therefore, inference is based on the so-called *symbolic computation*. There are two main inference methods: *backward chaining* and *forward chaining*. The former is guided by the goals, whereas the latter by the data.

Symbolic rules, as a knowledge representation formalism, have several advantages as well as some significant disadvantages (Reichgelt 1991, Gonzalez and Dankel 1993, McGarry et al. 1999).

The main advantages of rules are:

- *Naturalness*. Rules are a simple knowledge representation method with a high level of comprehensibility. It is easy to comprehend the knowledge encompassed in a rule. Rules emulate the expert's way of thinking in a natural way.
- *Modularity*. Each rule is a discrete, autonomous knowledge unit that can be easily inserted in or removed from a knowledge base, without requiring any other change. This greatly facilitates incremental development of a rule base.
- *Provision of explanations*. Rules can easily provide explanations for the reached conclusions. A simple backward tracing of the fired rules involved in the solution may give a sufficient form of explanation. This feature of symbolic rules is a direct consequent of their naturalness and modularity.
- *Knowledge interoperability*. Naturalness and modularity of rules enable transfer of knowledge between systems used in closely related application domains.

The main disadvantages of rules are:

- *Knowledge acquisition bottleneck*. The standard way of acquiring rules through interviews with experts is cumbersome. The main reason for this is the inability of the expert to express his/her knowledge. Therefore, the acquired knowledge may not be complete or even correct. An alternative way to acquire knowledge in the form of symbolic rules is the use of machine learning techniques, like decision trees. Those methods produce rules from existing training examples. However, it is not certain that the available set of examples covers the whole domain (e.g. exception situations) and thus the produced rule set may not be complete.
- *Brittleness of rules*. It is not possible to draw conclusions from rules when there are missing values in their input data (conditions). In addition, rules do not perform well in cases of unexpected input values or combinations of values.
- *Inference efficiency*. In certain cases, the performance of the inference engine is not the desired one, especially in very large rule bases. Rule-based systems may face the scalability problem in inference.

- *Difficulty in maintenance of large rule bases*. The maintanace of rule bases is a difficult process as the size of the rule base increases. The rule base may encompass problematic rules, such as redundant rules, conflicting rules, rules with redundant or missing conditions, missing rules required in the inference process. In order to deal with such problems, complex verification and validation methods are required.
- *Empirical knowledge is not exploited*. In several application domains there are available datasets with examples of solved problems. This available data cannot be taken into consideration by rule-based systems. However, they can contribute decisively in the inference process as they may represent special cases or exceptions not included in rules.

2.3 Artificial Neural Networks

An artificial neural network (or simply *neural net*) is a parallel and distributed structure (see Fig. 2). A neural net consists of a number of interconnected nodes, called *neurons*. There are *weights* attached to the *connections* between neurons: each connection from a neuron u_j to a neuron u_i is associated with a numerical weight $w_{i,j}$, which represents the influence of u_j to u_i. Each neuron has also a weight attached to itself, called the *bias*. Each neuron acts as a local processor, which computes its *output* (connection) (u_i) based on the weighted sum of (the values of) its *input connections* (u_1, u_2, …, u_n) and an *activation function f* (see Fig. 3). The activation function may be of various types, e.g. a threshold or a sigmoid function. The connection weights and the structure of a neural net define its behavior.

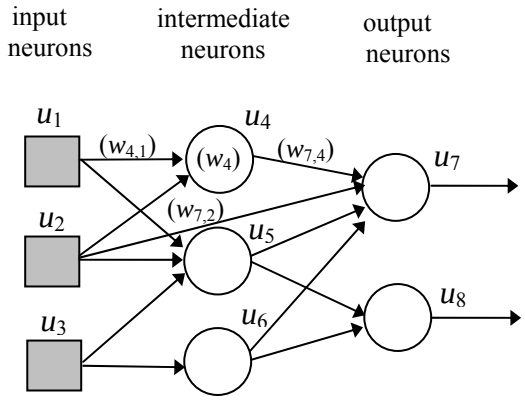
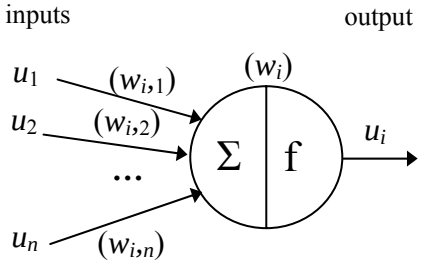


Figure 2. A feedforward neural net



$$a_i = w_i + \sum_{j=1}^{n} w_{j,i} u_{j,i}$$

$$u_i = f(a_i)$$

Figure 3. The computational model of a neuron

The most popular neural net class is feedforward nets, which are nets that do not contain cycles. They are usually organized in layers. So, we distinguish between the input layer, intermediate layer(s) and output layer (see Fig. 2). Input layer consists of input neurons (illustrated as rectangles in Fig. 3), which are pseudo-neurons, are used to transfer externally given values to neurons at further layer(s), do not perform any computation and are taken as the inputs of the network. The outputs of the neurons at the output layer are

taken as the outputs of the network. Intermediate neurons are used for intermediate computations and are often called *hidden* neurons.

A neural net can store empirical knowledge and serve as the knowledge base of a classification expert system. Empirical knowledge comes in the form of training examples. Each example consists of input values and the corresponding correct output. They are used to 'train' the net, i.e. to calculate the weights so that, the training examples are correctly classified (i.e. the combination of the inputs in each example produces the specified output). This is called the *supervised learning* model (the other is called the *unsupervised learning model*, where no correct outputs are specified). There are several training algorithms for supervised learning. A well known such training algorithm is *back-propagation*. Thus, neural nets are representatives of empirical machine learning systems. Empirical learning usually requires a large, but possibly incomplete, training set from which they can generalize. They may also need some domain knowledge such as information regarding the most relevant features of the training examples as well as the values they can take.

Knowledge can be represented in a neural net via its topology and its weights, if some semantics is attached to neurons and the activation values. For example, semantics may include associations between concepts of the problem and neurons of the network. We can say that the syntax of a neural net is of diagrammatic type. Inference in a neural net is not of symbolic nature, as in symbolic rules, but of numerical nature, and constitutes in the computation of its output(s). Hence, inference in neural nets is based on *numeric* or *sub-symbolic computation*.

As symbolic rules, neural nets also have a number of advantages as well as a number of disadvantages (Reichgelt 1991, Gonzalez and Dankel 1993, McGarry et al. 1999). The main advantages of neural networks are:

- *Ability to learn from training examples*. Neural networks can learn the knowledge contained in training examples, which can be easily available in several applications. They transform the knowledge in the examples into a compact form of a network topology and corresponding weights.
- *High performance level*. The output of a neural net is quite efficiently computed, since it is based on numerical calculations (soft computing).
- *Ability to generalize*. Neural nets may compute the correct output from input values combinations not present in the training set. Neural networks generalize better than other methods of empirical learning.
- *Robust output computation*. A neural net can compute its output(s) even when there is missing or noisy input data.

The disadvantages of neural nets come from two sources: the fact that they are empirical learning systems and their peculiar nature. The disadvantages that are common to other methods of empirical learning as well are:

- *Incomplete or unavailable training set*. The training set may not represent the whole domain (e.g. certain exceptions). In certain applications there may not be an available set of training examples.
- *Difficulty in feature selection*. The features/attributes used in the examples must be carefully chosen. Certain domain knowledge is required in order to discern the relevant from the irrelevant features. The existence of irrelevant features may negatively affect the learning process.
- *Existing rule bases cannot be directly exploited*. Available domain knowledge in the form of symbolic rules cannot not be exploited in a direct way.

Disadvantages due to the nature of neural networks are:

- *Training time and convergence problems*. The required training time may be extensive and convergence to an acceptable solution is not always assured.
- *Initialization problems*. The initialization of the weights may play an important role in the training process leading to different solutions. Usually, weights are initialized to random numbers belonging to small intervals. However, there is no way of making an initialization valid for all applications.
- *Topology design problems*. Determination of the neural network topology (such as finding the required number of hidden nodes) is done empirically (on a trial-and-error basis). There is no way of choosing a good topology for a neural network regardless of the application.
- *Black box semantics*. It is difficult to comprehend the knowledge encompassed in a neural network. It is difficult to associate the weights and the nodes of the neural network with specific domain concepts due to the fact that the knowledge of the training examples has been distributed over the whole network. Therefore, a neural network cannot be decomposed into components and form a modular structure. So, incremental development of a neural knowledge base is rather impossible. A further negative consequence of the black box semantics of neural nets is the difficulty in transferring the knowledge of a trained neural net to other related application domains.

- *Explanations cannot be provided*. Due to the above, provision of explanations for the computed output is almost impossible. In some applications, provision of explanations may not be required, but in others it is a prerequisite. The comprehension of the knowledge contained in neural networks can be achieved by rule extraction methods (Andrews et al. 1995, Palade et al. 2001). However, the extracted rules may not faithfully represent the behavior of the neural net.

3 Categorizing Neuro-symbolic Integrations: A critical overview

The frequent use of rules and neural networks for the development of intelligent systems as well as the fact that their advantages and disadvantages are complementary led to the development of hybrid systems integrating both approaches. Most of those approaches have successfully been applied to practical applications. Moreover, several hybrid approaches constitute general methodologies that can be applied to various application domains. Therefore, a systematic categorization of systems/approaches integrating rules and neural nets would be of great value for system designers.

There has been more than one effort to categorize approaches integrating symbolic rules and neural nets. Those efforts attempt to specify the particular characteristics of the hybrid systems such as, the types of the tasks they perform and the degree of interconnection-integration between the different components. Given the great number of hybrid systems developed, it is not an easy task to specify all the characteristics of the hybrid systems from all points of view. A first, rather simple, categorization is that of Medsker (Medsker 1995). Two more systematic categorization schemes are the ones presented in (Hilario 1997) and (McGarry et al. 1999).

Medsker's scheme categorizes the hybrid systems based on the *interconnection degree* between the component approaches (neural networks and expert systems) without taking into consideration other parameters. Five categories of hybrid systems are specified in the scheme: *standalone*, *transformational*, *loosely coupled*, *tightly coupled*, *fully integrated*. From those five categories, only the last three describe actually hybrid systems. In the case of standalone systems, there is no substantial hybridism, given that the different components are discrete and without any interaction between them. Also, the transformational model does not refer to hybrid systems, as it merely examines the most efficient implementation method through duplication, that is, by constructing a neural network and a rule-based system. The remaining three categories include systems in which there is an interaction between the incorporated components. In the loosely coupled systems, communication between the different components is performed by using shared files, in the tightly coupled models by using shared memory structures and in fully integrated models by using shared memory structures and knowledge representations.

In the categorization scheme proposed by Hilario (Fig. 4) there are two basic categories of integrated systems: the *unified* and the *hybrid* approach. The unified approach assumes that no symbolic structures and processes are needed, considering that all symbolic functions can be implemented by neural structures and functions. There are two basic trends in the unified approach: the *Neuronal Symbol Processing* and the *Connectionist (or Neural) Symbol Processing*.

The first trend is more related to the real biological neurons (that's where the term neuronal came from) and stems from the assumption that all cognitive processes can be explained via biological terms. Essentially, it follows a bottom-up approach starting from the biological neuron. The specific trend does not seem to be quite mature yet.

The second trend is not directly related to biology and uses artificial neural networks for the implementation of complex symbolic processes. This specific trend has given rise to important results in logic and automated reasoning (e.g. Touretzky and Hinton 1988, Dolan and Smolensky 1989, Samad 1992, Mani and Shastri 1993, Sun 1994, Ajjanagadde and Shastri 1995). An important problem that had to be dealt with is variable binding. There are three discrete categories of this trend: *localist, distributed* and *combined localist-distributed*. In the localist approaches, there is a one-to-one correspondence between each node of the neural network and the symbolic concepts. The main disadvantage of this approach is that the network size increases as the number of symbolic concepts increases. The distributed approach, on the contrary, stores knowledge to a number of nodes and remedies several deficiencies of the localist approach. Finally, the third approach (i.e. combined localist-distributed) attempts to combine the advantages of the localist and distributed approaches.

The hybrid approach is discerned into two subcategories: *translational* approach and *functional* approach. The translational approach is based on neural networks, which have been derived by combining domain knowledge (i.e. symbolic rules or automata) and training examples. In this approach, the domain knowledge is initially transformed into a neural network, which is then trained by using training examples.

The final network contains a refined version of the domain knowledge (Fu 1993, Mahoney and Mooney 1993, Towell and Shavlik 1994, Omlin and Giles 1996). The initial domain knowledge can be in various forms such as propositional rules (Towell and Shavlik 1994), certainty factor rules (Fu 1993), automata (Omlin and Giles 1996), etc. Well-known representatives of this approach are the so-called *knowledge-based neural networks* (KBNNs) (Fu 1993, Towell and Shavlik 1994). The use of domain knowledge for the creation of the initial neural network offers some advantages compared to classical neural networks. On the one hand, the training process becomes easier, because most of the nodes and the connections are defined from the beginning, weights are initialized properly and smaller training sets are required compared to classical neural networks. On the other hand, the final neural network is sparser compared to classical neural networks and surpasses them in naturalness, since most nodes correspond to symbolic concepts of the domain. Furthermore, this approach deals with problems found in rule-based systems, as far as completeness and correctness are concerned. Therefore, the final neural network is an indirect solution to the knowledge acquisition bottleneck. Compared to symbolic rules, however, it is inferior as far as naturalness is concerned (to a lesser or greater degree), because it gives preeminence to the neural component. In some cases, the domain knowledge incorporated into the final neural network is extracted from it, in order to better comprehend the encompassed knowledge (Towell and Shavlik, 1993, Giles and Omlin 1993).

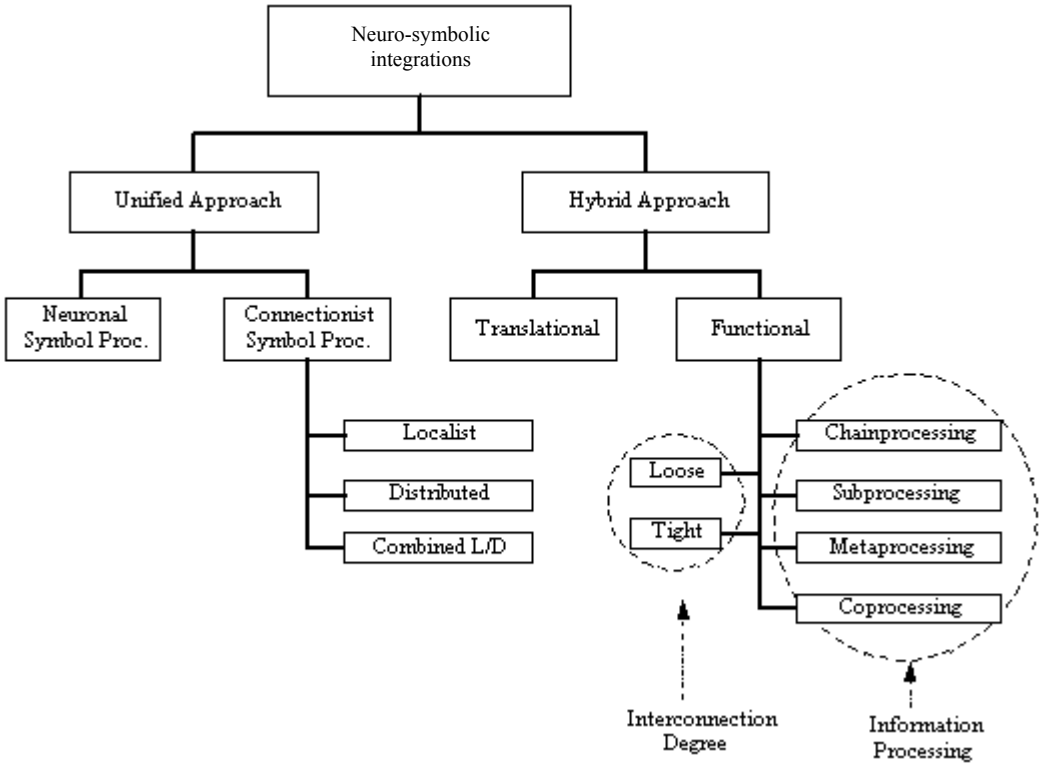


Figure 4. Hilario's categorization scheme

In functional approaches, rules and neural networks constitute discrete components of the hybrid system. They fully encompass functions of symbolic rules and neural networks, given that there is a cooperation and interaction between the discrete components. Functional approaches can be distinguished into further categories based on two parameters: interrelation degree and information flow.

According to the *interconnection degree*, there can be two types of functional approaches: the ones having a loose interconnection and the ones having a tight interconnection. In case of loose interconnection, each component works individually at a local level and the synergy of the components is accomplished by transferring data from the one to the other component. In case of the tight interconnection, there is no data transfer between the two components, as they use common internal structures.

According to *information flow*, there are four types of functional approaches: *chainprocessing*, *subprocessing*, *metaprocessing*, *coprocessing*. In the case of the chainprocessing approach there is a serial processing of information between the components, given that information is sequentially processed by each component. In the case of the subprocessing approach, one component having a secondary role is embedded into the other having a primary role. In the metaprocessing approach, the one component forms the basis for solving the problems and the other plays a metalevel role (e.g. surveillance, control). Finally, in the

coprocessing approach, the components have an equal status and interact between each other to solve problems.

The categorization model of McGarry et al. (Fig. 5) resembles Hilario's model in a large degree. It distinguishes hybrid approaches into three basic categories: *unified*, *transformational*, *modular*. The unified approach is the same as the unified approach of Hilario, however, without any further distinction. Essentially, the unified approach of McGarry et al. is the Connectionist (or Neural) Symbol Processing of Hilario. Moreover, compared to Hilario's model, there is a rather indirect reference to the distinction of the unified approach to localist, distributed and combined localist-distributed.

The transformational approach corresponds to Hilario's translational approach, whereas the modular approach to the functional one. In the model of McGarry et al., the transformational and modular approaches are not grouped into the general hybrid approach, as in Hilario's model, based on the logical argument that the term "hybrid approach" does not indicate anything important, since all three main categories (unified, transformational/translational, modular/functional) represent hybrid systems.

The most remarkable differences between the two categorization schemes are identified in the modular and functional approaches, as far as their decomposition into subcategories is concerned. Based on the information flow/processing, the systems following the modular approach can be distinguished into those performing *sequential processing* (corresponding to Hilario's chainprocessing subcategory) and those performing *parallel processing*. The parallel processing subcategory generally includes the other three subcategories specified by Hilario (subprocessing, metaprocessing, coprocessing).

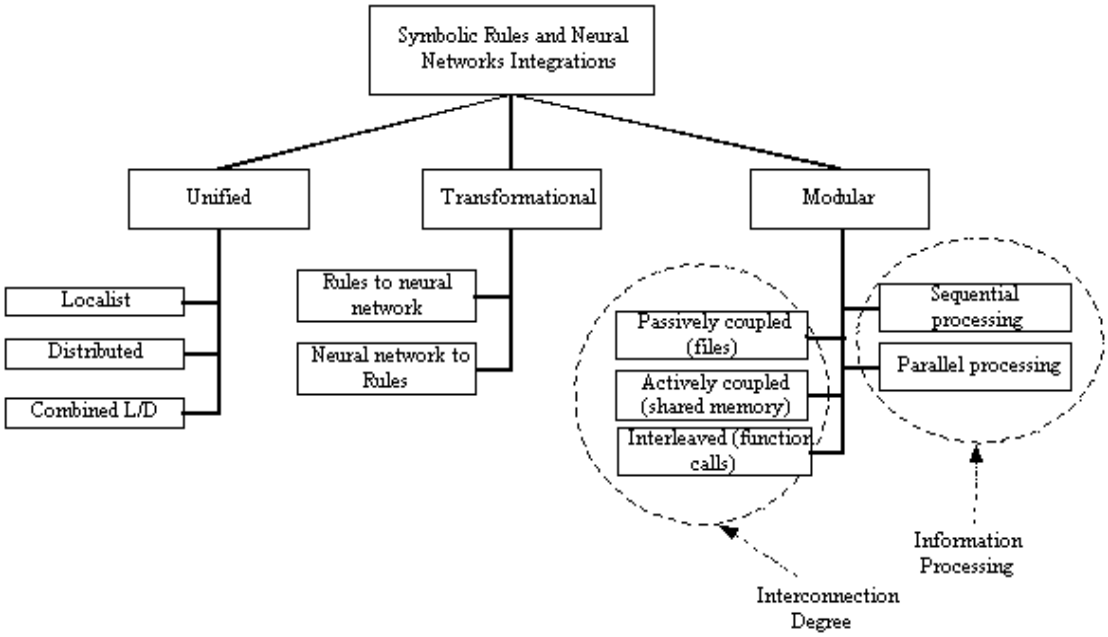


Figure 5. Categorization scheme of McGarry et al.

According to the interconnection degree, three subcategories of the modular approach are distinguished: *passively coupled, actively coupled, interleaved*. These three subcategories correspond to the following three categories of Medsker's categorization scheme respectively: loosely coupled, tightly coupled and fully integrated systems. In the case of the first subcategory (passively coupled), there is a loose connection between the components via the use of shared files. Technically, this subcategory corresponds to the subcategory 'loose interconnection' mentioned by Hilario. In the second subcategory (actively coupled) there is a tighter interconnection between the components with the use of shared memory and a higher synchronization level between the components. Finally, in the third subcategory (interleaved) there is a high level interaction between the components through function calls and by using complex communication protocols.

It should be mentioned that another difference between the two categorization schemes is that Hilario's scheme refers to neuro-symbolic approaches in general, whereas the one of McGarry et al. focuses on hybrid rule-based approaches.

4 Rule-Based Neuro-Symbolic Approaches for KR in ESs

4.1 Introducing a new category

Most neuro-symbolic approaches that the above categorizations are dealt with do not concern systems or schemes that aim at knowledge representation for ESs. Therefore, those integrations do not satisfy the requirements of an expert system, outlined in Section 2.1. More specifically, in those systems inference is performed as in neural networks. This means that the user supplies the values of all inputs (known or unknown, relevant or irrelevant) before the inference process begins and then the network output is computed. Also, there is no interaction with the user during inferences. Moreover, no explanations are provided to justify output.

There are, however, a few neuro-symbolic approaches that offer a unified representation scheme, which provides a unified interactive inference mechanism and an explanation mechanism, in the same way as knowledge representation and reasoning paradigms used in classical expert systems do. In such schemes, the two component approaches are so tightly integrated that are almost indistinguishable. We focus here on integrated approaches of the above type, which integrate symbolic rules (of propositional type) and neural networks. Such approaches are the so-called *connectionist expert systems*, in the sense they are defined in (Gallant 1993), such as (Gallant 1988, Ghalwash 1998, Sima 1995; Sima and Cervenka 2000). Also, another more recent such approach is *neurules* (Hatzilygeroudis and Prentzas 2000 and 2001).

The above approaches cannot be fully accommodated by anyone of the categories presented in Section 3. They do not fit into the functional subcategory of Hilario's categorization scheme or into the modular subcategory of the categorization scheme of McGarry et al. Those categories are related to integrated approaches that include distinct symbolic and neural components, as far as structures as well as processors (e.g. reasoners) are concerned. This is not exactly the case for the aforementioned neuro-symbolic approaches. They incorporate a common hybrid knowledge base as well as inference and explanation mechanisms.

Some of the above approaches bare resemblance to the translational subcategory of Hilario's scheme or the transformational subcategory of the scheme of McGarry et al. For instance, the approach presented in (Gallant 1988 and 1993) can be considered a forerunner of approaches belonging to this category, such as KBNNs. This is so, because it combines some kind of domain knowledge (called dependency information) and training examples for the construction of the knowledge base. Furthermore, neurules can be constructed by transformation from symbolic rules, leading however to an equivalent (and not a refined) knowledge base. This transformation results in more efficient inferences and compact forms of knowledge. Additionally, neurules as well as connectionist expert systems possess interactive inference and explanation mechanisms, not present in the approaches belonging to the translational/transformational subcategory. It must also be mentioned that the approach presented in (Sima 1995, Sima and Cervenka 2000) bares no resemblance to the translational/transformational subcategory, since for the construction of the knowledge base only training examples are used and not domain knowledge as in the case of (Gallant 1988 and 1993).

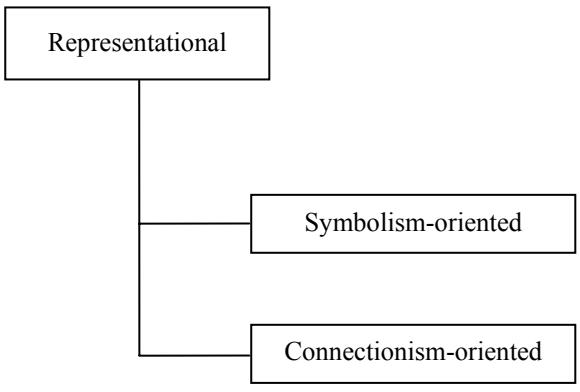


Figure 6. Representational integrations sub-categories

Connectionist expert systems could be considered as belonging to the 'unified' category, more specifically to the 'localist' subcategory, of the categorization schemes presented in Section 3. However,

they incorporate features that seem to make them different from unified approaches. To be more specific, connectionist expert systems do not deal with variable binding and can provide explanations, in contrast to unified approaches.

The difficulty to fit the above mentioned integrated approaches into some category of the existing categorization schemes becomes quite explicit in the case of connectionist expert systems from the following fact. Hilario technically considers this approach as belonging to the unified category. McGarry et al. classify them into the transformational subcategory. Finally, Medsker considers them as a variation of the fully integrated model.

From the above analysis, it is clear that a high level category is missing from the existing categorization hierarchies. Therefore, we introduce such a category, which we call *representational integrations* (or *approaches*). This category is to be placed alongside translational and functional categories in Hilario's scheme or alongside the unified, transformational and modular categories of McGarry, Wermter and MacIntyre's scheme. The subcategories of this new category are presented in Fig. 6 and explained in the following.

A parameter that is of interest in representational integrations is which component approach is given *pre-eminence* to. There are some neuro-symbolic integrations that give pre-eminence to the symbolic framework and some others that give pre-eminence to the connectionist framework. This concerns both aspects of KR and plays a central role on which of the advantages of the two component approaches can be retained in the integrated scheme. For example, if pre-eminence is given to the connectionist framework, naturalness and modularity are difficult to retain in some acceptable degree. Or, if pre-eminence is given to the symbolic framework, generalization capabilities may be reduced. The matter of 'pre-eminence' is generally important not only for neuro-symbolic, but also for other categories of integrations of such kind, such as symbolic-symbolic, neuro-fuzzy etc ones. For example, if we want to integrate logic and frames, then, if we want to retain formal semantics and soundness we should give pre-eminence to the logical framework, which means to incorporate frames within the logical framework or express frames in a logical way (e.g. Horrocks et al. 1999). If we want to retain the flexibility of the frame-based structure of knowledge, then we should incorporate logic into frames, but then formal semantics are partially lost (e.g. Hatzilygeroudis 1996).

Another characteristic of representational integrations is that integration concerns both aspects of the KR scheme, syntax and inference mechanism. One thing that may vary is the degree of integration in the two aspects.

In the sequel, we present representative approaches of the 'representational' category, focusing on approaches that integrate symbolic rules (of propositional type) and neural networks (as mentioned above too).

4.2 Connectionist expert systems: Giving pre-eminence to connectionism

4.2.1 Matrix Controlled Inference Engine (MACIE)

Gallant was the first to present an interactive inference engine and an explanation mechanism for neural networks with discrete nodes. *Connectionist expert systems* (Gallant 1988, 1993) are an approach retaining the basic functions of expert systems, having a neural network with discrete nodes as its knowledge base. To construct the initial neural network, domain concepts (input, intermediate and output) are assigned to network nodes and dependency information regarding the concepts is used to define their connections. One can consider that each node in Gallant's network corresponds to a symbolic rule. Consequently, the neural network is trained using an improved variation of the perceptron learning (i.e. the pocket learning algorithm). In case of inseparability (i.e. non-linear set of training patterns), random cells are inserted into the network. The introduction of those cells has a negative effect on the naturalness of the connectionist knowledge base, because they are meaningless.

The approach followed for the creation of the neural network is a forerunner of KBNNs. Compared to knowledge-based neural networks, the domain knowledge used for the construction of the initial neural network consists of concepts and dependency information, instead of rules, and also no weight initialization is required. The approach presented in (Gallant 1988, 1993) has been applied to a medical domain.

Gallant proposed an inference engine called MACIE (MAtrix Controlled Inference Engine). Its characteristic features are the ability to reach conclusions from partially known inputs, the interaction with the user, in order to provide input values, and the ability to focus on specific unknown network nodes, which are assumed to be most important to reach conclusions. Furthermore, MACIE includes an explanation mechanism.

MACIE combines backward and forward chaining with neural computing. It is based on the fact that for the computation of a node's output, not all of its inputs need to be known. For this reason, two sums are

calculated for a node u_i, the contribution of the known inputs ($KNOWN_i$) and the maximum contribution of the unknown inputs ($MAX\_UNKNOWN_i$) according to the following formulas:

$$KNOWN_i = \sum_{j:u_j\ known} w_{i,j} u_j \qquad (1)$$

$$MAX\_UNKNOWN_i = \sum_{k:u_k\ unknown} |w_{i,k}| \qquad (2)$$

Whenever $|KNOWN_i| > MAX\_UNKNOWN_i$, the output of node u_i can be computed, since the remaining unknown inputs cannot change the outcome. More specifically, the output of u_i becomes '1' if $KNOWN_i > 0$ or '–1' if $KNOWN_i < 0$. When the output of a node becomes known, its value is propagated to the next node levels and may lead to calculation of the outputs of other unknown nodes (forward chaining).

During inference, for each node u_i, its *confidence* $Conf(u_i)$ is calculated, thus providing an estimation of how much close is variable u_i to become True or False. $Conf(u_i)$ is used to compare unknown output nodes, when insufficient conclusions have been reached. In those cases, the inference process uses the confidence measure in order to focus on unknown variables considered most important to drawing further conclusions. $Conf(u_i)$ is computed as follows:

- For a node with known output,
$$Conf(u_i) = u_i.$$
- For an input node with unknown output,
$$Conf(u_i) = 0$$
- For the remaining nodes with unknown output,
$$Conf(u_i) = \frac{\sum_k w_{i,k} Conf(u_k)}{\sum_{k:u_k\ unknown} |w_{i,k}|}$$

where u_k are the unknown inputs of node u_i.

Prior to inference, the user may supply initial values for some input nodes that are propagated to the consequent level nodes. If insufficient conclusions are drawn, the output node u_i with maximum $|Conf(u_i)|$ is selected. The inference process subsequently focuses on the input node u_k of u_i having the maximum absolute weight to u_i, since this is the node with the maximum contribution to the output of u_i. If u_k is an input node, the user is asked to give its value. If u_k is not an input node, the inference process selects it and recursively focuses on its mostly contributing input node (backward chaining). This process carries on until sufficient conclusions have been drawn.

MACIE offers two types of explanations: one justifying how conclusions were drawn and one explaining why the user is queried to supply the values of input nodes. The 'how' explanations are in the form of symbolic rules having in their conditions and conclusions the variables corresponding to the network nodes. However, they lack naturalness, since they include concepts corresponding to the meaningless random cells inserted into the network, due to inseparability. The 'why' explanations provide a trace of the network nodes causing an input value to be asked from the user during the inference process.

4.2.2 Recency Inference Engine (RIE)

MACIE's inference engine performs well in cases where the neural network constituting the knowledge base has a small number of outputs, each depending on a large portion of the inputs. This was the type of the knowledge base for the medical domain, which MACIE was applied to. However, MACIE's performance is reduced, when it is used in sparse connectionist knowledge bases having a large number of outputs, each depending on a small portion of the inputs. This was the motivation for the development of an improved inference engine, called the Recency Inference Engine (RIE) (Ghalwash 1998). As mentioned, MACIE uses the confidence measure as a criterion for choosing the unevaluated output, which inference will focus on. RIE instead examines not only output but intermediate nodes as well. More specifically, it examines the "recently triggered" nodes, that is, the nodes affected from the last input value, given by the user, whose known inputs do not suffice for the evaluation of their output. From those nodes, the one whose output is closer to be activated is selected. Selection is based on a measure, called the *convergence ratio*, which defines the likelihood of an unevaluated node to be activated. The convergence ratio is calculated separately for each unevaluated node u_i according to the following formula:

$$c(u_i) = \frac{KNOWN_i}{MAX\_UNKNOWN_i}$$

where $KNOWN_i$ and $UNKNOWN_i$ are calculated according to equations (1) and (2) above. When $|c(u_i)| > 1$, the currently known inputs of node u_i suffice for the computation of its output. More specifically, the output of u_i is '1', if $c(u_i) > 1$, and '–1', if $c(u_i) < 1$. The inference engine focuses on the node having the maximum convergence ratio among all the recently triggered nodes.

The inference process works as follows. Given that conclusions have not been reached, the recently triggered nodes are examined and the node u_i with the maximum convergence ratio is selected. The inference process then examines the input nodes of u_i and focuses on node u_k, the one with the maximum absolute weight. Node u_k is considered as having the maximum influence on the computation of u_i's output. This last step is executed recursively until u_k is an input node and then the user supplies its value. When the user supplies an input value, the convergence ratios of all nodes are recomputed and possible node activations are propagated to the next level of nodes.

Ghalwash also presents an explanation mechanism of 'how' type (similar to Gallant's), justifying conclusions via symbolic explanation rules.

Experiments involving two domains, one using the medical knowledge base used by Gallant and the other using a sparse knowledge base, demonstrated the superiority of the RIE (Ghalwash 1988). More specifically, RIE requires fewer inputs to be supplied by the user in order to draw the same conclusions as MACIE. This was due to the convergence ratio criterion that enabled the inference process to focus on the nodes mostly relevant to the computation of the outputs.

4.2.3 EXPSYS

Sima presented a connectionist expert system shell called EXPSYS (Sima and Cervenka 2000). EXPSYS is an improvement of MACIE, since it provides interactive inference engine and explanation mechanism for multi-layer neural networks trained with back-propagation and using a differentiable activation function (Sima 1995). To handle partial input information, the concept of interval state is introduced for the network neurons and back-propagation is generalized for neural networks with such neurons. The introduction of the interval states, though, degrades the comprehensibility of the network compared to Gallant's approach and, furthermore, makes the inference process more complicated. The states of a neuron are within the interval [-1, 1]. A crisp value is represented by one-point intervals; whereas unknown values are encoded with complete intervals [-1, 1].

The inference process provides the user with partial conclusions and confidences when some input values have been supplied. Confidences and outputs have to be recomputed when a new input value is presented.

'How' type explanations are provided showing the percentage influence of the inputs to the drawn conclusion. These specific explanations are used during inference in order to ask the user to provide values for the unknown inputs having the greatest influence on the outputs.

4.3 Neurules: Giving pre-eminence to symbolic framework

Neurules are a type of hybrid rules integrating symbolic rules with neurocomputing, giving pre-eminence to the symbolic component (Hatzilygeroudis and Prentzas 2000, 2001a). Neurocomputing is used within the symbolic framework to improve the performance of symbolic rules. In contrast to the other hybrid approaches described in the previous sections, 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 also produced.

The form of a neurule is depicted in Figure 7a. 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. 7b). 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), 0 (false), 0.5 (unknown)]. This gives the opportunity to easily distinguish between the falsity and the absence of a condition in contrast to symbolic rules. Also, contributes to naturalness, since any false condition does not

contribute at all in drawing the conclusion. The *output D* represents the *conclusion* (decision) of the rule. The output can take one of two values ('-1', '1') representing failure and success of the rule respectively. The significance factor of a condition represents the significance (weight) of the condition in drawing the conclusion. Table 1 presents an example neurule, from a medical diagnosis domain.

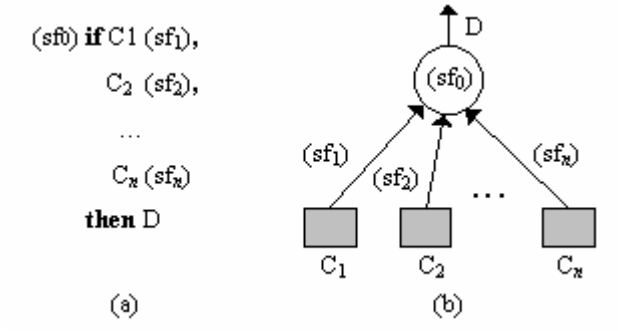


Figure 7. (a) Form of a neurule (b) a neurule as an adaline unit

Table 1. An example neurule

(-4.2) **if** pain is continuous (3.0), patient-class isnot man36-55 (2.8), fever is medium (2.7), fever is high (2.7) **then** disease-type is inflammation

Neurules can be constructed either from symbolic rules (Hatzilygeroudis and Prentzas 2000), thus exploiting existing symbolic rule bases, or empirical data (i.e., training examples) (Hatzilygeroudis and Prentzas 2001a). Each adaline unit is individually trained via the Least Mean Square (LMS) algorithm. In case of inseparability of training patterns, special techniques are used. In that case, more than one neurule having the same conclusion are produced. Actually, each neurule is a merger of more than one (propositional type) symbolic rule.

In general, the output of a neurule is computed according to the standard way used in a single neuron (see e.g. Gallant 1993). However, it is possible to deduce the output of a neurule without knowing the values of all of its conditions. To achieve this, we use a similar approach to that in (Ghalwash, 1988). We define for each neurule the *known sum* and the *remaining sum* as follows:

$$kn-sum = sf_0 + \sum_{cond_i \in E} sf_i C_i$$

$$rem-sum = \sum_{cond_i \in U} |sf_i|$$

where E is the set of evaluated conditions, U the set of unevaluated conditions and C_i is the value of condition $cond_i$. Hence, 'known-sum' is the weighted sum of the values of the already known (i.e. evaluated) conditions (inputs) of the corresponding neurule and 'rem-sum' represents the largest possible weighted sum of the remaining (i.e. unevaluated) conditions of the neurule. If $|kn\text{-}sum| > rem\text{-}sum$ for a certain neurule, then evaluation of its conditions can stop, because its output can be deduced regardless of the values of the unevaluated conditions. In this case, its output is guaranteed to be '-1' if $kn\text{-}sum < 0$ whereas it is '1', if $kn\text{-}sum > 0$. So, we define the *firing potential (fp)* of a neurule as follows:

$$fp = \frac{|kn-sum|}{rem-sum}$$

The firing potential of a neurule is an estimate of its intention that its output will become '±1'. Whenever $fp > 1$, the values of the evaluated conditions can determine the value of its output, regardless of the values of the unevaluated conditions. The rule then evaluates to '1' (true), if $kn\text{-}sum > 0$ or to '-1' (false), if $kn\text{-}sum < 0$.

There are two alternative neurule-based inference processes (Hatzilygeroudis and Prentzas 2002). The one gives pre-eminence to neurocomputing and is called *connectionism-oriented inference process*, whereas the other to symbolic reasoning and is called *symbolism-oriented inference process*. In the symbolism-oriented process, a classical rule-based backward chaining process is employed, where evaluation of a rule is based on the above neurocomputing measure. In the connectionism-oriented process, the choice of the next rule to be considered is based on the neurocomputing measure, thus the process jumps from rule to rule, but the rest is symbolic.

Experiments have shown that the connectionism-oriented process does better than those of MACIE and the Recency Inference Engine (Hatzilygeroudis and Prentzas 2001b and 2001c). On the other hand, a bit surprisingly, experiments have shown that the symbolism-oriented process does better than the connectionism-oriented one (Hatzilygeroudis and Prentzas 2002).

Neurules are also associated with an explanation mechanism, capable of providing 'how'and 'why-not' types of explanations in the form of if-then rules as well as 'why' type of explanations. Experiments have shown that neurules explanation mechanism produces more natural explanations with less rules (Hatzilygeroudis and Prentzas 2001b and 2001c).

Apart from the above, neurules support incremental development of neurule-bases, because they retain the naturalness and modularity of symbolic rules. One can easily add new neurules or remove old neurules from a neurule base without making any other changes to it, given that they do not affect existing knowledge, because neurules are functionally independent units. This is difficult for the other hybrid approaches. Also, neurule bases can be efficiently updated, i.e. without thorough reconstruction of them. As mentioned, neurules can be produced either from existing symbolic rules or from empirical data. The symbolic rules or the empirical data are called the 'source knowledge' of the corresponding neurule base. There have been methods for efficient update of a neurule base given changes to its source knowledge, in either case (Prentzas and Hatzilygeroudis 2002, Prentzas et al. 2002). Knowledge base update is a very difficult problem for the other approaches.

Due to the aforementioned features of neurules, a neurule-based system can compete more effectively with rule-based expert systems than the other hybrid approaches as far as friendliness of the interaction with the user is concerned. Not only interactive inference and natural explanation is provided, but also the naturalness and modularity of the knowledge base is retained since pre-eminence is given to the symbolic framework. On the contrary, the other hybrid approaches follow a different approach by enriching a neural network with an interactive inference engine and an explanation mechanism retaining to a great degree the deficiencies of a neural knowledge base regarding comprehensibility and modularity.

5. Conclusions

In this paper, we focus on approaches integrating symbolic rules and neural networks, which offer a unified neuro-symbolic knowledge representation and reasoning scheme possessing the basic functionalities of an expert system (i.e. separation of knowledge from its use, interactive inference and provision of explanations). Such functional features are deemed important as they improve the user-friendliness of the hybrid system during its interaction with the user.

First, we point out that the above category of integrations cannot be fully accommodated by existing categorization schemes. Therefore, we introduce a new category, namely 'representational integrations', in the existing schemes. Furthermore, we distinguish between two sub-categories of such approaches: those giving pre-eminence to connectionism and those giving pre-eminence to the symbolic framework.

A prominent disadvantage of the systems belonging to the first sub-category involves the deficiencies of the knowledge base (to a lesser or greater dergee) as far as naturalness and modularity are concerned. As explained, this drawback is not a characteristic of neurules, a representative of the second sub-category. By giving pre-eminence to the symbolic framework, neurules retain the naturalness and modularity of symbolic rules to a large degree. On the other hand, neurules are proved to be even more efficient than the other approaches. Additionally, incremental development and update capabilities of a neurule base are retained.

We've noticed that this category of integrations, especially the second sub-category, has been overlooked. However, we believe that it constitutes a good direction for further research.

Acknowledgements

This work was supported by the Research Committee of the University of Patras, Greece, Program "Karatheodoris", project No 2788.

References

Ajjanagadde, V. and Shastri, C. (1995), "Reasoning with Rules and Variables in Neural Networks", in Goonatilake, S. and Khebbal, S. (eds.) *Hybrid Architectures for Intelligent Systems*, John Wiley and Sons, pp.209-220.

Andrews, R., Diederich, J. and Tickle, A. (1995), "A Survey and Critique of Techniques for Extracting Rules From Trained Artificial Neural Networks", Knowledge-Based Systems, vol. 8(6), pp. 373-389.

Bookman, L. and Sun, R. (eds.) (1993), Special Issue on Integrating Neural and Symbolic Processes, Connection Science, vol. 5(3-4).

Buchanan, B. G. and Shortliffe, E. H. (1984), Rule-Based Expert Systems, Eddison-Wesley, Reading, MA.

Cloete, I. and Zurada, J. M. (eds.) (2000), Knowledge-Based Neurocomputing, The MIT Press, Cambridge.

Dolan, C. P. and Smolensky, P. (1989), "Tensor Product Production System: a Modular Architecture and Representation", Connection Science, vol. 1, pp. 53-68.

Fu, L. M. (1993), "Knowledge-based connectionism for revising domain theories", IEEE Transactions on Systems, Man and Cybernetics, vol. 23(1), pp. 173-182.

Fu, L. M. (ed.) (1994), Proceedings of the International Symposium on Integrating Knowledge and Neural Heuristics, Pensacola, Florida.

Gallant, S. I. (1988), "Connectionist Expert Systems", Communications of the ACM, vol. 31(2), pp. 152-169.

Gallant, S. I. (1993), Neural Network Learning and Expert Systems, MIT Press.

Garcez, A. S. d'Avila, Broda, K. and Gabbay, D. M. (2002), Neural-Symbolic Learning Systems: Foundations and Applications, Perspectives in Neural Computing, Springer-Verlag.

Ghalwash, A. Z. (1998), "A Recency Inference Engine for Connectionist Knowledge Bases", Applied Intelligence, vol. 9(3), pp. 201-215.

Giles, C.L. and Omlin, C.W. (1993), "Extraction, Insertion and Refinement of Symbolic Rules in Dynamically Driven Recurrent Neural Networks", Connection Science, vol. 5(3-4), pp. 307-337.

Gonzalez, A. and Dankel, D. (1993), The Engineering of Knowledge-Based Systems: Theory and Practice, Prentice-Hall, Inc.

Hatzilygeroudis I. (1996), "SILO: Integrating Logic in Objects for Knowledge Representation and Reasoning", International Journal on Artificial Intelligence Tools, 5(4), pp. 403-446.

Hatzilygeroudis, I. and Prentzas, J. (2000), "Neurules: Improving the Performance of Symbolic Rules", International Journal on Artificial Intelligence Tools, 9(1), pp. 113-130.

Hatzilygeroudis, I. and Prentzas, J. (2001a), "Constructing Modular Hybrid Rule Bases for Expert Systems", International Journal on Artificial Intelligence Tools, 10(1-2), pp. 87-105.

Hatzilygeroudis, I. and Prentzas, J. (2001b), "An Efficient Hybrid Rule Based Inference Engine with Explanation Capability", Proceedings of the 14th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2001), Key West, Florida, USA, May 21-23, pp. 227-231.

Hatzilygeroudis, I. and Prentzas, J. (2001c), "HYMES: A Hybrid Modular Expert System with Efficient Inference and Explanation", Proceedings of the 8th Panhellenic Conference on Informatics, Nicosia, Cyprus, Vol. 1, 422-431.

Hatzilygeroudis, I. and Prentzas, J. (2002), "Multi-inference with Multi-neurules", Proceedings of the 2nd Hellenic Conference on AI (SETN 2002), Thessaloniki, Greece, as I. Vlahavas and C. Spyropoulos (eds.), Methods and Applications of AI, LNAI vol. 2308, Springer-Verlag, 30-41.

Hilario, M. (1997), "An Overview of Strategies for Neurosymbolic Integration", in R. Sun and E. Alexandre (eds.), Connectionist-Symbolic Integration: From Unified to Hybrid Approaches, Lawrence Erlbaum, ch.2.

Horrocks, I., Sattler, U. and Tobies, S. (1999), "Practical reasoning for expressive description logics", in H. Ganzinger, D. McAllester and A. Voronkov (eds.), Proceedings of LPAR'99, LNAI vol. 1705, Springer-Verlag, 161-180.

Jackson, P. (1999), Introduction to Expert Systems, 3rd Edition, Addison Wesley.

Kasabov, N. and Shishkov, S.I. (1993), "A Connectionist Production System with Partial Match and its Use for Approximate Reasoning", Connection Science, vol. 5(3-4), pp. 275-305.

Mahoney, J.F. and Mooney, R.J. (1993), "Combining Connectionist and Symbolic Learning to Refine Certainty Factor Rule Bases", Connection Science, vol. 5(3-4), pp. 339-364.

Mani, D.R. and Shastri, L. (1993), "Reflexive Reasoning with Multiple Instantiation in a Connectionist Reasoning System with a Type Hierarchy", Connection Science, vol. 5(3-4), pp. 205-242.

McGarry, K., Wermter, S. and MacIntyre, J. (1999), "Hybrid Neural Systems: From Simple Coupling to Fully Integrated Neural Networks", Neural Computing Surveys, vol. 2, pp. 62-93.

Medsker, L. R. (1994), Hybrid Neural Network and Expert Systems, Kluwer Academic Publishers.

Medsker, L. R. (1995), Hybrid Intelligent Systems, Kluwer Academic Publishers, Second Printing, 1998.

Omlin, C. W. and Giles, C. L. (1996), "Rule Revision with Recurrent Neural Networks", IEEE Transactions on Knowledge and Data Engineering, vol. 8(1), pp. 183-188.

Palade, V., Neagu, D. and Patton, R. J. (2001), "Interpretation of trained neural networks by rule extraction", in Reusch, B. (ed.), *Computational Intelligence: Theory and Applications*, LNCS 2206, Springer-Verlag, pp. 152-161.

Prentzas, J. and Hatzilygeroudis, I. (2002), "Updating a Hybrid Rule Base with Changes to its Symbolic Source Knowledge", Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002), Lyon, France, July 21-26, pp. 250-254.

Prentzas, J., Hatzilygeroudis, I. and Tsakalidis, A. (2002), "Updating a Hybrid Rule Base with New Empirical Source Knowledge", Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence(ICTAI-2002), Washington, D.C., USA, November 4-6, pp. 9-15.

Reichgelt, H. (1991), Knowledge Representation, An AI perspective, Ablex.

Samad, T. (1992), "Hybrid Distributed/Local Connectionist Architectures", in A. Kandel (ed.), Hybrid Architectures for Intelligent Systems, pp. 200-218.

Sima, J. (1995), "Neural Expert Systems", Neural Networks, vol. 8(2), pp. 261-271.

Sima, J. and Cervenka, J. (2000). "Neural Knowledge Processing in Expert Systems", in

Cloete, I. and Zurada, J. M. (eds.), Knowledge-Based Neurocomputing, The MIT Press, Cambridge, pp. 419-466.

Sun, R. (1994), Integrating Rules and Connectionism for Robust Commonsense Reasoning, John Wiley and Sons.

Sun, R. and Alexandre, E. (eds.) (1997), Connectionist-Symbolic Integration: From Unified to Hybrid Approaches, Lawrence Erlbaum.

Touretzky, D. S. and Hinton, G. E. (1988), "A Distributed Connectionist Production System", Cognitive Science, vol. 12, pp. 423-466.

Towell, G. and Shavlik, J. (1993), "Extracting refined rules from knowledge-based neural networks", Machine Learning, vol. 13(1), pp. 71-101.

Towell, G. and Shavlik, J. (1994), "Knowledge-based artificial neural networks", Artificial Intelligence, vol. 70(1-2), pp. 119-165.

Wermter, S. and Sun, R. (eds) (2000), Hybrid Neural Systems, Springer-Verlag.