INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center St. • Suite 600 • Berkeley, California 94704-1198 • (510) 643-9153 • FAX (510) 643-7684

Hybrid Approaches to Neural Network-based Language Processing

Stefan Wermter

TR-97-030

1997

Abstract

In this paper we outline hybrid approaches to artificial neural network-based natural language processing. We start by motivating hybrid symbolic/connectionist processing. Then we suggest various types of symbolic/connectionist integration for language processing: connectionist structure architectures, hybrid transfer architectures, hybrid processing architectures. Furthermore, we focus particularly on loosely coupled, tightly coupled, and fully integrated hybrid processing architectures. We give particular examples of these hybrid processing architectures and argue that the hybrid approach to artificial neural network-based language processing has a lot of potential to overcome the gap between a neural level and a symbolic conceptual level.

1 Motivation for hybrid symbolic/connectionist processing

In recent years, the field of hybrid symbolic/connectionist processing has seen a remarkable development [48, 38, 18, 2, 36, 59, 55, 20, 37, 61, 9]. Currently it is still an open issue whether connectionist or symbolic approaches alone will be sufficient to provide a general framework for processing natural language [51, 11, 27, 56]. However, since human language capabilities are based on real neural networks in the brain, artificial neural networks (also called connectionist networks) provide one essential starting point for modeling language processing. On the other hand, human language capabilities include rule-based reasoning which is supported well by symbolic processing. Given this general situation, the motivation for examining hybrid connectionist models of natural language processing comes from different directions.

From the perspective of cognitive neuroscience, a symbolic interpretation of a connectionist architecture is desirable, since the brain has a neuronal structure and has the capability to perform symbolic reasoning. This leads to the question how different processing mechanisms can bridge the large gap between, for instance, acoustic or visual input signals and symbolic reasoning for language processing. The brain uses a complex specialization of different structures. Although a lot of the functionality of the brain is not yet known in detail, its architecture is highly specialized and organized at various levels of neurons, networks, nodes, cortex areas and their respective connections. Furthermore, different cognitive processes are not homogeneous and it is to be expected that they are based on different representations [54]. Therefore, there is evidence from the brain that multiple architectural representations may also be involved in language processing.

From the perspective of knowledge-based natural language processing, hybrid symbolic/connectionist representations are advantageous, since different mutually complementary properties can be combined. Symbolic representations have advantages with respect to easy interpretation, explicit control, fast initial coding, dynamic variable binding and knowledge abstraction. On the other hand, connectionist representations show advantages for gradual analog plausibility, learning, robust fault-tolerant processing, and generalization to similar input. Since these advantages are mutually complementary, a hybrid symbolic connectionist architecture can be useful if different processing strategies have to be supported.

The development of hybrid symbolic/connectionist architectures is still a new research area and there is no general theory of hybrid architectures [6]:"The development of additional constraints for defining appropriate mappings between hybrid models and the resultant types of hybrid models remain areas for future research." However, the question for an appropriate architecture for a given task is very important [54]: "We need to consider seriously ways of structuring these different components; in other words, we need to consider *architectures*, which thus occupy a clearly more prominent place in this area of research compared with other areas in AI."

In general, there are different global possibilities for a symbolic/connectionist integration. Based on the argument from cognitive neuroscience, symbolic connectionist integration relies on a symbolic interpretation of a connectionist architecture and connectionist processing. There are just connectionist representations for different tasks which can be interpreted symbolically. On the other hand, based on the argument from knowledge-based natural language processing, symbolic connectionist integration relies on a combination of different symbolic and connectionist representations. However, these interpretation approaches and representation approaches are closely related and in general there is a continuum of possible connectionist/symbolic architectures.

2 Types of symbolic/connectionist integration for language processing

Although there has been quite a lot of work in the field of hybrid and connectionist natural language processing, most work has concentrated on a single individual system rather than on types of hybrid systems, their interpretation, and communication principles within various architectures. Previous characterizations of architectures have covered certain specific connectionist architectures, for instance recurrent networks [31, 39], or they have covered expert systems/knowledge-based systems [37, 24, 55]. In contrast, here we will concentrate on various types of hybrid connectionist natural language processing.

In figure 1 there is an overview of different possibilities for integration in natural language processing. Continuous connectionist representations are represented by a circle, discrete symbolic representations by a square. Symbolic interpretations of connectionist representations are shown as squares with dotted lines.

Connectionist structure architectures are the first type of symbolic/connectionist architectures. They can model higher cognitive functions and rely solely on connectionist representations. Symbolic knowledge arises by an interpretation process of the connectionist representations. Often specific knowledge of the task is built into the connectionist structure architecture.

Hybrid transfer architectures transfer symbolic representations into connectionist representations or vice versa. Using a transfer architecture it is possible to insert or extract symbolic knowledge into or from a connectionist architecture. The main processing is performed by connectionist representations but there are automatic procedures for transferring from connectionist representations to symbolic representations or vice versa. Hybrid transfer architectures differ from connectionist structure architectures by the automatic transfer into and from symbolic representations. While certain units in connectionist structure architectures may be interpreted symbolically by an observer, only hybrid transfer architectures allow the knowledge transfer into rules, automata, grammars, etc.

Hybrid transfer architectures transfer symbolic and connectionist representations, but the symbolic and connectionist knowledge is not yet applied at the same time to combine the complementary advantages of each representation for a given task. Such a combination

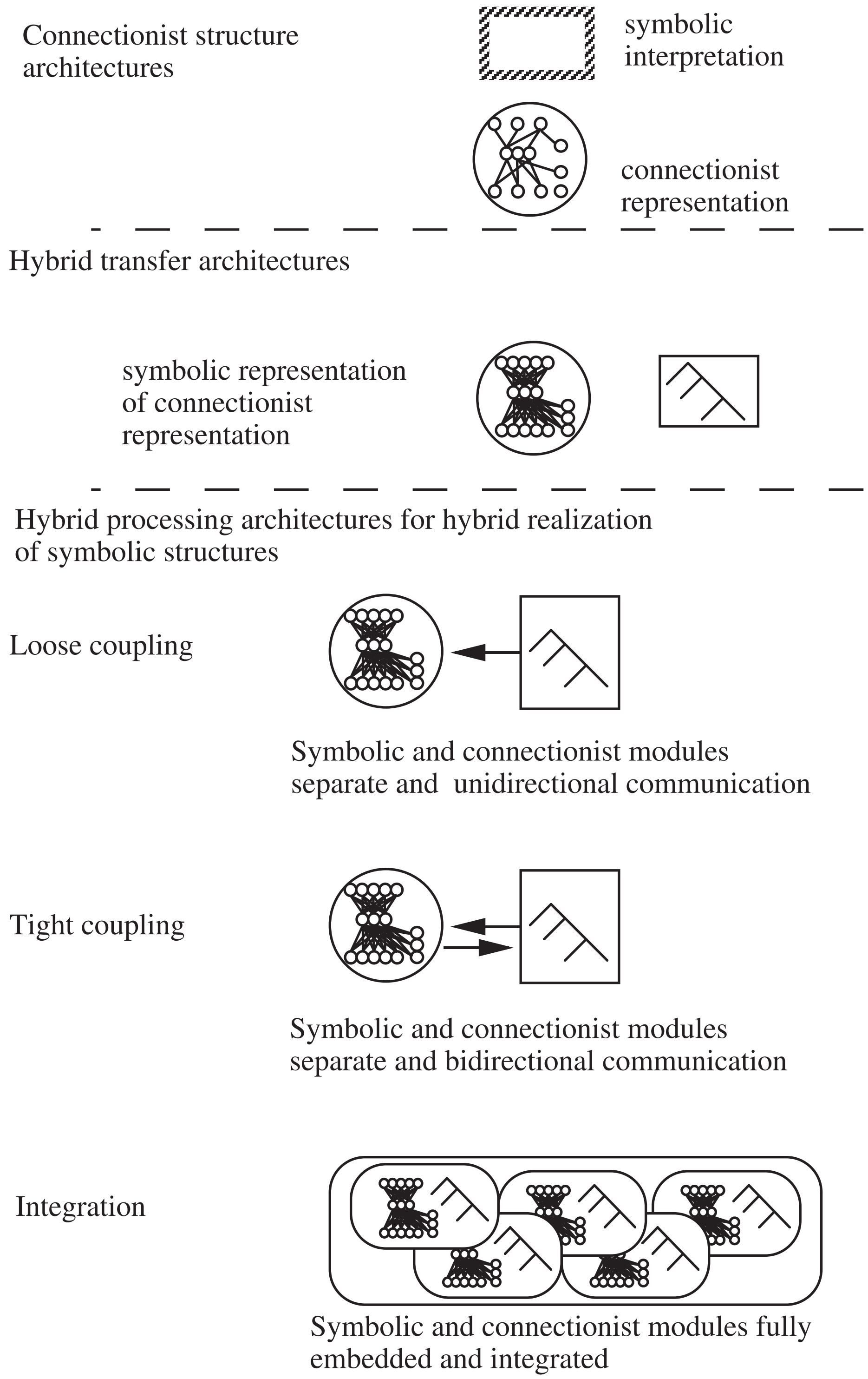

Figure 1: Overview about various types of symbolic/connectionist integration

of symbolic and connectionist representations is possible in *hybrid processing architectures*, which contain both symbolic and connectionist modules appropriate to the task. Here, symbolic representations are not just initial or final representations. Rather, they are combined and integrated with connectionist representations in many different ways.

Connectionist and symbolic modules in hybrid processing architectures can be loosely coupled, tightly coupled or completely integrated (see figure 1). A *loosely coupled hybrid architecture* has separate symbolic and connectionist modules. The control flow is sequential in the sense that processing has to be finished in one module before the next module can begin. Only one module is active at any time, and the communication between modules is unidirectional.

A *tightly coupled hybrid architecture* contains separate symbolic and connectionist modules, and control and communication are via common internal data structures in each module. The main difference between loosely and tightly coupled hybrid architectures is common data structures which allow bidirectional exchanges of knowledge between two or more modules. Processing is still in a single module at any given time, but the output of a connectionist module can have direct influence on a symbolic module (or vice versa) before it finishes its global processing. In this way feedback between two modules is possible and the communication can be bidirectional.

In a *fully integrated hybrid architecture* there is no discernible external difference between symbolic and connectionist modules, since the modules have the same interface and they are embedded in the same architecture. The control flow may be parallel and the communication between symbolic and connectionist modules is via messages. Communication may be bidirectional between many modules, although not all possible communication channels have to be used. This is the most advanced of the hybrid processing architectures. In the remainder of this article we will show detailed examples of each of these types of symbolic/connectionist architectures.

3 Connectionist structure architectures

In this section we describe principles of connectionist structure architectures. Knowledge of some task is built into the connectionist structure architecture, and a symbolic interpretation is assigned by an observer. Much early work on structured connectionism can be traced back to work by Feldman and Ballard, who provided a general framework of structured connectionism [15]. This framework was extended for natural language processing in many different directions, for instance for parsing [12, 13, 46], language acquisition [14, 25], explanation [8], and semantic processing [26, 47].

More recent work along these lines focuses on the so-called *NTL, neural theory of language* which attempts to bridge the large gap between neurons and cognitive behavior [16, 49]. The NTL framework is challenging since it tries to study neural processing mechanisms for high conceptual cognitive processing like embodied semantics, reasoning, metaphor interpretation, etc.

In order to attack such a challenging task it is necessary to focus on different processing mechanisms and certain representative problems. One such problem is verbal aspect which describes the temporal character of events (like past tense or progressive form). Narayanan has developed and implemented a computational model of verbal aspect and he argues that the semantics of aspect is grounded in sensory-motor primitives [41]. This model was developed around processing mechanisms like schemas and petri-networks which could also be transferred into structured connectionist networks. In related work Bailey developed a computational motor-control model of children's acquisition of verb semantics for words like push, pull [1]. This work supports the claim that the semantics of verb actions is grounded in sensory-motor primitives. Statistic Bayesian model merging is used for learning the task of verb semantics. Both models focus on the computational level but they have close relationships to neural plausible models or recruitment of neuron-like elements.

While these two related models for embodied event modeling can be viewed at a computational level, there is more neurally inspired work for event modeling and temporal processing at a connectionist level as well (e.g. [49, 21]. For instance, schemas from a computational level can be modeled using structured connectionist terms from the SHRUTI system which was previously developed by Shastri and Ajjanagadde [21]. A particularly interesting recent work is a model of rapid memory formation in the Hippocampal System [49]. Focusing on events like "John gave Mary a book on Tuesday", it is studied how a neural code can be transfered rapidly into a structural code so that it can be retrieved later. This SMRITI system is one of few connectionist systems which take temporal neural processing seriously at a neurobiological level. It relates the function of the system to the human Hippocampal System which is taken to be central for memory formation and event encoding.

In general, within the NTL framework, structured connectionist networks are seen as central for modeling systems of natural language processing. However, above this level of structured connectionist networks there is a computational level which provides interpretations at a symbolic level and which provides links to a conceptual linguistic level. At the computational level well-known techniques like Petri Nets, statistical model merging, feature structures are used for symbolic interpretation. On the other hand, below the level of structured connectionist networks there is a computational neurobiology level which links connectionist networks with the biological level. Because of this large challenging task of linking biological, neurobiological, connectionist, computational and cognitive level, it will be interesting to see what a complete detailed NTL architecture containing structured connectionist networks will look like in the future.

We will now focus on a few examples of different structured connectionist architectures which have been developed in recent years. One example of a connectionist structure architecture is provided as part of the SCAN project[59]. One task within this framework is structural disambiguation of a given sentence or phrase. In general many different structural interpretations of phrases are possible and only the integration of different constraints allows the system to make a correct structural disambiguation decision. For instance, there are syntactic locality constraints which realize a preference for a local attachment of constituents. On the other hand, there are also semantic plausibility constraints which have to be considered for the structural disambiguation. Different constraints may differ in strength and have to be integrated.

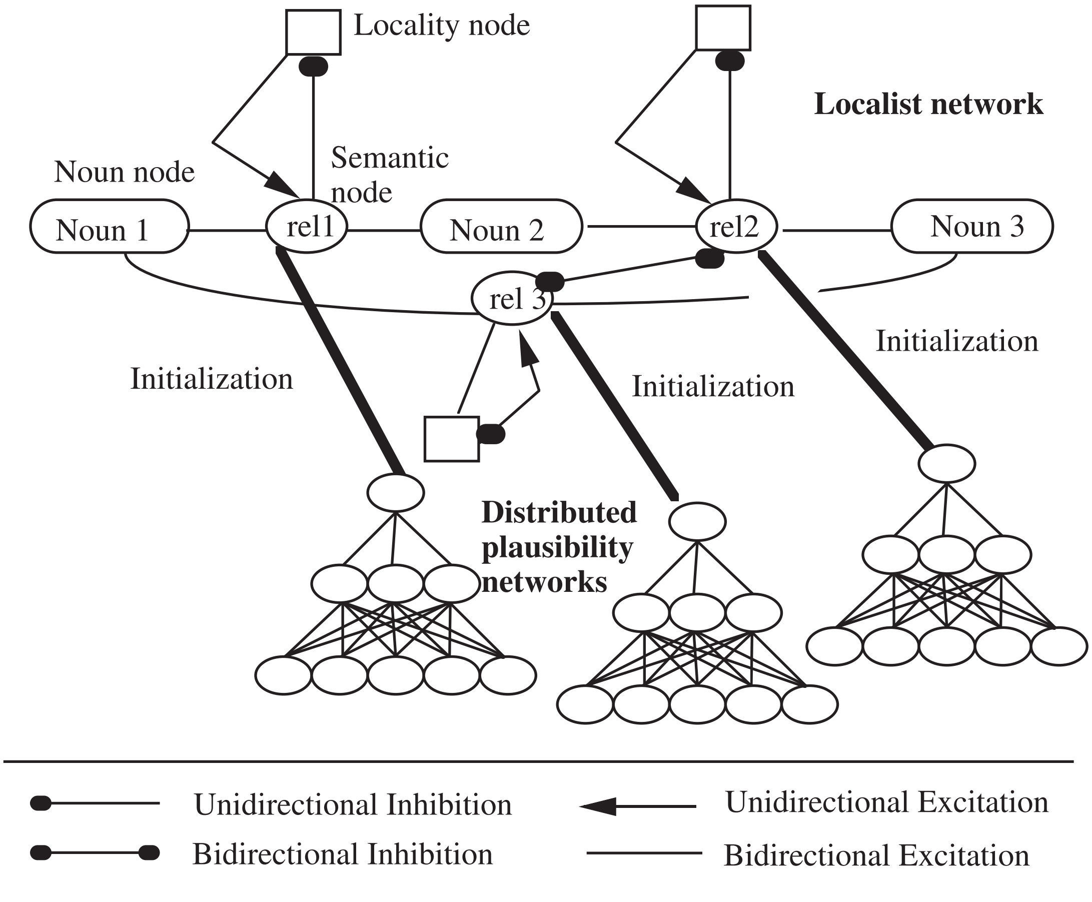

Figure 2: Connectionist structure architecture for disambiguation

Such an integration of competing constraints of varying strengths can be represented well in a localist connectionist network, since the explicit nodes and connections in the network can be used to model these competing constraints (see figure 2). There is one noun node for each head noun of a phrase group, one semantic node for the semantic plausibility of a semantic relation between two head nouns, and one locality node for the syntactic locality of a head noun. Excitatory connections link the semantic nodes and the noun nodes; competing semantic nodes are connected via inhibitory connections. Furthermore, locality nodes provide activation via excitatory connections for the semantic nodes and the locality nodes are inhibited by the semantic nodes. Using a localist network, phrases like "conference on hydrodynamics in the outer atmosphere" and "conference on hydrodynamics in room

113" could be assigned different structural interpretations depending on the attachment of the last constituent.

This localist network represents the possible static connections for the structural disambiguation task. The relevant semantic knowledge can be learned in distributed connectionist plausibility networks. While the syntactic representation in the localist network can be applied for different domains, the semantic representations have to be adapted to different domains. Therefore it is important that the knowledge acquisition effort is restricted to the semantic representations; this is one motivation for learning semantic relationships in plausibility networks. There is one plausibility network for each semantic node. Each plausibility network has a feedforward structure and receives the semantic representations of the two connecting nouns as input. The output of such a network consists of a plausibility value which indicates the likelihood of the semantic relationship.

In this connectionist structure architecture, then we have distributed plausibility networks which provide bottom-up activation for the plausibility of semantic relationships within a localist connectionist network. After initialization, activation spreads within the localist network. In each cycle the activation of a particular node is updated. The new activation of a node is calculated from the activation of the previous cycle plus the activation over excitatory connections minus the activation over inhibitory connections. This spreading activation process in the localist network stops after the different possibly competing constraints have been integrated. The final activation of the nodes determines the structural disambiguation of a phrase.

Another recent connectionist structure architecture is CONSYDERR [53]. The task of this system is to make inferences, as they are required for instance in natural language understanding systems. The system has an interesting architecture since it consists of a structured connectionist architecture with two different components. The localist level represents inference knowledge at a more abstract concept level, while the distributed level represents the details of the inferencing process at the feature level (see figure 3).

Inferences are represented either directly within the localist component ($A \rightarrow B$) or indirectly via flow of activation within the distributed component. Each node within the overall architecture is connected in a structured meaningful way to other nodes in the architecture. Each node has a certain activation, receives activation and sends activation in each cycle. Each cycle of communication between the nodes in the localist and distributed components is divided into three phases. First the activations of the nodes are computed in the localist inference component. Next, activation is spread top-down from the localist nodes to the nodes in the distributed component (phase 1). Then, the activations are updated in the distributed component (phase 2) and finally these activations are propagated upwards to the localist component again (phase 3).

In general, this architecture is a good representative of connectionist structure architectures. Each node has a meaning, that is, each node represents a concept or a feature, and each node is selectively connected to other nodes. Even the flow of activation is structured according to a 3-phase cycle. Furthermore, concepts are organized in the localist concept component while more detailed features for these concepts are organized in the distributed feature component. Each node in this architecture can be given a symbolic interpretation based on its current activation.

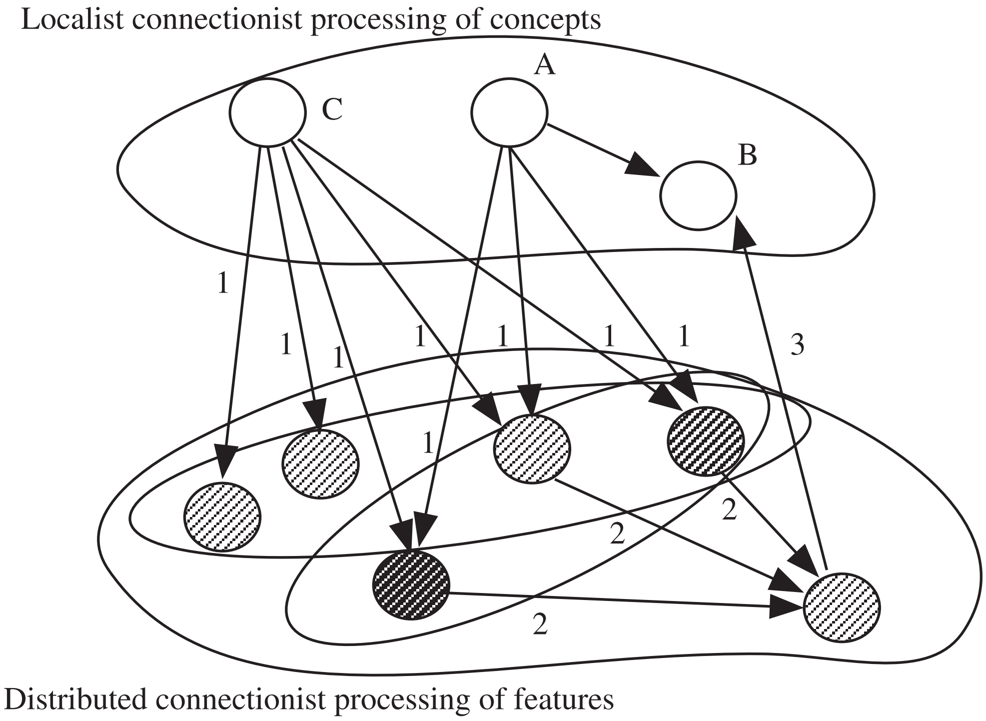

Figure 3: Connectionist structure architecture for inferencing

As we have seen, connectionist structure architectures can also contain distributed components. In such a case, there are several modular parts within the overall connectionist architecture. The individual modules provide a kind of structure within this architecture, even though some internal elements within a module may constitute distributed representations. A good example of a connectionist structure architecture of this modular distributed category was developed for automatic feature acquisition. Connectionist architectures can provide an automatic formation of distributed representations for input and output (e.g., [10, 11, 38]) which have the potential to increase robustness and facilitate learning. One way of building distributed input representations is by determining them dynamically during the learning process. A distributed connectionist representation of a symbol is held in a global lexicon and is updated automatically based on its relationships to the distributed representations of other lexical entries. This update of the representation has been called symbolic recirculation [10, 38] since the representations are constantly changed and recirculated within the architecture during learning.

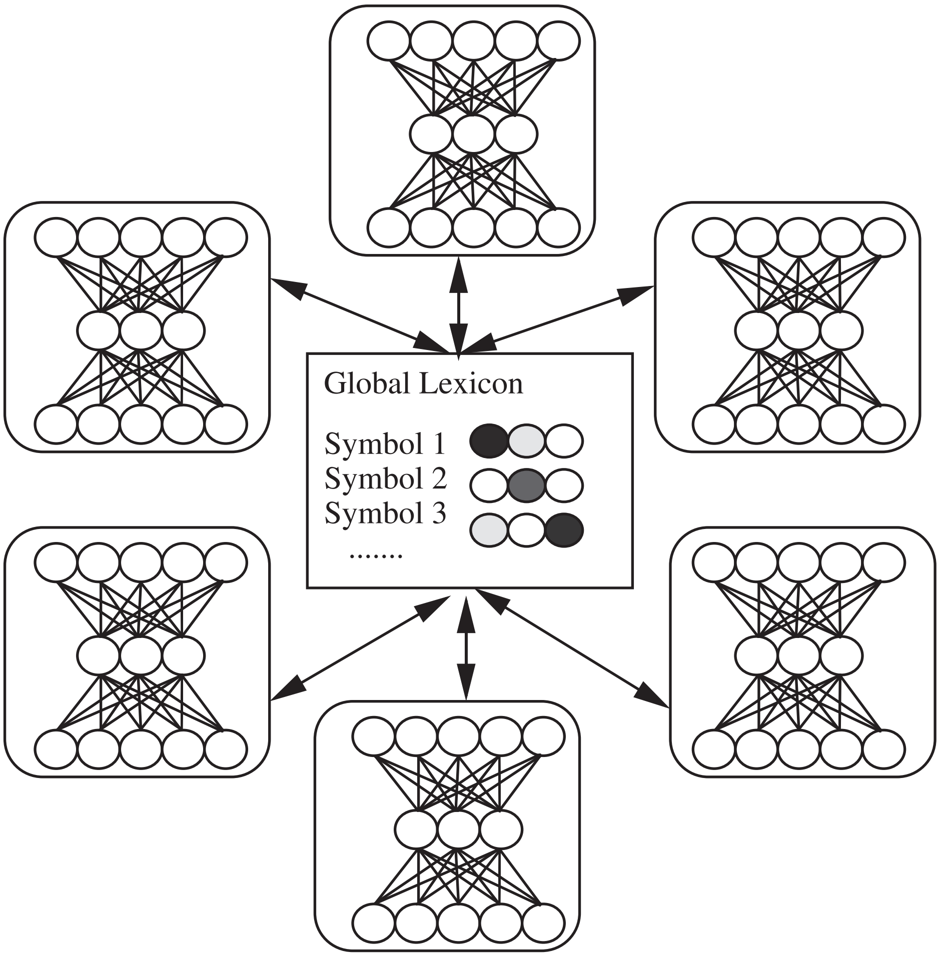

Figure 4: Connectionist structure architecture using symbolic recirculation for script understanding

Symbolic recirculation can be used within a single network, for instance a feedforward network, but symbolic recirculation can also be used within a larger architecture. We focus first on the recirculation principle within a single network, and then go on to describe a larger connectionist structure architecture with symbolic recirculation. For determining the input representation dynamically, the error δ is not only computed for the output layer and hidden layers but also for the input layer. Then the new representation values of the input units are computed based on their values in the preceding cycle plus the current change, which is based on the error within the input layer.

During this automatic feature formation process all the communication is directed by a central lexicon, which contains the learned distributed representation of each symbol. In figure 4 we show a generalization for a connectionist structure architecture with six networks. Input and output of these networks are symbols which are represented as distributed vectors. For each learning step within each network it is possible to change the input representation and therefore to change the distributed representation of the respective symbols in the global lexicon.

This connectionist structure architecture has been used for parsing simple sentences and for providing case role and script role representations. It is a connectionist structure architecture, since only connectionist networks are used for the assignment of case and script roles. Furthermore, it is possible to interpret the distributed representations symbolically.

There are many more connectionist architectures which allow a symbolic interpretation for instance at the output layers of their network. There are modular network architectures for case role assignment in Gestalt-networks [52], confluent dual RAAM-networks for translation [5], the mixture of expert networks for hierarchical control [28] and the two-tier architecture for corpus analysis [3]. All this recent work contains various forms of modularity and structuring within an overall connectionist architecture. Furthermore, symbolic interpretation of the results is possible, at least at certain nodes.

4 Hybrid transfer architectures

Hybrid transfer architectures transfer symbolic representations into connectionist representations or vice versa. Using a transfer architecture it is possible to insert or extract symbolic knowledge into or from a connectionist architecture. Hybrid transfer architectures differ from connectionist structure architectures by this automatic transfer into and from symbolic representations. While certain units in connectionist structure architectures may be interpreted symbolically only hybrid transfer architectures allow the knowledge transfer into or from rules, automata, grammars, etc.

There are many ways to insert or extract symbolic representations into or from connectionist representations. The range of possibilities depends on the form of the symbolic representations (e.g. context-free rules, automata, etc) and the connectionist representations (e.g. use of weights, activations, etc). In this section, we will show two representative approaches for extracting knowledge from connectionist networks. We have chosen these approaches since they have been widely explored during the last few years and since they are relatively straightforward. At the end of this section we will refer to additional approaches using hybrid transfer architectures.

One major problem for hybrid transfer architectures is that the architecture itself has to support the transfer. A good example is the work on activation-based automata extraction from recurrent networks [19, 42, 43]. First simple finite state automata are used for generating training examples of a given regular grammar. These training sequences are used to train a second order connectionist network as an acceptor for the regular grammar.

This network has been designed particularly to meet the constraints of symbolic automata behavior. For a given input and a given state, the network determines the next state. So this network structure directly supports the underlying symbolic transfer to and from the network by integrating architectural constraints from symbolic automata.

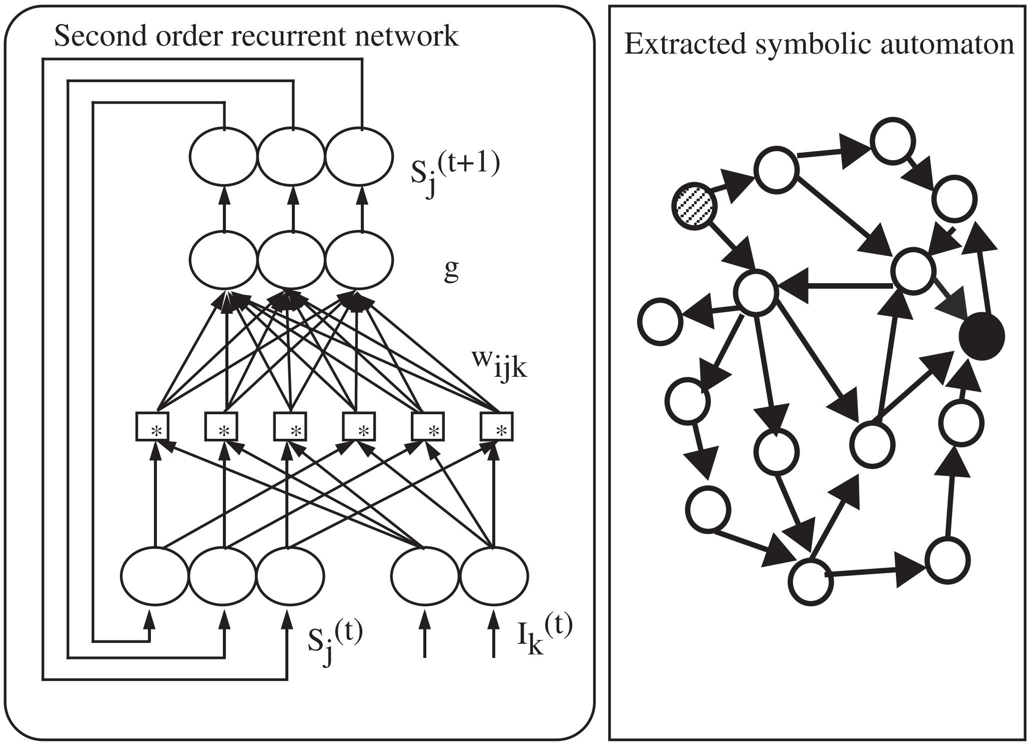

Figure 5: Hybrid transfer architecture for automata extraction from a recurrent network

Figure 5 shows the connectionist architecture. The network receives the representation of the input $I_k(t)$ and the representation of the current state $S_j(t)$ at time t. Then the output representation $S_j(t+1)$ is the state at the next time step. For the extraction of symbolic representations from a trained connectionist network, the output unit activations are partitioned into q discrete intervals. Therefore, there are q^n partitions for n units. The presentation of all input patterns generates a search tree of transitions of an automaton where one node corresponds to one partition. Then a minimization algorithm can be used, which reduces this deterministic automaton. The extracted rules have the form: If the state is x and the input is y, then go to state z. In general, this architecture is a hybrid transfer architecture where symbolic automata are extracted from connectionist networks.

The activation-based extraction of automata is just one example of a hybrid transfer architecture and there are several further possibilities. For instance, a weight-based transfer between symbolic rules and feedforward networks has been extensively examined in knowledge based artificial neural networks (KBANN) [50, 7]. This weight-based transfer uses the weights rather than the activations as the main knowledge source for induction and extraction. While an activation-based transfer is based on the activations of certain units a weight-based transfer focuses on a more detailed weight level. Therefore, the architectures are fairly simple, in this case three-layer feedforward networks.

The underlying assumption of this approach is that simple rule knowledge is often available, and that this knowledge should be used for initializing connectionist networks [50]. Then connectionist learning can be used to refine this knowledge. Ultimately the symbolic knowledge can be extracted from the connectionist network yielding an improved set of interpretation rules. Therefore, this KBANN approach is a good example for a hybrid transfer architecture.

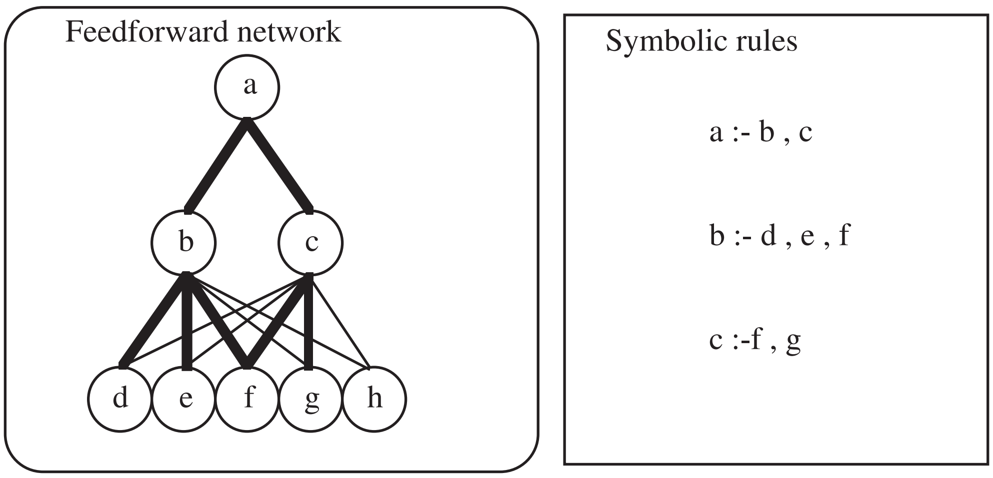

Figure 6: Hybrid transfer architecture for rule extraction and insertion from and to a feedforward network

Figure 6 shows a simple set of rules and the corresponding network architecture. Thick connections are associated with high weights, thin connections are low weights. That way initial domain knowledge can be integrated into the connectionist architecture. After a training period the weights can be transfered again to symbolic rules.

In contrast to activations, there are many more weights in a connectionist network, so that heuristics for grouping, eliminating and clustering of weights are particularly important. For extraction, the weights associated with a connectionist unit are clustered and replaced by the average of the weights of the cluster. If there are clusters which do not contribute to the functionality of a unit (since the weights are too small), these clusters are eliminated. In a final step, the weights are written as rules. Here it is assumed that units can only have two states. Typical simple rules could be "if 2 of A B G then J". This so-called N-of-M-method assumes that N of M units are equally important for the conclusion. The rules in this KBANN approach are still fairly simple, propositional, binary, acyclic, without recursion and variables. However, simple transfers of connectionist and symbolic knowledge can be performed in a general manner, as for example in simple forms of natural language processing but also for many other simple domains.

In further work, activation-based transfer between context-free rules and feedforward networks is described in a symbolic manipulator for context-free languages [40], and another form of weight-based insertion of symbolic rules has been proposed for recurrent networks [33]. Extraction of symbolic rules can also be based on multiplicative networks which control the association of conditions and actions [35, 34].

5 Hybrid processing architectures

Hybrid transfer architectures do not apply symbolic and connectionist knowledge simultaneously to combine the complementary advantages of each representation in solving a given task. Hybrid transfer architectures just transfer symbolic and connectionist representations into each other. However, a combination of mutual advantages of symbolic and connectionist representations may be advantageous for "hybrid tasks". This combination of symbolic and connectionist representations during task processing is performed in *hybrid processing architectures* which contain symbolic and connectionist modules.

5.1 Loosely coupled hybrid processing architectures

As we have outlined above, connectionist and symbolic modules in hybrid processing architectures can be loosely coupled, tightly coupled or completely integrated (see figure 1). In a *loosely coupled hybrid architecture* processing has to be completed in one module before the next module can begin. For instance, the WP model [57, 44] for phenomenologically plausible parsing used a symbolic chart parser to construct syntactic localist networks which were also connected to semantic networks. Although the network itself may be viewed as a purely connectionist architecture, the overall architecture is a hybrid processing architecture. First a symbolic parser is used for network building, and after the chart parser has finished its processing the connectionist spreading activation in localist networks takes place.

Another architecture where the division of symbolic and connectionist work is even more loosely coupled has been described in a model for structural parsing within the SCAN framework [59]. First, a chart parser is used to provide a structural tree representation for a given input, for instance for phrases or sentences (see figure 7). This symbolic structural tree representation is used to extract important head nouns and their relationships. In a second step, these triples of "noun relationship noun" are used as input for several feedforward networks which produce a plausibility measure of the relationship. Based on this connectionist output, another symbolic restructuring component changes the original tree representation if the semantic feedforward networks indicate that this is necessary.

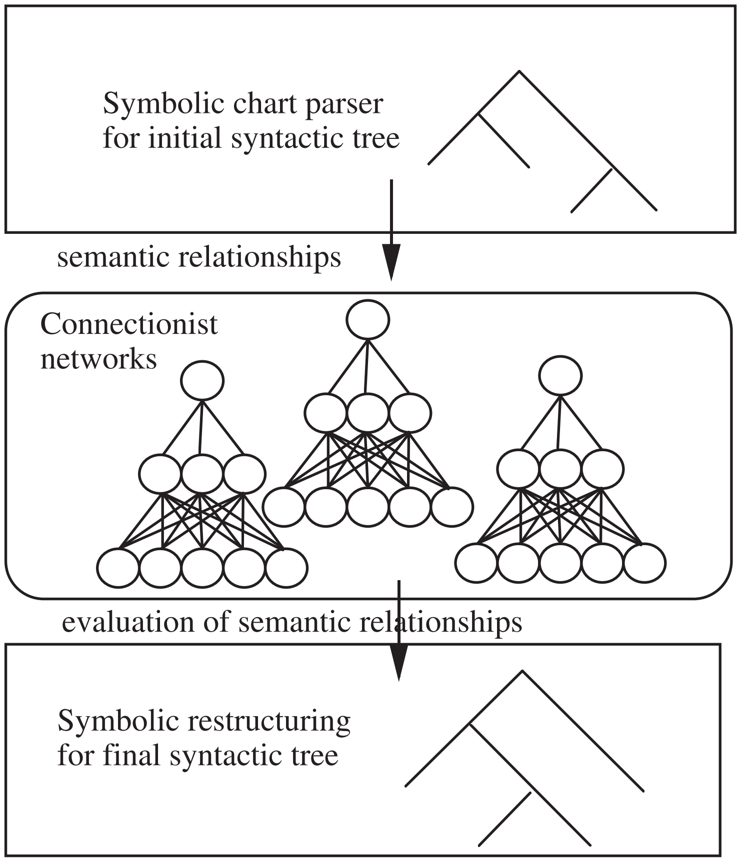

Figure 7: Loosely coupled symbolic chart parser and connectionist plausibility networks

This system has a loosely coupled hybrid processing architecture since there is a clear division between symbolic parsing, connectionist semantic analysis, and symbolic restructuring. Only if the preceding module has finished completely, will the subsequent module start its processing. The architecture and interfaces are relatively simple to realize due to the sequential control between symbolic and connectionist processing. On the other hand, this simple sequential sequence of symbolic and connectionist processing does not support feedback mechanisms.

There are several other loosely coupled hybrid processing architectures. For instance, in the SICSA system, connectionist recurrent networks and symbolic case role parsing are combined for semantic analysis of database queries [4]. In BerP, connectionist feedforward networks for speech processing and symbolic dialog understanding are combined in a spoken language dialog system [30, 65]. In FeasPar, connectionist feedforward networks for feature assignment are combined with symbolic search for finding better analysis results. As in the WP model, in all these loosely coupled hybrid processing architectures the connectionist module has to finish before the symbolic module can start (and vice versa).

5.2 Tightly coupled hybrid processing architectures

A *tightly coupled hybrid architecture* allows multiple exchanges of knowledge between two or more modules. Processing is still a single module at a time, but the result of a connectionist module can have a direct influence on a symbolic module (or vice versa) before it finishes its global processing. For instance, CDP is a system for connectionist deterministic parsing [32]. While the choice of the next action is performed in a connectionist feedforward network, the action itself is performed in a symbolic module (see figure 8). During the process of parsing, control is switched back and forth between these two modules, but processing is confined to a single module at a time. Therefore, a tightly coupled hybrid architecture has the potential for feedback to and from modules.

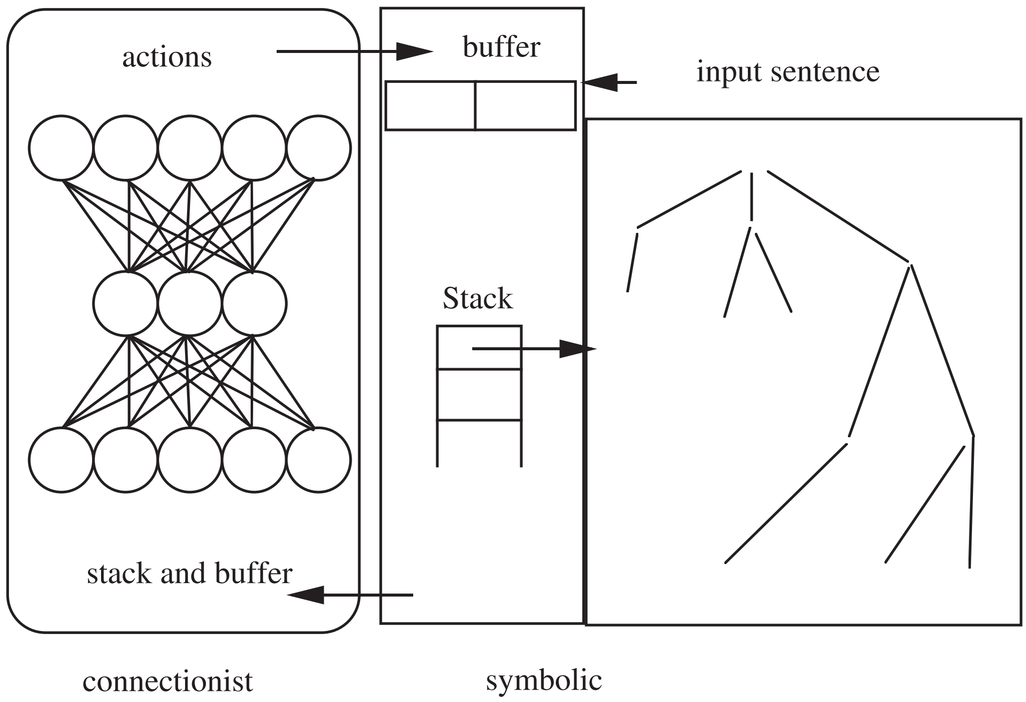

Figure 8: Tightly coupled symbolic deterministic parser and connectionist control network

Other tightly coupled hybrid processing architectures include [23, 22] where the control changes between symbolic marker passing and connectionist similarity determination. In ProPars two connectionist feedforward networks are loosely coupled with a symbolic module for syntactic shift reduce parsing [45]. These tightly coupled hybrid processing architectures interleave symbolic and connectionist processing at the module level and allow the control switch between these different modules. Therefore, this tight coupling has the potential for more powerful interactions. On the other hand, the development of such architectures needs more dynamic and more complex interfaces in order to support the dynamic control between symbolic and connectionist modules.

5.3 Fully integrated hybrid processing architectures

The modules in a *fully integrated hybrid architecture* have the same interface. They are embedded in the same architecture and externally there is no way to distinguish a symbolic from a connectionist module. The control flow may be parallel and the communication between symbolic and connectionist modules is via messages. This is the most integrated and interleaved version of the hybrid processing architectures.

One example of an integrated hybrid architecture is SCREEN which was developed for exploring integrated hybrid processing for spontaneous language analysis. In order to give the reader an impression of the state of the art of current hybrid language technology we focus on this architecture in slightly more detail. Here we focus primarily on the architectural principles of this approach while other task-related details can be found in [58, 60, 62, 63]. One main architectural motivation is the use of a common interface between symbolic and connectionist modules which are externally indistinguishable. This allows incremental and parallel processing involving many different modules. Besides this architectural motivation there are also other task-oriented motivations of exploring learning and fault-tolerance in a hybrid connectionist architecture for robust processing of faulty spoken language.

SCREEN consists of many modules which can be grouped into six organizational parts (see figure 9). The input to the system consists of word hypotheses from a speech recognizer which, typically, provides many word hypotheses for a given signal segment. Each word hypothesis is assigned a plausibility value, a start time, and an end time. Such word hypotheses have to be connected to word hypothesis sequences, and this subtask is performed in the word hypothesis construction component. Since this is a constructive problem and since symbolic operations perform well on such constructive problems, we used a symbolic mechanism for performing this task incrementally on incoming word hypotheses.

During this process of word hypothesis sequence construction it is possible to evaluate many candidate sequences using the word hypothesis evaluation component. Depending on the plausibility of a sequence of semantic and syntactic categories, word hypothesis sequences with a high plausibility are retained while sequences with a lower plausibility are eliminated. Since this subproblem relies on the continuous plausibility of syntactic or semantic category sequences and since connectionist networks support such plausibility evaluations, we used connectionist networks to evaluate these sequences.

The retained word hypothesis sequences are analyzed incrementally in the component

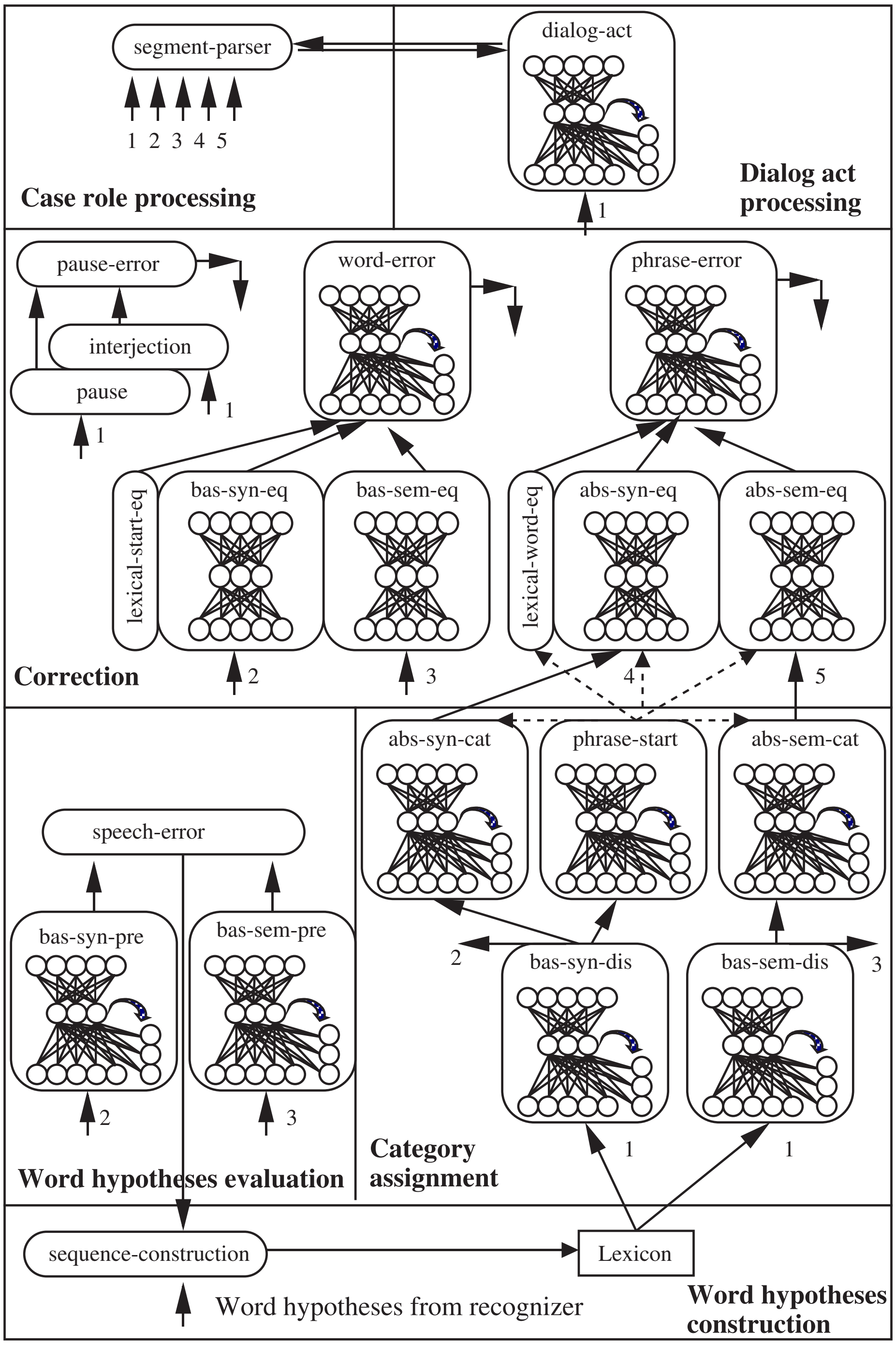

Figure 9: SCREEN: an integrated hybrid processing architecture for spoken language analysis. The following abbreviations are used: abs(tract), bas(ic), dis(ambiguation), cat(egorization), syn(tactic), sem(antic), pre(diction), eq(uality). So for instance bas-sem-dis is the module for basic semantic disambiguation. See text for more details.

for category assignment. In particular, this component performs a syntactic and semantic category assignment. Each word hypothesis within each sequence is assigned the most plausible basic syntactic, abstract syntactic, basic semantic, and abstract semantic categories. Basic syntactic categories describe a word hypothesis sequence at a level of noun, adjective, verb, determiner, etc while abstract syntactic categories describe a sequence at a phrase level as noun phrase, verb phrase, prepositional phrase, etc. Similarly there are basic semantic categories and abstract semantic categories which will depend on the specific domain. For instance in a corpus for scheduling meetings basic semantic categories could be "meet", "select", "animate object" etc, while abstract semantic categories could be "action" "object", etc. Furthermore, the word hypothesis sequences are separated into phrases in order to support various other tasks.

All these category assignments can be learned in recurrent connectionist networks based on examples. The use of connectionist networks is desirable in order to reduce the knowledge acquisition effort. Furthermore, the data driven learning from connectionist networks allows the category assignment component to learn regularities from the data which otherwise may not be manually encoded. Such unforeseeable regularities are particularly important when working with noisy spoken language data. These were in fact the reasons why we used connectionist networks for category assignment.

Spoken language may contain "errors" such as interjections like "oh", "eh", repetitions, corrections, false starts, and all kinds of syntactic and semantic irregularities. While it is impossible to predict all possible irregularities there are some classes of "error" which occur relatively often. Therefore we model the elimination of interjections, word and phrase errors in the correction component, since these are the most frequent mistakes. Some error detections can be supported by lexical comparisons, for instance the lexical equality of two subsequent words may support the hypothesis that there is a repetition (the the meeting...). Such lexical comparisons can be encoded directly in a symbolic manner and therefore they are realized in symbolic modules. On the other hand a comparison between two real-valued vectors of basic syntactic categories could be realized in a connectionist module.

So far we have dealt with words and phrase sequences. However, it is also important to understand complete sentences and turns where a turn is the whole sequence of utterances before the next speaker begins. Boundaries between the utterances in a turn have to be detected in order to trigger different frames for different utterances. Furthermore, the analyzed constituents have to be transfered to case roles in order to be ready for later retrieval. All this is performed in a case role parser. Since there is a lot of explicit control necessary to pull out the required knowledge and fill the case frame incrementally from the underlying modules, this component is realized symbolically.

In addition to the syntactic and semantic knowledge from the category assignment component, there is also knowledge at the dialog level which plays an important role. Each utterance can be assigned to a certain dialog act, for instance, that a certain utterance is a request. As with syntactic and semantic category assignment, this knowledge can be learned in connectionist networks in the dialog processing component.

Before we present an example of processing in SCREEN, we summarize the categories for our flat analysis in table 1. The categories belong to five different knowledge sources: a basic syntactic and a basic semantic word description, an abstract syntactic and an abstract semantic phrase description, and a dialog act description.

basic syntax	basic semantics	abstract syntax	abstract semantics	dialog act
noun	**sel**ect	verb group	**action**	**acc**ept
adjective	**sug**gest	noun group	**aux**-action	**query**
verb	**meet**	adverbial group	**agent**	**rej**ect
adverb	**utter**	prepositional group	**obj**ect	**re**quest-**c**omment
preposition	**is**	conjunction group	**recip**ient	**re**quest-**s**uggest
conjunction	**have**	modus group	**instr**ument	**state**
pronoun	**move**	special group	**manner**	**sug**gest
determiner	**aux**iliary	interjection group	time-**at**	**misc**ellaneous
numeral	**question**		time-**from**	
interjection	**phys**ical		time-**to**	
participle	**anim**ate		location-**at**	
other	**abstract**		location-**from**	
pause (/)	**here**		location-**to**	
	source		**conf**irmation	
	destination		**neg**ation	
	location		**quest**ion	
	time		**misc**ellaneous	
	negative evaluation (**no**)			
	positive evaluation (**yes**)			
	unspecific (**nil**)			

Table 1: The categories for the flat analysis. Abbreviations are depicted in bold face.

SCREEN is an integrated architecture since it does not crucially rely on a single connectionist or symbolic module. Rather, there are many connectionist and symbolic modules, but they have a common interface, and they can communicate with each other in many directions. For instance, figure 9 gives an impression for many communication paths between various modules using arrows and numbers. From a module-external point of view it does not matter whether the internal processing within a module is connectionist or symbolic. In fact, during the design of the system, several modules, for instance in the correction component, were replaced by their symbolic or connectionist counterparts. This architecture therefore exploits a full integration of symbolic and connectionist processing at the module level. Furthermore, the modules can run in parallel and produce an analysis in an incremental manner.

Now we will give an example of how a simple spoken sentence is processed. Figure 10 shows the system environment of SCREEN. Word hypothesis sequences are depicted horizontally. They are ordered in a decreasing manner according to their combined acoustic, syntactic and semantic plausibility, that is, the top word hypothesis sequence represents the highest ranked word hypothesis sequence at that time step. Word hypotheses are processed incrementally, and for each new possible word hypothesis there is an evaluation of whether the best n possible word hypothesis sequences can be continued.

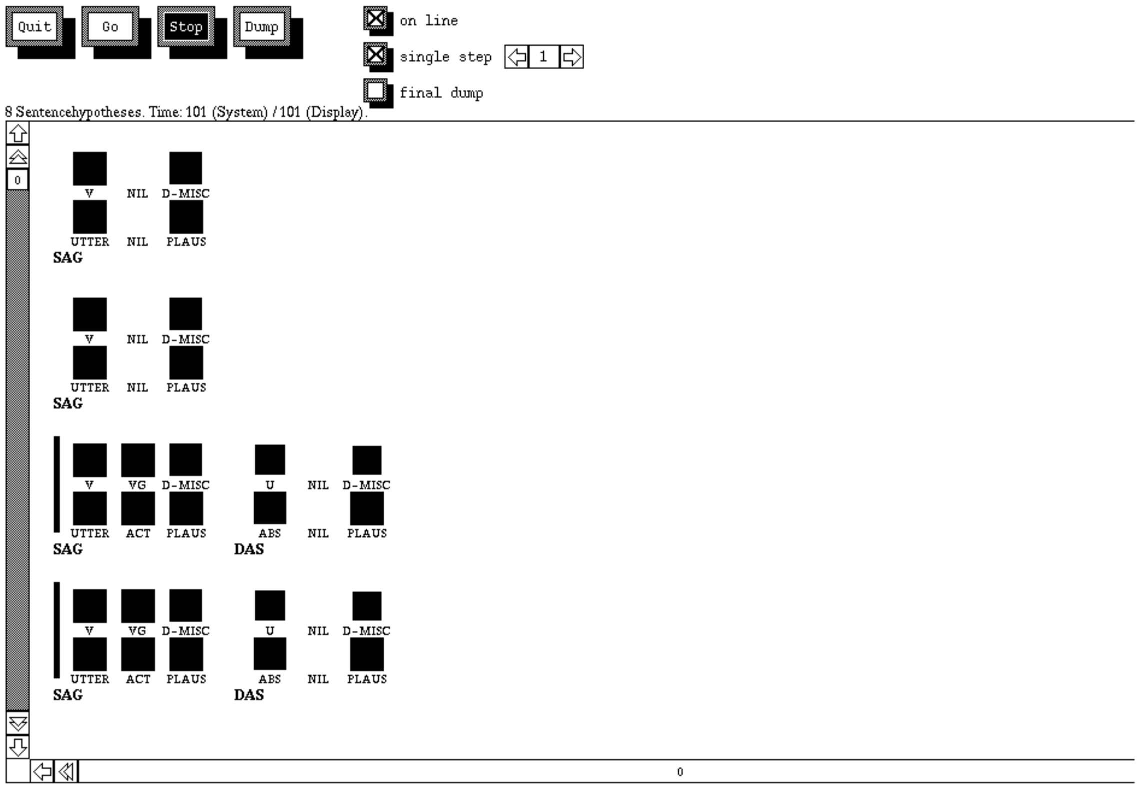

Figure 10: SCREEN snapshot 1 for sentence “sag das doch gleich” (literal translation: say that yet immediately). See table 1 for abbreviations.

In analyzing the incremental word hypotheses it is possible to stop processing (Stop), to start (Go) and to continue processing in different steps (single, multiple step). Furthermore we can store the current internal state of all categories. Using the left vertical scroll bar we can inspect further lower ranked word hypothesis sequences, using the lower horizontal bar we can view arbitrarily long word hypothesis sequences.

The example sentence in German is “sag das doch gleich” (literal translation: say that yet immediately). This sentence is processed incrementally and the state after 1010ms is shown in figure 10. Each word hypothesis in a sequence is shown with the activations of the highest ranked category assignments for that word hypothesis, as computed by the underlying connectionist networks (for a list of abbreviations see table 1). In general, for each word hypothesis six boxes are shown: syntactic basic category (upper left), semantic basic category (lower left), abstract syntactic category (upper middle), abstract semantic category (lower middle), dialog act category (upper right), and plausibility of word hypothesis sequence (lower right).

For instance, if we focus on the first word hypothesis "sag" (say) of the third word hypothesis sequence "sag das" (say that), the word hypothesis "sag" has the basic syntactic category V (verb), the basic semantic category of an utterance (UTTER), the abstract syntactic category of a verbal group (VG), the abstract semantic category of an action (ACT), the dialog act category miscellaneous (MISC), and a particular combined acoustic/syntactic/semantic plausibility value (PLAUS). After 2760ms we can see the final analysis in figure 11. The higher ranked word hypothesis sequences are also the desired sequences. The two lower word hypothesis sequences show an additional undesired "hier" (here) at the end of the sequence.

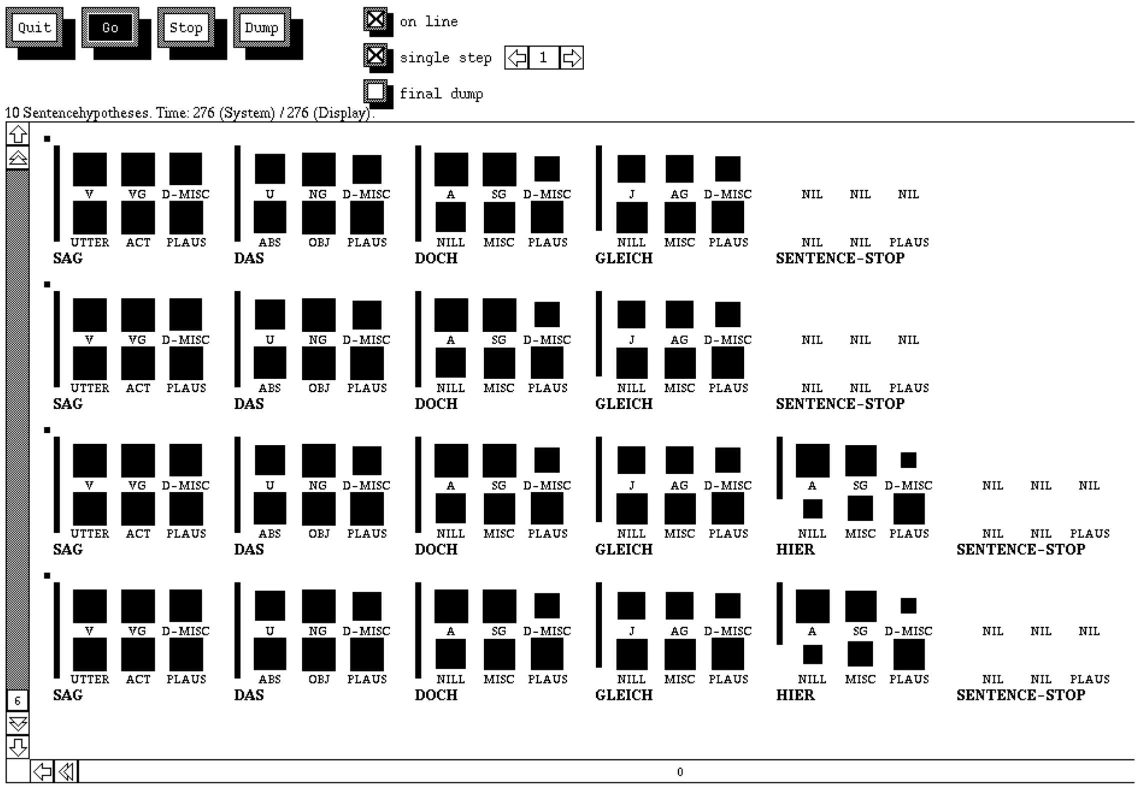

Figure 11: SCREEN snapshot 2 for sentence "sag das doch gleich" (literal translation: say that yet immediately). See table 1 for abbreviations.

Although we cannot go into all the details of the whole hybrid system, we believe it is important to demonstrate the applicability of hybrid techniques in real world environments of language processing. For more details the interested reader is referred to [58, 60, 62, 63]. During the development of the SCREEN system we have taken advantage of the integrated hybrid processing architecture which provides the same interface independently of whether it contains a connectionist or symbolic module. For instance, in the correction part or the case role part of the architecture we have interchanged symbolic and connectionist versions of the same module. Our goal was to use connectionist techniques wherever possible due to their advantages for robustness and learning capabilities but to use also symbolic techniques wherever convenient. While all modules have a symbolically interpretable interface the connectionist or symbolic techniques are hidden within the modules.

Other integrated architectures can be found in PARSEC, a system for language analysis in the conference registration domain [29], which uses Jordan networks to trigger symbolic transformations within the modules, and CONNCERT [64], which has been developed for technical control problems using a mixture of expert networks controlled by symbolic supervisors.

6 Summary and Conclusions

Recently there has been an increasing interest in hybrid symbolic/connectionist interpretation, combination and integration. We have provided an overview of the foundations of hybrid connectionist architectures and a classification of architectures. Furthermore, we have described some representative and current examples of such hybrid connectionist architectures. From this perspective it is clear that hybrid symbolic/connectionist techniques have become an important class of artificial intelligence techniques. Hybrid techniques can be used successfully for those problems where the tasks require different discrete and analog modes of computation and representation. Natural language processing is one important area for hybrid architectures, since natural language processing can be supported well by symbolic structure manipulations as well as connectionist gradual plausibility, learning, and robustness.

Hybrid architectures constitute a continuum from connectionist structure architectures with symbolic interpretation to hybrid transfer architectures and hybrid processing architectures with symbolic and connectionist representations. Today, hybrid connectionist architectures provide technology for realizing larger real-world systems for natural language processing. For instance, the SCREEN system uses a hybrid processing architecture including many different modules at speech, syntax, semantics, and dialog processing levels. While early work on connectionist and hybrid architectures often had to focus on small tasks and small architectures in order to gain initial knowledge about the possibilities of combination and integration, we now begin to see the potential of hybrid connectionist systems being used for larger real world tasks in natural language processing.

Acknowledgments

This research was funded by the German Research Association (DFG) under Grant DFG We1468/4-1. I would like to thank Jerry Feldman, Lokendra Shastri, Srini Narayanan, David Bailey and the members of the NTL research group at ICSI, Berkeley for their support and discussions. I would like to thank the members of the hybrid connectionist language group at Hamburg University (J. Chen, S. Haack, M. Loechel, M. Meurer, U. Sauerland, M. Schrattenholzer, and V. Weber) for their cooperation as well as many discussions. I am also grateful to H. Moisl who provided detailed comments on a previous draft.

References

[1] D. Bailey and J. Feldman and S. Narayanan and G. Lakoff. Modelling Embodied Lexical Development. In *Proceedings of the Meeting of the Cognitive Science Society*, Stanford, 1997.

[2] J. A. Barnden and K. J. Holyoak, editors. *Advances in connectionist and neural computation theory*, volume 3. Ablex Publishing Corporation, 1994.

[3] L. Bookman. *Trajectories through knowledge space.* Kluwer, Boston, 1994.

[4] Y. Cheng, P. Fortier, and Y. Normandin. A system integrating connectionist and symbolic approaches for spoken language understanding. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1511–1514, Yokohama, 1994.

[5] L. Chrisman. Learning recursive distributed representations for holistic computation. *Connection Science*, 3(4):345–365, 1991.

[6] R. Cooper and B. Franks. How hybrid should a hybrid model be? In *Proceedings of the ECAI Workshop on Combining Symbolic and Connectionist Processing*, pages 59–66, Amsterdam, 1994.

[7] M. W. Craven and J. W. Shavlik. Using sampling and queries to extract rules from trained neural networks. In *Proceedings of the 11th International Conference on Machine Learning*, pages 37–45, Rutgers University, 1994.

[8] J. Diederich and D. L. Long. Efficient question answering in a hybrid system. In *Proceedings of the International Joint Conference on Neural Networks*, Singapore, 1992.

[9] G. Dorffner. *Neural Networks and a New AI.* Chapman and Hall, London, UK, 1996.

[10] M. G. Dyer. Distributed symbol formation and processing in connectionist networks. *Journal of Experimental and Theoretical Artificial Intelligence*, 2:215–239, 1990.

[11] M. G. Dyer. Symbolic neuroengineering for natural language processing: a multilevel research approach. In J. A. Barnden and J. B. Pollack, editors, *Advances in Connectionist and Neural Computation Theory, Vol.1: High Level Connectionist Models*, pages 32–86. Ablex Publishing Corporation, Norwood, NJ, 1991.

[12] M. Fanty. Context-free parsing in connectionist networks. Technical Report 174, University of Rochester, Rochester, NY, 1985.

[13] M. A. Fanty. Learning in structured connectionist networks. Technical Report 252, University of Rochester, Rochester, NY, 1988.

[14] J. Feldman. Structured connectionist models and language learning. *Artificial Intelligence Review*, 7(5):301–312, 1993.

[15] J. A. Feldman and D. H. Ballard. Connectionist models and their properties. *Cognitive Science*, 6:205–254, 1982.

[16] J. A. Feldman, G. Lakoff, D. R. Bailey, S. Narayanan, T. Regier, and A. Stolcke. L_0 - the first five years of an automated language acquisition project. *AI Review*, 8, 1996.

[17] L. Fu. Rule generation from neural networks. *IEEE Transactions on Systems, Man and Cybernetics*, 24:1114–1124, 1994.

[18] S. I. Gallant. *Neural Network Learning and Expert Systems*. MIT Press, Cambridge, MA, 1993.

[19] C. Lee Giles and C. W. Omlin. Extraction, insertion and refinement of symbolic rules in dynamically driven recurrent neural networks. *Connection Science*, 5:307–337, 1993.

[20] S. Goonatilake and S. Khebbal. *Intelligent Hybrid Systems*. Wiley, Chichester, 1995.

[21] D. Grannes and L. Shastri and S. Narayanan and J. Feldman. A connectionist encoding of schemas and reactive plans. In *Proceedings of the Meeting of the Cognitive Science Society*, Stanford, 1997.

[22] J. Hendler. Developing hybrid symbolic/connectionist models. In J. A. Barnden and J. B. Pollack, editors, *Advances in Connectionist and Neural Computation Theory, Vol.1: High Level Connectionist Models*, pages 165–179. Ablex Publishing Corporation, Norwood, NJ, 1991.

[23] J. A. Hendler. Marker passing over microfeatures: towards a hybrid symbolic connectionist model. *Cognitive Science*, 13:79–106, 1989.

[24] M. Hilario. An overview of strategies for neurosymbolic integration. In *Proceedings of the Workshop on Connectionist-Symbolic Integration: From Unified to Hybrid Approaches*, pages 1–6, Montreal, 1995.

[25] S. Hollbach Weber and A. Stolcke. L0: A testbed for miniature language acquisition. Technical Report TR-90-010, International Computer Science Institute, Berkeley, CA, 1990.

[26] S. Hollbach Weber. A structured connectionist approach to direct inferences and figurative adjective noun combination. Technical Report 289, University of Rochester, Rochester, NY, 1989.

[27] V. Honavar and L. Uhr. *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Academic Press, Cambridge, MA, 1994.

[28] R. A. Jacobs, M. I. Jordan, and A. G. Barto. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive Science*, 15:219–250, 1991.

[29] A. N. Jain. Parsec: A connectionist learning architecture for parsing spoken language. Technical Report CMU-CS-91-208, Carnegie Mellon University, Pittsburgh, PA, 1991.

[30] D. Jurafsky, C. Wooters, G. Tajchman, J. Segal, A. Stolcke, E. Fosler, and N. Morgan. The Berkeley Restaurant Project. In *Proceedings of the International Conference on Speech and Language Processing*, pages 2139–2142, Yokohama, 1994.

[31] S. C. Kremer. A theory of grammatical induction in the connectionist paradigm. Technical Report PhD dissertation, Dept. of Computing Science, 1996.

[32] S. C. Kwasny and K. A. Faisal. Connectionism and determinism in a syntactic parser. In N. Sharkey, editor, *Connectionist natural language processing*, pages 119–162. Lawrence Erlbaum, 1992.

[33] R. Maclin. *Learning from Instruction and Experience: Methods for Incorporating Procedural Domain Theories into Knowledge-Based Neural Networks.* PhD thesis, Department of Computer Sciences, University of Wisconsin-Madison, 1995. (Also appears as UW Technical Report CS-TR-95-1285).

[34] C. McMillan, M. Mozer, and P. Smolensky. Dynamic conflict resolution in a connectionist rule-based system. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1366–1371, Chambéry, France, 1993.

[35] C. McMillan, M. C. Mozer, and P. Smolensky. The connectionist scientist game: rule extraction and refinement in a neural network. In *Proceedings of the 13th Annual Conference of the Cognitive Science Society*, pages 424–430, 1991.

[36] L. R. Medsker. *Hybrid Neural Network and Expert Systems.* Kluwer Academic Publishers, Boston, 1994.

[37] L. R. Medsker. *Hybrid Intelligent Systems.* Kluwer Academic Publishers, Boston, 1995.

[38] R. Miikkulainen. *Subsymbolic Natural Language Processing.* MIT Press, Cambridge, MA, 1993.

[39] M. C. Mozer. Neural net architectures for temporal sequence processing. In A. Weigend and N. Gershenfeld, editors, *Time series prediction: Forecasting the future and understanding the past*, pages 243–264. Addison-Wesley, Redwood City, CA, 1993.

[40] M. C. Mozer and S. Das. A connectionist symbol manipulator that discovers the structure of context-free languages. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 863–870. Morgan Kaufmann, San Mateo, CA, 1993.

[41] S. Narayanan. Talking the talk is like walking the walk: a computational model of verbal aspect. In *Proceedings of the Meeting of the Cognitive Science Society*, Stanford, 1997.

[42] C. W. Omlin and C. L. Giles. Extraction and insertion of symbolic information in recurrent neural networks. In V. Honavar and L. Uhr, editors, *Artificial Intelligence and Neural Networks: Steps towards principled Integration*, pages 271–299. Academic Press, San Diego, 1994.

[43] C. W. Omlin and C. L. Giles. Extraction of rules from discrete-time recurrent neural networks. *Neural Networks*, 9(1):41–52, 1996.

[44] J. B. Pollack. On connectionist models of natural language processing. Technical Report PhD thesis, Technical Report MCCS-87-100, New Mexico State University, Las Cruces, NM, 1987.

[45] T. Polzin. Parsing spontaneous speech: a hybrid approach. In *Proceedings of the ECAI Workshop on Combining Symbolic and Connectionist Processing*, pages 104–113, Amsterdam, 1994.

[46] J. E. Rager. Self-correcting connectionist parsing. In R. G. Reilly and N. E. Sharkey, editors, *Connectionist Approaches to Natural Language Processing.* Erlbaum, 1992.

[47] T. Regier. The acquisition of lexical semantics for spatial terms: a connectionist model of perceptual categorization. Technical Report Technical Report, International Computer Science Institute, 1992.

[48] R. G. Reilly and N. E. Sharkey. *Connectionist Approaches to Natural Language Processing.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

[49] L. Shastri. A model of rapid memory formation in the hippocampal system. In *Proceedings of the Meeting of the Cognitive Science Society*, Stanford, 1997.

[50] J. Shavlik. A framework for combining symbolic and neural learning. In V. Honavar and L. Uhr, editors, *Artificial Intelligence and Neural Networks: Steps towards principled Integration*, pages 561–580. Academic Press, San Diego, 1994.

[51] P. Smolensky. On the proper treatment of connectionism. *Behavioral and Brain Sciences*, 11:1–74, 1988.

[52] M. F. St. John and J. L. McClelland. Learning and applying contextual constraints in sentence comprehension. *Artificial Intelligence*, 46:217–257, 1990.

[53] R. Sun. *Integrating Rules and Connectionism for Robust Commonsense Reasoning.* Wiley, New York, 1994.

[54] R. Sun. Hybrid connectionist-symbolic models: A report from the ijcai95 workshop on connectionist-symbolic integration. *Artificial Intelligence Magazine*, 1996.

[55] R. Sun and F. Alexandre. *Proceedings of the Workshop on Connectionist-Symbolic Integration: From Unified to Hybrid Approaches.* McGraw-Hill, Inc., Montreal, 1995.

[56] Ron Sun and L.A. Bookman. *Computational Architectures Integrating Neural and Symbolic Processes.* Kluwer Academic Publishers, Boston, MA, 1995.

[57] D. L. Waltz and J. B. Pollack. Massively parallel parsing: a strongly interactive model of natural language interpretation. *Cognitive Science*, 9:51–74, 1985.

[58] S. Wermter. Hybride symbolische und subsymbolische Verarbeitung am Beispiel der Sprachverarbeitung. In I. Duwe, F. Kurfeß, G. Paaß, and S. Vogel, editors, *Herbstschule Konnektionismus und Neuronale Netze.* GMD, Sankt Augustin, 1994.

[59] S. Wermter. *Hybrid Connectionist Natural Language Processing.* Chapman and Hall, Thompson International, London, UK, January, 1995.

[60] S. Wermter and M. Löchel. Learning dialog act processing. In *Proceedings of the International Conference on Computational Linguistics*, Kopenhagen, Denmark, 1996.

[61] S. Wermter, E. Riloff, and G. Scheler. *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing.* Springer, Berlin, 1996.

[62] S. Wermter and V. Weber. SCREEN: Learning a flat syntactic and semantic spoken language analysis using artificial neural networks. *Journal of Artificial Intelligence Research*, 6(1):35–86, 1997.

[63] S. Wermter and M. Meurer. Building lexical representations dynamically using artificial neural networks. In *Proceedings of the International Conference of the Cognitive Science Society*, Stanford, 1997.

[64] A. Wilson and J. Hendler. Linking symbolic and subsymbolic computing. *Connection Science*, 5:395–414, 1993.

[65] C. C. Wooters. Lexical modeling in a speaker independent speech understanding system. Technical Report TR-93-068, International Computer Science Institute, Berkeley, 1993.