Building Lexical Representations Dynamically Using Artificial Neural Networks

Stefan Wermter (WERMTER@INFORMATIK.UNI-HAMBURG.DE)
Manuela Meurer (MEURER@INFORMATIK.UNI-HAMBURG.DE)
Department of Computer Science
University of Hamburg
Vogt-Koelln-Str. 30
22527 Hamburg
Germany

Abstract

The topic of this paper is the development of dynamic lexical representations using artificial neural networks. In previous work on connectionist natural language processing a lot of approaches have experimented with manually encoded lexicon representations for words. However from a cognitive point of view as well as an engineering point of view it is difficult to find appropriate representations for the lexicon entries for a given task. In this context, this paper explores the use of building word representations during a training process for a particular task. Using simple recurrent networks, principal component analysis and hierarchical clustering we show how lexical representations can be formed dynamically, especially for neural network modules in large, real-world, computational speech-language models.

Introduction

In the last decade a lot of progress has been made in connectionist natural language processing. Many different tasks have been covered and many different forms of representations and architectures have been developed (Kawamoto & McClelland 1986, Wermter 1989, St. John & McClelland 1990, McMillan et al. 1993, Wermter et al 1996). However, most of these architectures are still fairly limited with respect to processing natural language in a "real-world" environment, for instance processing a real-world speech dialog.

Comparing the conversational capabilities of connectionist models and the conversation capabilities of human beings, we think there are several reasons why the performance of connectionist models is still fairly moderate in real-world settings. First, after a decade of ground-level work on essential connectionist learning algorithms and representations we are only now in the position to focus on *larger architectures*. In order to make progress on larger areas of human language processing capabilities we have to go beyond individual tasks.

For instance, for understanding spoken language from conversations we have to integrate speech recognition, syntactic, semantic and pragmatic processing in a robust manner. These subtasks provide different constraints (e.g. robustness at speech, syntax and semantics levels) and we cannot expect that one single small network can handle a large portion of human language processing capabilities (Jain 1991, Wermter & Weber 1997). Therefore, it is essential to focus on larger modular architectures in order to make progress on real-world natural language tasks.

Another second main point of limitation of many connectionist models of natural language processing is their static representation. Typically, for a given task a static connectionist architecture is developed and static representations are used for testing the architecture on this task. However, human language processing in the brain is always influenced by changing input from the environment. Neurons die all the time, we learn and forget all the time; so there is plenty of evidence why it is important to explore dynamic architectures and dynamic representations in connectionist architectures .

In this paper we will mainly focus on how dynamic representations can be developed for a larger hybrid connectionist architecture. In order to examine this issue we have developed a hybrid connectionist architecture SCREEN[1] for analyzing spoken language from real-world conversations on scheduling meetings. SCREEN is built on principles of an incremental flat scanning understanding (Wermter 1995, Wermter & Weber 1997). Input to the system is real-world speech, including errors from the speech recognizer or from humans (hesitations, corrections, repetitions, interjections). Output is a syntactic, semantic and dialog analysis of the spoken sentences. Several properties of human language processing are addressed in this system, for instance robustness for errors and incremental parallel processing of syntax and semantics.

After the development of a first version of a comprehensive architecture, we are now in the position to explore the possibility of forming lexical representations dynamically during learning. Using simple recurrent networks, principal component analysis and hierarchical clustering we show how lexical representations can be formed dynamically, especially for neural network modules in large, real-world, computational speech-language models.

The framework: hybrid connectionist speech parsing

There has been surprisingly little work on developing dynamic lexicon representations using supervised learning techniques. Major exceptions are the symbolic/connectionist recirculation work (Dyer 1991) and the work on DISCERN, a connectionist model for understanding simple written sentences (Miikkulainen 1993). However, most network architectures have used vector representations from a static lexicon. While static representations make it easier to add new entries, static entries may not be cognitively plausible. Furthermore, developing lexical representations automatically integrates learning with representation and reduces the language acquisition effort. Therefore, we will explore to what

[1]Symbolic Connectionist Robust EnterprisE for Natural language

extent representations can be formed dynamically in a real-world spoken language environment.

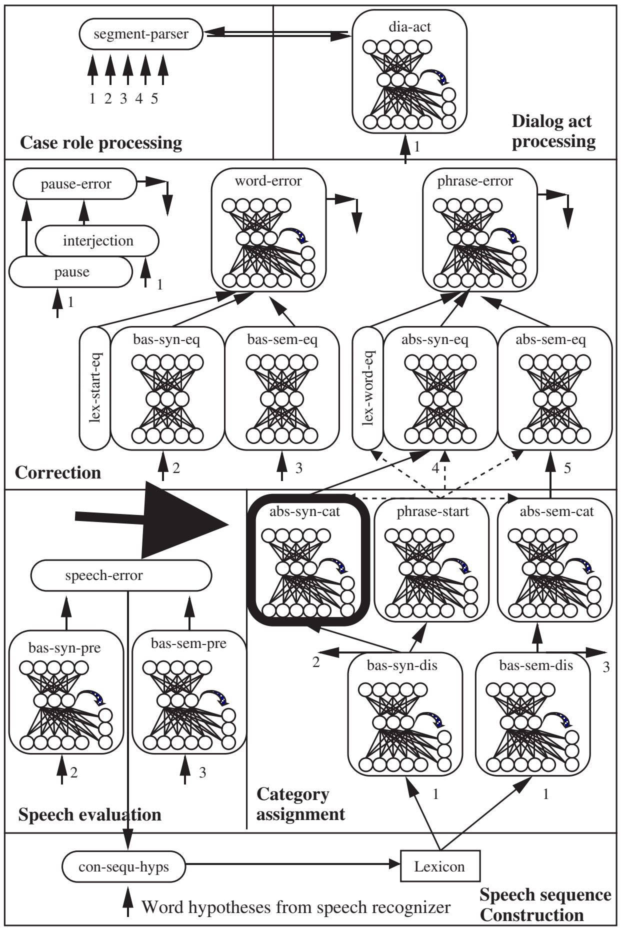

Figure 1: Overall SCREEN architecture. The large arrow shows the subtask of assigning abstract syntactic categories at the phrase level. This subtask is examined for forming dynamic lexical representations automatically.

In this section we will give a brief overview of our SCREEN system (see figure 1). There are three fundamental principles which are addressed in SCREEN based on earlier experience with hybrid connectionist systems (Wermter 1995, Wermter & Weber 1997). First, we want to examine hybrid connectionist *learning* techniques in a real-world speech/language system. Second, we want to explore to what extent hybrid connectionist techniques can provide the necessary *robustness and incremental processing*. Third, we want to examine a *screening approach* to spoken language analysis; that is, rather than an in-depth understanding we aim at a flat, scanning understanding, but we want the understanding to be robust and learned.

In general, our long-term perspective has been to examine the architectural consequences in hybrid connectionist architectures based on these principles. SCREEN consists of six main parts each of which contains several modules. The following description gives just a brief overview of the framework of the SCREEN architecture using a static lexicon; details can be found in (Wermter & Weber 1997). This serves as motivation for the examination of dynamic lexicon representations which are initially explored using the subtask of abstract syntactic categorization (large arrow in figure 1). The data flow in figure 1 is shown by arrows between modules, in some cases we have used numbers to replace arrow drawings that are too complex.

The speech sequence construction part at the bottom of figure 1 receives incrementally single word hypotheses from a speech recognizer and constructs possible partial sentence hypotheses (module con-sequ-hyps). The speech evaluation part at the lower left side contains modules for evaluating individual partial sentence hypotheses and chooses better sentence hypotheses based on acoustic, syntactic and semantic knowledge (bas-syn-pre, bas-sem-pre, speech-error).

Category knowledge is learned and generalized in the category assignment part at the lower right side. Furthermore, phrase starts are detected for identifying phrase boundaries (phrase-start). The category assignment part contains several modules for a flat syntactic and semantic analysis of a current sentence hypothesis (bas-syn-dis, bas-sem-dis, abs-syn-cat, abs-sem-cat). The syntactic and semantic analysis is performed at two syntactic and semantic levels. The large arrow shows the subtask of assigning abstract syntactic categories at the phrase level. This subtask is examined for forming dynamic lexical representations automatically in this paper.

The correction part above contains modules for often occurring mistakes which have to be dealt with explicitly in spontaneous language. For instance, there are modules for detecting interjections and pauses, word repairs, and phrase repairs (pause, interjection, word-error, phrase-error). Furthermore, there are some assistance modules for preprocessing (lex-start-eq, bas-syn-eq, bas-sem-eq, lex-word-eq, abs-syn-eq, abs-sem-eq). The case role part contains a segmentation parser for segmenting complete dialog turns into utterances segments and for filling the contents of a case frame with the utterance constituents (segment-parser). The dialog act part (dia-act) is responsible for recognizing dialog acts of utterances and interacts with the case frame part.

As shown in figure 1, we have chosen primarily feedforward connectionist networks and simple recurrent networks (Elman 1990). Simple recurrent networks were found to be very effective based on their potential for sequential context processing and fault tolerance. Gradient descent is used to train these networks (Rumelhart et al 1986). If a module does not contain a connectionist network it uses simple symbolic rules, for instance for a lexical comparison. We will not go into further details of the architecture which has been described recently in more detail in (Wermter & Weber 1997). Rather, we will now start to focus on our new experiments on forming dynamic representations. In particular, we will describe the development of dynamic syntactic representations for the task of abstract syntactic category assignment (see the large arrow for the module in figure 1).

Focusing on an example: dynamics of syntactic representations

Syntactic analysis can be interpreted as the process of assigning higher abstract syntactic categories (nonterminals) to basic syntactic categories (terminals). In order to support the necessary robustness for spoken language analysis we have used a restricted number of basic and abstract syntactic categories in our domain of meeting scheduling. Table 1 shows the basic syntactic categories and table 2 shows the abstract syntactic categories which have been used in our comparison experiments with manually defined lexicon representations.

Category	Example	Category	Example
noun (N)	date	adjective (J)	late
verb (V)	meet	adverb (A)	often
preposition (R)	at, in	conjunction (C)	and
pronoun (U)	I, you	determiner (D)	the, a
numeral (M)	fourth	interjection (I)	eh, oh
participle (P)	taken	other (O)	particle
pause (/)	pause		

Table 1: Basic syntactic categories

Category	Example
verb group (VG)	mean, would propose
noun group (NG)	a date, the next possible slot
adverbial group (AG)	later, as early as possible
prepositional group (PG)	in the dining hall
conjunction group (CG)	and, either ... or
modus group (MG)	interrogatives, confirmations
special group (SG)	additives: please, then
interjection group (IG)	interjections, pauses: eh, oh

Table 2: Abstract syntactic categories

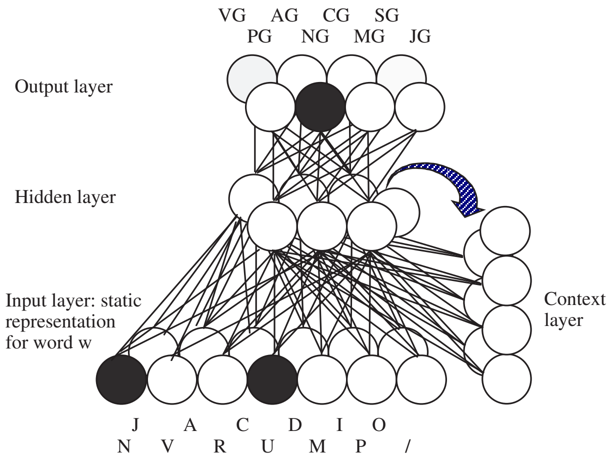

Figure 2: Original simple recurrent network with static lexicon representations as used in the overall SCREEN architecture. Static basic syntactic categories for one word are used as input to the network.

First, we have trained a simple recurrent network to assign abstract syntactic categories to basic syntactic categories in a static lexicon framework (see figure 2). There was one input unit for each basic syntactic category and one output unit for each abstract syntactic category. For a very simple sentence, e.g. "I would suggest eh a meeting on Friday" a basic category representation "pronoun verb verb interjection determiner verb/noun preposition noun" has to be mapped to an abstract category representation "noun-group verb-group interjection-group noun-group prepositional-group". This network (13 input, 7 hidden, 8 output units) had a fairly good performance on a 2300 word corpus reaching 91% category accuracy on the training set and 84% on the unknown test set. However, this network had static lexicon entries.

Using these experiments with static predefined input representations as a bottom line comparison, now we turn our attention to the development of dynamic representations for the task of assigning abstract syntactic categories. The simple recurrent network was modified so that the learning algorithm was able to change the input representations over time. Because of the possibility of developing input representations automatically we can restrict the knowledge which is provided to the network.

While the static network received the basic syntactic category representation as the input and assigned the abstract syntactic category representation as the output for each subsequent word, now we will only provide the abstract syntactic category representation as the output of the network during learning (see the network architecture in figure 3). Then the learning process backpropagates errors to the units in the hidden layer but also back to the input layer. This process allows the learning algorithm to assist in the development of representations which are particular useful for the given task. Over time input representations emerge based on their use in different contexts.

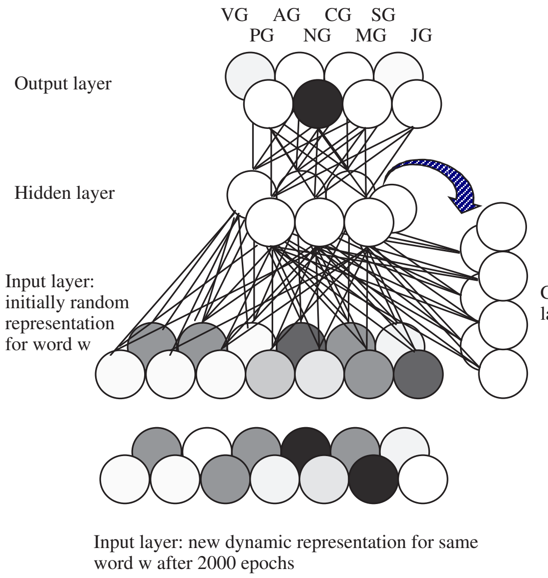

Figure 3: New simple recurrent network with dynamic basic syntactic feature formation. Only the abstract syntactic category knowledge is provided to the network during training. The dynamic input representations are formed dynamically during training.

Now we will describe the augmentations which are necessary for building dynamic representations. The standard backpropagation learning algorithm is a gradient descent learning procedure which minimizes the squares of the differences between actual and desired output values over all output units and all training instances:

$$E = \frac{1}{2} \sum_p \sum_j (d_{pj} - y_{pj})^2 \tag{1}$$

where E is the global error function, p is the pattern of the current training instance, j is the index of the output units, d_{pj} is the desired value and y_{pj} is the current computed value. It has been shown (Rumelhart et al 1986) that this error function E is minimized if the weights are updated according to the following equation:

$$\Delta_p w_{ij} = \eta \delta_{pj} y_{pi} \tag{2}$$

where w_{ij} is the weight from unit i to unit j, η is the learning rate, δ_{pj} is the error associated with unit j, and y_{pi} is the output value of unit i.

The error δ_{pj} for a unit j is computed differently for output units (3) and hidden units (4). This computation minimizes the total sum squared error of equation (1). Furthermore, the function f is a semilinear function, that is, the function f is non-decreasing and differentiable.

$$\delta_{pj} = (d_{pj} - y_{pj}) f'_j \left(\sum_i w_{ij} y_{pi} + \theta_j \right) \tag{3}$$

$$\delta_{pj} = f'_j \left(\sum_i w_{ij} y_{pi} + \theta_j \right) \sum_k \delta_{pk} w_{jk} \tag{4}$$

Up to this point we have the well-known backpropagation learning rule. But it is possible to extend this backpropagation of errors to the input layer (Miikkulainen 1993).

$$r_i = \eta \delta_i = \eta \sum_j \delta_j w_{ij} \tag{5}$$

The value δ_i is the error for unit i of the input layer, δ_j the error for a unit j in the hidden layer and w_{ij} the weight from unit i of the input layer to unit j of the hidden layer. The actual change r_i of an element in the input layer is multiplied with the learning rate η and limited by the interval $[0, 1]$. The new representations $r_i(t+1)$ are calculated by the sum of the old representation $r_i(t)$ plus the change r_i.

$$r_i(t+1) = max(0, min(1, r_i(t) + r_i)) \tag{6}$$

Performance

Using a corpus of 184 turns from real-world spoken conversations about meeting scheduling we trained and tested network architectures with 384 utterances (containing 2356 words). Two thirds of the corpus belonged to the training set, 1/3 to the test set. Using the hand-coded static representations we could reach 91% accuracy on the training set and 84% on the test set (see table 3). An assignment was counted as correct if the abstract syntactic category of the output element with the highest value was also the desired abstract syntactic category.

Input representation	training set	test set
Static, hand-coded	91%	84%
Dynamic, learned	99%	79%

Table 3: Performance for abstract syntactic categorization task

In contrast, the dynamic representation network used initially random lexical representations for the basic syntactic representation of each word. During training only the current abstract syntactic category representation of a word is shown at the output layer. Over time, words with a similar use and distribution in the corpus developed similar representations based on equations 5 and 6.

Comparing the network with the static input representation and the network with the dynamic representation, training performance of the network which used dynamic representations was better. However, the generalization performance on the test set dropped from 84% to 79%. This can be explained by the fact that the dynamic representation network gave the learning algorithm more conceptual freedom. Therefore the input representations were particularly adapted according to the occurring distributions of syntactic category assignments in the training set.

The dynamic representations perform better on the training set, the static representations on the test set. In spite of all differences these percentages for the static and dynamic representations are roughly in the same area. However, it is more important to point out that the network with the dynamic representations received much less knowledge since it did not receive the knowledge about the static basic syntactic categories. Seen from this perspective the necessary tagging effort can be reduced by 50% while still getting a rather similar performance.

Clustering the learned input representations

In order to examine the overall learning effect on the development of the word representations we performed a principal component analysis analysis on all vector representation of the input layer. Figure 5 shows the distribution of the initialized vector representations before learning has started. For this visualization we used the first and second principal component. The word representations are distributed fairly equal in this initial state.

The state after 1000 learning epochs is shown in figure 5. As we can see there is a clear tendency to form clusters and this demonstrates the general effect of learning. Furthermore, at this high level of abstraction of showing all vector representations we can identify two major clouds. These two clouds correspond to the major division between noun-related knowledge and non-noun-related knowledge. That is, the first cloud contains mainly nouns and pronouns like “wir” (English: we), while the second cloud contains other syntactic categories. This main distinction seems to be useful for the learning algorithm since nouns and pronouns occur very often and they occur in similar contexts.

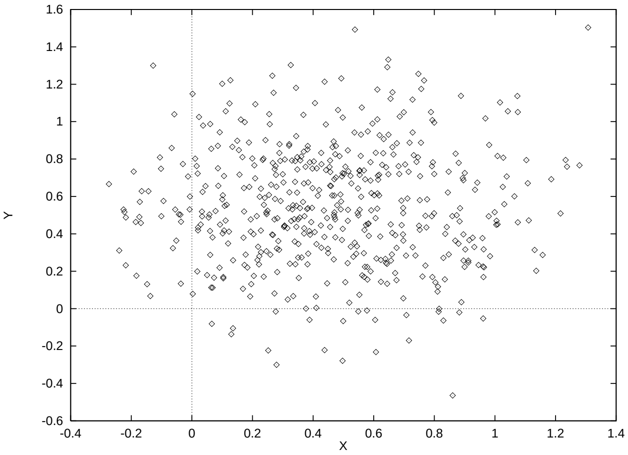

Figure 4: Clustering before learning

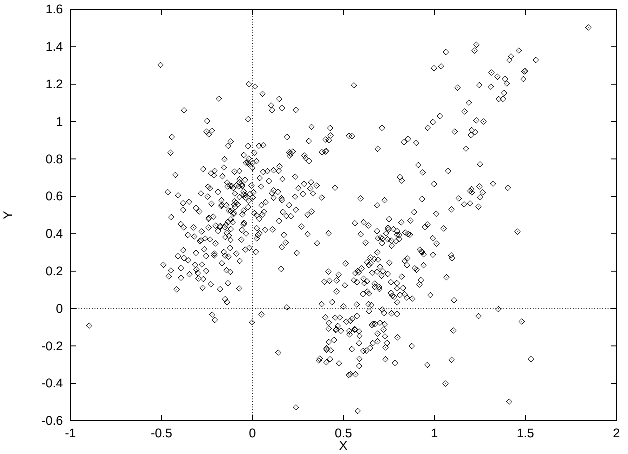

Figure 5: Clustering after 1000 training steps

After we have shown the general learning effect at a higher level of abstraction we now zoom in on a certain example part in order to illustrate the learning effect for individual words. When we analyzed the learned word representations in more detail we found that the network had developed a distributed representation of syntactic categories. A hierarchical cluster analysis on the vectors of the word representation showed that words which belonged to the same basic syntactic category were clustered together. Figure 6 shows a small portion of the hierarchical cluster tree. In this part conjunctions ("und" "dass" "obwohl" are German conjunctions) are clustered together.

Discussion

Although many connectionist models use static lexicon representations, there has been some previous work on dynamic lexicon formation in connectionist natural language processing (Pollack 1988; Dyer 1991; Miikkulainen 1993). There are two main differences between our approach here and this previous work. First, we focus on noisy real-world speech (including various mistakes, etc) rather than well-formed sentence schemata. Second, we use dynamic representations

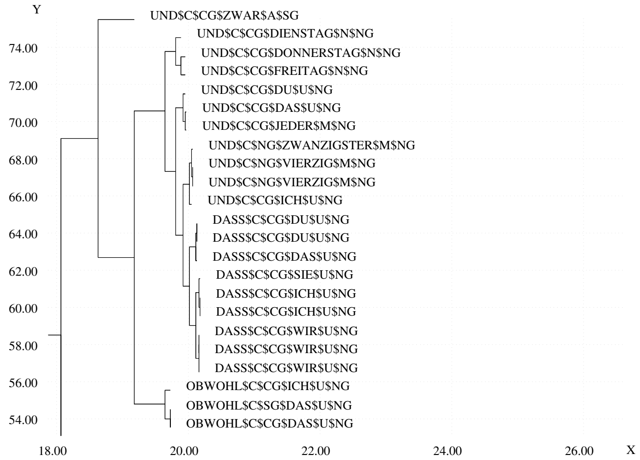

Figure 6: Part of hierarchical cluster analysis of the learned word representations

only at the input layer and guide the network through the output layers.

What have we learned from this? We found that reducing the constraints on the network by forming dynamic word representations leads to a classification improvement for the accuracy on the training set, but also to a deterioration on the test set. The additional degrees of representational freedom are responsible for this behavior. In most cases the dynamic word representations are clustered according to their syntactic basic categories. The networks developed distributed word representations throughout all experiments. We did not find evidence for localist encodings of word representations.

Another important point is that the network which has to form dynamic representations received much less knowledge to perform the classification task. While the static network received both abstract and basic syntactic category knowledge, the dynamic network received only knowledge about the abstract syntactic categories. Therefore much less manual labeling work is necessary. However there is also a performance drop for the test set of the network with the dynamic representations. So we find a tradeoff here. If manual labeling is reliably good, a network with static representations can outperform a network with dynamic representations. However, if manual labeling is expensive or unreliable, a network with dynamic representations with a slightly lower classification accuracy might be a good choice, in particular for developing computational models in a flexible manner.

One important aspect is the question whether manual labeling can be avoided using automatic word representation formation. Our experiments suggest that automatic representation formation is possible for syntactic lexicon construction, and the user has to provide less manually encoded knowledge. In a similar manner, the assignment of categories for semantics or pragmatics should be possible.

On the other hand, the dynamically learned representations are particularly tuned for the particular task and therefore the representations may not be easy to interpret. However, additional means like hierarchical cluster analysis and principal component analysis can assist the user to interpret the learned representations. In contrast, manually determined static lexicon representations can be interpreted naturally and new entries can be defined easily, but manually determined representations may not reflect the exact task knowledge and they are difficult to develop.

We have examined the possibility of learning syntactic representations automatically within a larger real-world spoken language analysis system. Although we do not claim in general SCREEN to be a cognitively valid model of human language processing in general, we think it is important for building computational models to integrate as much cognitively valid aspects of human language processing as possible. While we already focused on incremental processing, robustness, as well as parallel syntactic and semantic processing in previous work, here we examined dynamic word representation formation. Using this motivation from cognitive language processing and knowledge engineering, it is not only possible to improve the coverage of a computational model but also gain insights on certain aspects of cognitive language understanding.

Acknowledgments

This research was funded by the German Federal Ministry for Research and Technology (BMBF) under Grant #01IV101A0 and by the German Research Association (DFG) under Grant DFG Ha 1026/6-3, and Grant DFG We 1468/4-1. We would like to thank V. Weber, S. Haack, M. Löchel, U. Sauerland, M. Schrattenholzer for their work on the SCREEN project.

References

Dyer M. G. (1991). Symbolic neuroengineering for natural language processing: a multilevel research approach. J. A. Barnden & J. B. Pollack (Eds.), *Advances in Connectionist and Neural Computation Theory: High Level Connectionist Models* (pp. 32-86) Norwood, NJ: Ablex Publishing Corporation.

Churchland P. S. & Sejnowski T. (1991). *The computational brain*. Cambridge, MA: MIT Press.

Elman J. L. (1990). Finding Structure in Time. *Cognitive Science*, 14. pp. 179-221.

Jain A. N. (1991). *PARSEC: A Connectionist Learning Architecture Parsing Spoken Language*. PhD thesis, Report CMU-CS-208, Carnegie Mellon University.

McClelland J. L. & Kawamoto A. H. (1986). Mechanisms of sentence processing: assigning roles to constituents. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel Distributed Processing* (pp. 272-326) Cambridge, MA: MIT Press.

McMillan C., Mozer M. & Smolensky P. (1993). Dynamic Conflict Resolution in a Connectionist Rule-Based System. In Proceedings of the International Joint Conference on Artificial Intelligence. Chambery, France. pp. 1366–1371.

Miikkulainen, R. (1993). *Subsymbolic Natural Language Processing*. Cambridge, MA: MIT Press.

Pollack, J. (1988). Recursive auto-associative memory: devising compositional distributed representations. In Proceedings of the Meeting of the Cognitive Science Society.

Rumelhart D. E., Hinton G. E. & Williams R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel Distributed Processing* (pp. 318-362) Cambridge, MA: MIT Press.

St. John M. F. & McClelland J. L. (1990). Learning and applying contextual constraints in sentence comprehension. *Artificial Intelligence*, 46. pp 217-257.

Wermter S. (1989). Learning semantic relationships in compound nouns with connectionist networks. In Proceedings of the Eleventh Conference of the Cognitive Science Society. Ann Arbor, MI. pp. 964-971.

Wermter, S. (1995). *Hybrid Connectionist Natural Language Processing*. London: Chapman and Hall, Thompson International.

Wermter S. & Weber V. (1997). SCREEN: Learning a flat syntactic and semantic spoken language analysis using artificial neural networks. *Journal of Artificial Intelligence Research*, 6. pp 35-85.

Wermter, S., Riloff E. & Scheler G. (1996). *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*. Berlin: Springer.