Connectionist Learning of flat syntactic Analysis for Speech/Language Systems*

Stefan Wermter, Matthias Löchel
Department of Computer Science, University of Hamburg
22765 Hamburg, Germany

1 Learning flat syntactic analysis

There are several key properties which have to be considered for dealing with unrestricted language and speech processing, in particular, learning of regularities, robustness against errors, and an appropriate interpretation level. We argue that *a flat analysis at the phrase group level* supports these key properties for real world speech/language processing and that it can be learned in connectionist networks. A flat representation at the phrase group level structures an utterance U with words w_1 to w_n according to the properties of phrase groups, e.g. according to the abstract syntactic categories like noun group and prepositional group. While complete symbolic syntax trees have a certain potentially incorrect preference, a flat representation only goes as far as possible using only local syntactic knowledge from connectionist networks.

Flat syntactic analysis can be described as a function from basic syntactic categories to abstract syntactic categories. The basic syntactic categories we used were adjective, adverb, conjunction, determiner, noun, preposition, and verb. The abstract syntactic categories were abstract groups, e.g. noun group, prepositional group and verb group. For our experiments we used German and English phrases taken from a real world library corpus. This corpus contained a substantial number of phrases and was used as a representative precursor and testbed for testing flat syntactic analysis for further experiments with unrestricted speech or unrestricted texts. Our task is to train, learn and generalize flat syntactic connectionist representations with abstract syntactic categories based on basic syntactic categories. Flat syntactic analysis is an incremental process where hypotheses about an abstract syntactic category are made based on the previous words and their representation.

2 Evaluating different network architectures

For the training and test sets we computed a *basic syntactic category representation* for each word based on the plausibility of the basic categories for this word. Each word was represented with a vector of seven units where each unit stands for one basic syntactic category. The value of a unit represented the normalized occurrence of this word across the basic syntactic categories in the corpus. While most words have just one syntactic category in our corpus there

*This research was supported in part by the Federal Secretary for Research and Technology under contract #01IV101A0.

are words which belong to several categories, for instance the word "alternative" occurs with the same frequency as an adjective and a noun so that this word is represented with a vector of (0.5 0 0 0.5 0 0 0). The hypotheses about the *abstract syntactic category representation* at the current word are represented based on the five abstract syntactic categories. Each abstract syntactic category is represented by a unit which is on (value 1) if the word belongs to this abstract syntactic category and 0 otherwise.

The training and testing in three architectures was performed with a training set of 541 phrases; 277 phrases were used for training, 264 phrases whose representations were different from the training set were used for testing the generalization performance. Then for each of the three architectures we presented the training set 3000 times to the networks adapting the network states according to the backpropagation learning rule [3]. Figure 1 gives an overview about three different architectures for learning abstract syntactic representations.

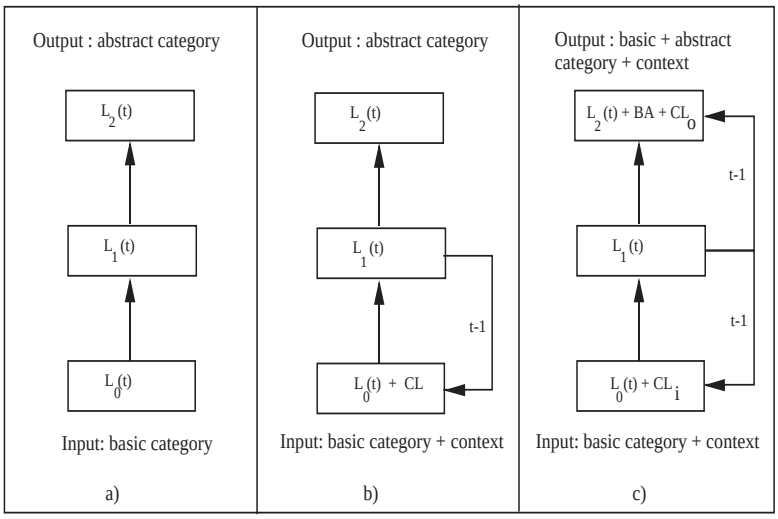

Figure 1: Network architectures: a) F-PN, b) R-PN, c) RAAM network

2.1 The bottom line: F-PN network

As a first architecture we used a simple Feedforward version of Plausibility Networks F-PN. Plausibility networks are general classification networks based on real computed plausibility vectors [4]. The F-PN for syntactic analysis is shown in figure 1 a). The input to the network is the basic syntactic category representation (7 units), the output the abstract syntactic category representation (5 units) and there is one layer of hidden units (6 units). The best tested learning rate was 0.01. The average of the correct word assignments over 3

training runs was only 53.2% on the training set and 56.4% on the test set since the network did not know about the context of preceding words.

2.2 Introducing input context: R-PN network

In a second battery of experiments we extended the F-PN network to a recurrent R-PN by using an additional context layer CL (see figure 1 b))[1]. This context allowed the network to form an internal representation at the hidden layer and to use this context as input to the subsequent decision for an abstract syntactic category. Since the values of the hidden layer are copied to the context layer of the subsequent word the context layer has the same number of six units as the hidden layer. The average performance in the R-PN architecture after 3000 epochs of training was 95.3% on the training set and 90.4% on the unknown test set. This experiment clearly shows the improvement of the performance using additional context and experimentally verifies our initial assumption that additional context is needed for assigning abstract syntactic categories.

2.3 Introducing context as input and output: RAAM

While the R-PN network used context as an extension for the input layer now the hidden layer is not only copied to the context layer at the input layer CL_i but also at the output layer CL_o (see figure 1 c)). This principle of using the hidden layer as context for subsequent input and output representations is characteristic for RAAM networks [2]. Furthermore, we also extended the output layer with the autoassociation of basic syntactic categories BA. In each training step the 7 units for the basic category representation plus the 6 units for the context are associated with the 12 units for basic and abstract categories and 6 context units at the output layer. The performance of this modified RAAM network was 95.2% for the training set and 92.2% for the test set, that is worse on the training set and better on the test set than the R-PN network.

3 Analysis and Discussion

We have shown that the missing context in a FF network dramatically restricts the performance, that a R-PN network reaches good performance and that a much more complex RAAM network provides worse training and better testing results. Since R-PN networks are much smaller and therefore faster to train than RAAM networks, R-PN networks are judged superior to the RAAM architectures for this task of learning abstract syntactic representations for phrases. For the R-PN network, 94.1% of the nouns in noun groups were generalized correctly and 98.1% of the nouns in prepositional groups. This is a very clear example for the learned preceding context and the generalization to the new unknown test data. Especially the prepositional groups and noun groups and their context generalized very well. Assignments with weaker performance, for instance all assignments below 75%, are only underrepresented mappings since only 52 (2.4%) of all mappings between basic and abstract syntactic categories are below 75% correct assignments. The example below illustrates that

[1] R-PN networks differ from SRN networks by [1] by the classification task, their more general network structure [4] and the use of plausibilities for word representations.

the same basic category representations (determiner, noun) can be mapped to different abstract syntactic categories depending on the context (noun group, prepositional group).

```
THE         -> determiner   -> NG (noun group)
DEVELOPMENT -> noun         -> NG
OF          -> preposition  -> PG (prepositional group)
THE         -> determiner   -> PG
AMERICAN    -> adjective    -> PG
MODERN      -> adjective    -> PG
STYLE       -> noun         -> PG
```

In summary and conclusion, we have identified the learning of flat syntactic representations as an important task for language processing. We implemented and tested three different classes of network architectures and found that R-PN networks are most useful for learning flat syntactic representations. Although we experimented with a certain set of abstract syntactic categories the used techniques have not used domain-dependent knowledge and therefore can be ported to different domains and be used for different purposes.

Flat syntactic representations were developed with the long-term goal of supporting the integration of speech and language sources. So far we have concentrated on learning semantic and contextual processing [4], robust fault tolerant processing [5], and syntactic flat analysis (this paper) using flat representations. Flat connectionist representations are very promising for examining the interaction of speech and language since these representations are learnable, fault-tolerant, flat-structured and incremental so that important key properties of speech/language systems can be supported with these connectionist representations.

References

[1] J. L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

[2] J. B. Pollack. Recursive auto-associative memory: devising compositional distributed representations. In *Proceedings of the Tenth Conference of the Cognitive Science Society*, pages 33–39, Montreal, Canada, 1988.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume Vol. 1, pages 318–362. MIT Press, Cambridge, MA, 1986.

[4] S. Wermter. A hybrid connectionist approach for a scanning understanding of natural language phrases. Technical Report Doctoral thesis, University of Hamburg, Hamburg, FRG, 1993.

[5] S. Wermter. Learning natural language filtering under noisy conditions. In *Proceedings of the Tenth IEEE Conference on Artificial Intelligence for Applications*, San Antonio, USA, 1994 (to appear).