CONNECTIONIST REPRESENTED CONTROL KNOWLEDGE FOR PROLOG

A. Ultsch, R.Hannuschka, U.Hartmann, M. Mandischer, V.Weber
Department of Computer Science, University of Dortmund
D 4600 Dortmund 50, PO Box 500 500, Germany
e-mail: ultsch@exunido.uucp

We describe an approach to extract and utilize control knowledge for Prolog proofs by means of connectionist networks. We examined three types of connectionist networks.

DERIVING, LEARNING AND USING CONTROL KNOWLEDGE

Prolog interpreters use a fixed control strategy. Even for simple programs, this implicit control strategy may be not sufficient to obtain efficient computations. A Prolog program can be thought of as consisting of a pure logic program and an encoding of control knowledge. I.e. an encoding of the sequence of successive clause and literal selections. As we want to improve the proof strategy we need the explicit expression of control knowledge. We propose not to formulate this control knowledge explicitly but rather to extract it from the observation of proofs. Basically control knowledge is formulated as a pair of descriptions, one concerning the selection situation and a second of the executed selection. A description consists of the types and identity of the terms and subterms for each argument. This kind of description for the control knowledge is not restricted to a given proof, since the control knowledge forms an abstraction of the executed proof. The system can take advantage of the ability of connectionist networks to generalize and transfer the knowledge to new proofs.

In order to derive and use control knowledge, we built a system which gives us access to the inference process: during a training phase, Prolog proofs are observed, control knowledge is derived, and the networks are trained with these descriptions. After successful training the networks are used to optimize new proofs. In order to generate control knowledge we use a sample Prolog program and a set of queries. The program is scanned to determine the size of the networks. A generating interpreter observes the proof of each query and describes each occurring selection situation. The description of each partial goal and the corresponding selection are derived and the networks are trained with the encoded descriptions. In a working phase, the control knowledge represented in the trained connectionist networks is used to improve the inference strategy for new proofs. Each literal, resp. clause selection situation appearing in a new proof is encoded and passed to the networks. The networks then determine the next literal selection to be made.

USED NETWORKS

We examined three types of connectionist networks to learn control knowledge: Backpropagation, ART1 and Kohonen. In Backpropagation networks we used separated hidden layers of about 150 units each to cope with the different control knowledge features like instantiations and identities of arguments. The sizes of the input and output layer are determined by the size of the Prolog program. For our examples, we found a typical size of 300 to 600 units in input layer and up to 20 units in the output layer. The results presented below are achieved by a network trained for 100,000 epochs.

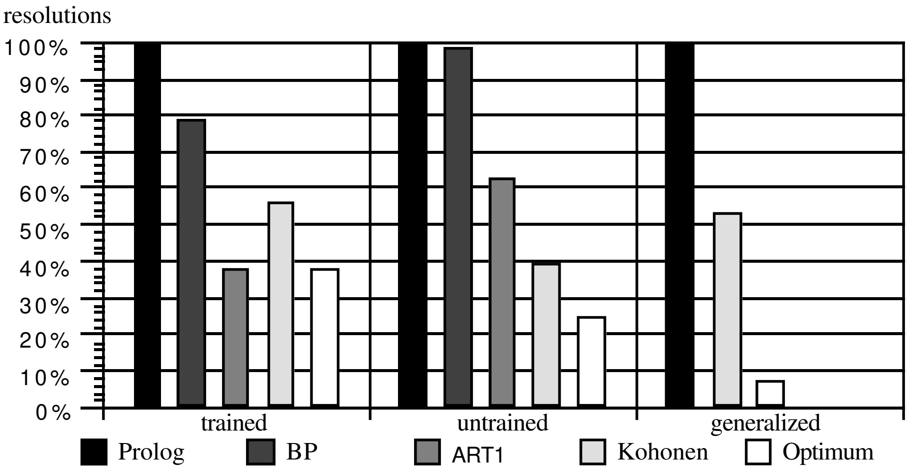
In order to perform supervised learning ART1 was modified in the following way: the output layer is connected with the representation of features of a Prolog Program by the use of a pointer, symbolic mismatch is handled by increasing the vigilance parameter, the bottom-up weights of the network are used as a Competitive Learning Network if the ART1-Network does not compute a convenient output pattern.

As Kohonen's Selforganizing Feature Maps we used a rectangular grid with 30 x 30 units. A training vector consists of two parts: a description of the proof selection situation and the selection that was effective for that situation. A proof selection situation is described by a binary vector of features. The selection is encoded as a sequence of integer numbers indicating the qualification of literals. To determine a selection the description of the selection situation is considered for matching. The vector is completed using the weights of the matched unit. The Feature Maps were trained for 350 epochs.

RESULTS

Using different networks to learn control knowledge, we found that self-organizing feature maps perform this task best. A surprising result is that this network and also the modified ART1-network can perform typical supervised learning tasks better than the backpropagation algorithm. The results depicted in the figure below are derived by example programs performing map-coloring problems. Networks have been trained with proofs of 500 randomly generated queries. The number of resolutions for proofs with Prologs, the networks, and an optimal strategy have been counted for the following sets of queries:

- 500 queries used to generate training patterns
- 500 additional queries generated by random
- 500 queries for a new, but similar program



With the backpropagation network, the performance has not significantly improved. The modified ART1-network is able to recall the trained control knowledge correctly. The cause of this behaviour is that patterns representing control knowledge for a proof are stored separately. The network is also able to generalize to unknown proofs. Results for unknown queries are better than Prologs but worse than those achieved by self-organizing feature maps. These maps are not able to reproduce all the trained proofs correctly, however their performance in generalizing to unknown proofs is quite impressing. Even for a program with a similar structure, which was not used to generate control knowledge, the improvement of resolution steps is satisfying. Our observation suggest a trade off between the ability to generalize and the ability to recall knowledge in connectionist networks. Networks are able to generalize a strategy for clause and literal selections to new, but similar proofs. In particular, we found that appropriately modified unsupervised learning paradigms can perform typical supervised learning tasks better than the standard backpropagation algorithm. An important advantage of this approach is the ability to learn proof heuristics from rather few examples.

[Ultsch 90] A. Ultsch (ed.): Coupling of declarative and connectionist knowledge representation, Res.-Rep.352, Dep. of Computer Science, University of Dortmund, 1990, in German

[Ultsch et al. 90] A. Ultsch, R. Hannuschka, U. Hartmann, V. Weber: Learning of Control Knowledge for Symbolic Proofs with Backpropagation networks, in: R. Eckmiller, G. Hartmann, G. Hauske (eds.); Parallel Processing in Neural Systems and Computers, Elsevier Science Publishers (North Holland), Amsterdam, 1990