Inducing Relational Concepts with Neural Networks Via the LINUS System

Rodrigo Basilio **Gerson Zaverucha** **Artur S. Garcez**

Programa de Engenharia de Sistemas e Computação
COPPE, Federal University of Rio de Janeiro
Caixa Postal 68511
Rio de Janeiro, Brazil - 21945-970
{rbasilio, gerson}@cos.ufrj.br

Department of Computing - Imperial College
London SW7- 2AZ UK
aag@doc.ic.ac.uk

Abstract

This paper presents a method to induce relational concepts with neural networks using the inductive logic programming system LINUS. Some first-order inductive learning tasks taken from machine learning literature were applied successfully, thus confirming the quality of the hypothesis generated by neural networks.

Keywords: Knowledge Acquisition, Architectures, Learning and Generalization

1. Introduction

The application of neural networks has been successfull in a wide range of "real world" tasks, such as speech understanding, handwritten-character recognition, control of dynamic systems and language learning. [Towell, Shavlik 94], [Shavlik et al. 91] and [Thrun et al 91] have favorably compared neural networks with other machine learning methods, specially when data are noisy. These applications strongly suggest that connectionist learning is a powerful approach, and the research of new methods to increase its cognition power merits exploration.

[McCarthy 88] pointed out that neural networks applications have all the basic predicates unary and even applied to a fixed object and a concept is a propositional function of these predicates. Therefore they cannot perform concept description, but only discrimination. An example can be found in [Hinton 86].

Towards relational concept learning [Holldobler, Kurfess 92] and [Shastri, Ajjanagadde 90] presented methods to represent First-Order Logic in neural networks. However, these networks do not have acceptable learning capability.

Knowledge based neural networks by [Towell, Shavlik 94] and [Garcez et al. 96] are propositional theory refinement systems. Propositional rules describing a domain problem are inserted into a neural network, then this network is trained with examples and a refined propositional theory corresponding to it is extracted. The use of prior knowledge has shown to lead to better generalization, specially when there is a reduced number of training examples. [Pazzani, Kibler 92]

[Shavlik 96] pointed out that first-order theory refinement with neural networks is an open problem.

An alternative way to induce relational concepts using neural networks , that is described in this paper, is using the LINUS system [Lavrac, Dzeroski 94], [Lavrac et al 91].

In section two, the LINUS system is reviewed and applied in an example. In section three, LINUS is used with neural networks and applied in the same example. In section four, some conclusions and future work are presented.

2. The LINUS system

LINUS' goal is to use attribute-value propositional inductive learning algorithms, such as ASSISTANT and NEWGEN, to induce relational concept descriptions in a subset of First-Order Logic, namely constrained DHDB (Deductive Hierarchical Database) formalism [Lloyd 87].

Given a DHDB background knowledge and a set of training examples (ground facts of the target predicate), LINUS transforms a first-order problem into an attribute-value form. This information is sent to an attribute-value learning system, which generates an attribute-value hypothesis. Finally, this hypothesis is converted

back to DHDB form. Figure 1 illustrates this process.

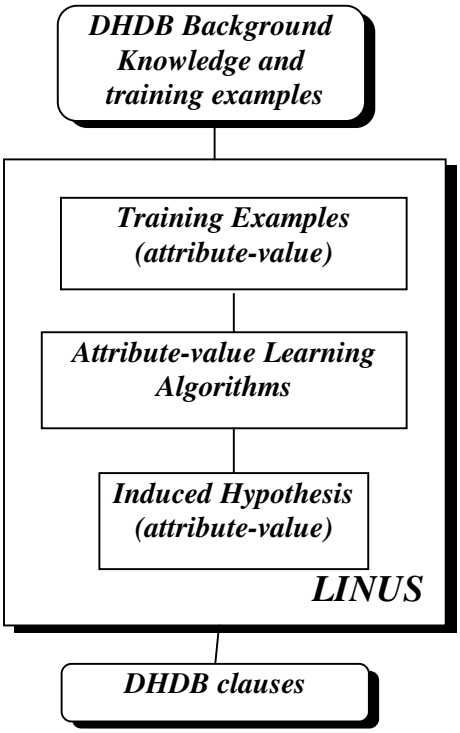

Figure 1. The structure of LINUS.

A simple example of how LINUS works is presented here. We use an example of learning the target predicate mother(x,y) given the background predicates father, wife, son and daughter.

TARGET POSITIVE EXAMPLES (negative examples are generated by Closed world assumption (CWA))	
mother(penelope,arthur)	mother(maria, emilio).
mother(penelope,victoria).	mother(maria, lucia).
mother(christine,jennifer).	mother(lucia,alfonso).
mother(victoria,charlotte).	mother(lucia, sophia).
mother(francesca, marco).	mother(victoria,colin).
mother(francesca, angela).	mother(christine,james)

FATHER PREDICATE	
father(andrew, james).	father(james, colin).
father(andrew,jennifer).	father(chris, arthur).
father(james, charlotte).	father(roberto,lucia).
father(roberto, emilio).	father(pierro,marco).
father(marco, sophia).	father(pierro,angela).
father(marco,alfonso).	father(chris,victoria).

SON PREDICATE	
son(arthur, penelope).	son(colin, james).
son(james, christine).	son(emilio, roberto).
son(colin, victoria).	son(marco, pierro).
son(emilio, maria).	son(alfonso, marco).
son(marco, francesca).	son(alfonso, lucia).
son(james, andrew).	son(arthur, chris).

DAUGTHER PREDICATE
daughter(victoria,penelope).
daughter(jennifer, christine).
daughter(charlotte, victoria).
daughter(lucia, maria).
daughter(angela, francesca).
daughter(sophia, lucia).
daughter(jennifer, andrew).
daughter(charlotte, james).
daughter(lucia, roberto).
daughter(angela, pierro).
daughter(sophia, marco).
daughter(victoria, chris).

WIFE PREDICATE	
wife(penelope, chris).	wife(francesca,pierro).
wife(christine,andrew).	wife(gina, emilio).
wife(margaret,arthur).	wife(lucia, marco).
wife(victoria, james).	wife(angela, tomaso).
wife(jennifer, charles).	wife(maria, roberto).

Table 1. The DHDB Background Knowledge and training examples.

The possible applications of the background predicates on the arguments of the target relation are determined, taking into account argument types. Each such application introduces a new attribute. In our example, all variables are of the same type person. The corresponding attribute-value learning problem is given in table 2.

ATTRIBUTES :
A1 - wife(mother,mother)
A2 - wife(mother, son_daughter)
A3 - wife(son_daughter, mother)
A4 -wife(son_daughter, son_daughter)
A5 -father (mother, mother)
A6 -father (mother, son_daughter)
A7 -father (son_daughter, mother)
A8 -father (son_daughter, son_daughter)
A9 -daughter (mother, mother)
A10- daughter (mother, son_daughter)
A11 - daughter (son_daughter, mother)
A12 - daughter (son_daughter,son_daughter)
A13 - son(mother, mother)
A14 - son (mother, son_daughter)
A15 - son (son_daughter, mother)
A16 - son (son_daughter, son_daughter)

Table 2. The set of attributes generated.

The attribute-value tuples are generalizations (relative to the given background knowledge) of the individual facts about the target relation. The system picks a target ground predicate from the examples, one at a time, and applies its terms in all attributes on table 2. If each resulting ground predicate can be proved by the DHDB, it is set to TRUE, otherwise it is set to FALSE. The application of this procedure to the seventh and ninth ground predicates of the

target positive examples are shown in table 3.

Mother(X,Y)	Son(X,X) (A13)	Son(Y,X) (A15)
(maria, emilio) TRUE	(maria,maria) ***FALSE***	(emilio, maria) ***TRUE***
(lucia, alfonso) TRUE	(lucia,lucia) ***FALSE***	(alfonso,lucia) ***TRUE***

Table 3. Example of setting attribute value.

For the above task, LINUS generates 576 training patterns, 12 positive and the remaining 564 negative.

3. Using LINUS with Neural Networks

This work proposes the use of neural networks as another attribute-value inductive learning algorithm for LINUS.

The set of attribute-values generated by LINUS, as described by table 3, is used as a set of training examples for the neural network.

The method described in [Garcez et al. 96] is used to extract the knowledge from the neural network. Despite the fact that this method is not suitable for large networks, it was satisfactory applied in the above example and in other experiments we have performed. In fact, any rule extraction method [Andrews et al. 95] could be used. All the rules extracted were postprocessed to eliminate irrelevant literals in the body of the rules, before they were returned to LINUS.

Various examples from Machine Learning literature described in [Lavrac, Dzeroski 94; Lavrac et al 91] were used to analyze the performance of neural networks' learning via LINUS: family relationships, the concept of an arch, rules that govern card sequences and the description of robots.

In all the experiments, a standard multilayer feedforward backpropagation neural network of three layers has been used. It is important to notice that any neural network architecture can be used. The number of neurons in the hidden layer was chosen arbitrarily for each experiment, usually 10. The training stopping criteria was 100% of accuracy in the training set performance.

In the above examples, the results showed that the neural networks learned the hypothesis that correctly describes the problem. In the last example, the robot descriptions, besides having a training set it also has a test set. The network learned a hypothesis that classified correctly 100% of the test set.

In the sequel, we apply the above described method in the example of section two.

The resulting propositional hypothesis is:

PROPOSITIONAL HYPOTHESIS (FROM NN)	
mother <-~A6, A15. mother <-~A6, A11.	
The following rules are irrelevant, and removed after theory is postprocessed.	
not_mother<-A2. not_mother <-~A11,~A15. not_mother <-A14.	not_mother <-A7. not_mother <-A6. not_mother <-A3. not_mother <-A10.

The First-Order hypothesis generated by LINUS is:

FIRST-ORDER HYPOTHESIS (BY LINUS)
mother(_mother,_son_daughter) :- not father(_mother,_son_daughter), son(_son_daughter,_mother). mother(_mother,_son_daughter) :- not father(_mother,_son_daughter), daughter(_son_daughter,_mother).

4. Conclusion and Future Work

Theory refinement of first-order theory is an open problem [Shavlik 96]. Differently from [Pinkas 9?] and [Holldobler, Kurfess 92], the method presented here does not generate structured neural networks that represent first-order theories. Alternatively, LINUS is used to transform a first-order theory (in DHDB form) into an attribute-value form, perform a neural learning process and apply a rule extraction algorithm; finally, the knowledge obtained is converted back to DHDB form.

As future works, we intend to apply this method to some real world applications. We also intend to use DINUS [Lavrac, Dzeroski 92], that allows to induce a class of programs slightly larger than LINUS.

It would be interesting to explore the suitability of knowledge based neural networks like KBANN [Towell and Shavlik 94], CIL^{2}P [Garcez et al 96] and CASCADE ARTMAP [Tan, Ah-Hwee 97]. In the last system, the learning process is incremental, local, and preserves the rule structure - this makes easier the extraction task - and the ability to discover new rules. We could also apply the method described in [Menezes et al. 98] for rule extraction from knowledge based neural networks.

Acknowledgements

This work is part of ICOM project, funded by CNPq/ProTeM. The authors are partially financially supported by CAPES and CNPq.

References

[Andrews et al. 95] R. Andrews, J. Diederich and A. B. Tickle; "*A Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks*"; Knowledge-based Systems; Vol. 8, nº 6, pp.1-37; 1995.

[Garcez et al. 96] A. S. Garcez, G. Zaverucha e L. A. Carvalho; "*Logical Inference and Inductive Learning in Artificial Neural Networks*"; Workshop on Neural Networks and Structured Knowledge, ECAI 96, Budapest; 1996.

[Holldobler, Kurfess 92] S. Holldobler and F. Kurfess; "*CHCL: A Connectionist Inference System*"; Parallelization in Inference Systems, B. Fronhofer and G. Wrightson (ed.), Springer LNAI 590, pp.318-342; 1992.

[Hertz et al. 91] J. Hertz, A. Krogh and R. G. Palmer; "*Introduction to the Theory of Neural Computation*"; Addison-Wesley, 1991.

[Hinton 86] Hinton G.E. "*Learning distribuied representations of concepts*". Proc. Eighth Annual Conference of the Cognitive Science Society , Amherst, MA, 1986.

[Lavrac et al. 91] N. Lavrac and S. Dzeroski M.Grobelnik; "*Experiments in Learning Nonrecursive Definitions of Relations in LINUS* "; Technical Report, Jozef Stefan Institute; 1991.

[Lavrac, Dzeroski 92] Lavrac, N. and Dzeroski, S. "*Background knowledge and declarative bias in inductive concept learning*" In Jantke, K., editor, Proc. Third International Workshop on Analogical and Inductive Inference, pp 51-71. Springer, Berlin.

[Lavrac, Dzeroski 94] N. Lavrac and S. Dzeroski; "*Inductive Logic Programming: Techniques and Applications*"; Ellis Horwood Series in Artificial Intelligence; 1994.

[Lloyd 87] J. W. Lloyd; "*Foundations of Logic Programming*" (Second Edition); Springer - Verlag; 1987.

[McCarthy, John 88] J. McCarthy, "*Epistemological challenges for connectionism*", Behavior and Brain Sciences, vol. 11, n. 1; pp. 44; 1988.

[Menezes et al. 98] R. Menezes, G. Zaverucha, V. C. Barboza. "*A Penalty-Function Approach to Rule Extraction from Knowledge-Based Neural Networks*". In this proceedings; 1998

[Minsky 91] M. Minsky; "*Logical versus Analogical, Symbolic versus Connectionist, Neat versus Scruffy*"; AI Magazine, Vol. 12-2, pp.34-51; 1991.

[Pazzani, Kibler 92] Pazzani, M.J. & Kibler, D., "*The utility of knowledge in inductive learning*", Machine Learning, n.9, vol. 1, pp.57-94, 1992.

[Quinlan 90] Quinlan, J.R.; "*Learning logical definitions from relations.*" Machine Learning, n. 5, pp.239-266.

[Shastri, Ajjanagadde 90] L. Shastri and V. Ajjanagadde; "*From Simple Associations to Semantic Reasoning: A Connectionist Representation of Rules, Variables and Dynamic Binding*"; Technical Report, University of Pennsylvania; 1990.

[Shavlik et al. 91] Shavlik J., Mooney R., Towell G., "*Symbolic and neural net learning algorithm: An empirical comparison.*" Machine Learning, n. 6, pp. 111-143., 1991.

[Shavlik 94] Shavlik J.; "*Combining symbolic and neural learning." ,* Machine Learning, n.2, vol. 14, pp.321-331, 1994.

[Shavlik 96] J. W. Shavlik; "An Overview *of Research at Wisconsin on Knowledge-Based Neural Networks" ,* ICNN, pp. 65-69; 1996

[Smolensky 88] P. Smolensky; "*On the proper treatment of connectionism" ,* Behavior and Brain Sciences, vol. 11, n. 1;pp. 1-23; 1988.

[Tan, Ah-Hwee 97] Ah-Hwee Tan; "*Cascade ARTMAP: Integrating Neural Computation and Symbolic Knowledge Processing.*"; IEEE Transactions on Neural Networks vol.8 no.2; pp. 237-250;1997

[Towell, Shavlik 94] G. G. Towell and J. W. Shavlik; "*Knowledge-Based Artificial Neural Networks*"; Artificial Intelligence, Vol. 70, pp.119-165; 1994.

[Thrun et al. 91] S. B. Thrun et al. "*The MONK's Problem: A Performance Comparison of Different Learning Algorithms*"; Technical Report, Carnegie Mellon University; 1991.