30 JAIIO

30 Jornadas Argentinas de Informática
e Investigación Operativa

Simposio Argentino en Inteligencia Artificial

Editado por

Alejandro Ceccatto, Universidad Nacional de Rosario, Argentina

Eduardo Ferme, Universidad de Buenos Aires, Argentina

Buenos Aires, 10 –14 de Septiembre de 2001

Organizado por

SADIO, Socledad Argentina de Informática e Investigación Operativa

Uruguay 252 2º "D" (C1015ABF) - Ciudad de Buenos Aires
Tel: 4371-5755 Tel/Fax: 4372-3950
E-mail: jaiio@sadio.org.ar / sadio@mbox.servicenet.com.ar
Página Web: www.sadio.org.ar

Dealing with Inconsistencies and Knowledge Loss in Combinatorial Neural Model

HÉRCULES ANTONIO DO PRADO[1], PAULO MARTINS ENGEL[2], KÁTIA CILENE DA SILVA[3]

Universidade Federal do Rio Grande do Sul

Instituto de Informática

Av. Bento Gonçalves, 9500 - Bairro Agronomia

Porto Alegre / RS - Brasil

Caixa Postal 15064 - CEP 91.501-970

e mail: {prado, engel, katiacs}@inf.ufrgs.br
telephone: +55(051)3316-6829
fax: +55(051)3316.7308

Abstract

During the last years much effort has been devoted to generate knowledge through Data Mining techniques. Despite of all advances, few efforts have been addressed to cope with post processing problems. This paper is about those problems in Combinatorial Neural Model (CNM). CNM, a supervised learning algorithm introduced by Machado, received many improvements that made it useful for Data Mining. Two main problems are approached. The first one corresponds to the conflicts that can emerge in the knowledge base since the model acquires knowledge from different sources, either specialists or examples. In this way, we apply the concept of extended negation, as the Boolean negation is not so natural. The second problem arises after applying the pruning process to CNM. Since the model is incremental, any part of the knowledge base pruned after the training process in time t_i can be important to the training process in time t_{i+1}. Disregarding this pruned part can lead to loss of knowledge. Considering that it is not possible to avoid pruning, and thus to maintain the knowledge base untouched during all its lifetime, we propose an approach to mitigate the problem.

Keywords: knowledge discovery from databases, data mining, neural networks, inconsistency handling.

[1] Lecturer at Catholic University of Brasilia and researcher at EMBRAPA - Brazilian Enterprise for Agricultural Research
[2] Professor at UFRGS - Federal University of Rio Grande do Sul
[3] M.Sc. Student at UFRGS - Federal University of Rio Grande do Sul

1 Introduction

The issue of post-processing has been frequently mentioned in Data Mining texts [2, 3, 4], although few works has been addressed in this direction, probably as a consequence of the model adherence of this kind of problem. In this paper we focus on the post-processing problems of Combinatorial Neural Model (CNM). This model was proposed by Machado [8] and received many improvements [1, 5, 9, 10, 11, 12, 13, 14] to become suitable for Data Mining. However, there are some problems in this model that were not approached yet. One of these problems are the inconsistencies generated when acquiring knowledge from different sources, either introducing background knowledge to the model or learning from examples. Another problem is the knowledge loss caused by the pruning process. In the next sections, after describing the CNM, we discuss each of these problems, specifying our approach to deal with them.

2 The Combinatorial Neural Model

CNM (see Fig. 2.1) is a hybrid architecture for intelligent systems that integrates symbolic and connectionist computational paradigms. This model is able to recognize regularities from high-dimensional symbolic data, performing mappings from this input space to a lower dimensional output space. Another important advantage of CNM is that the model overcomes the plasticity-stability dilemma [6, 7].

CNM uses supervised learning and a feedforward topology with one input layer, one hidden layer – here called combinatorial layer – and one output layer. Each neuron in the input layer corresponds to an evidence (denoted by e) of the world, expressed in an object-attribute-value form. In the combinatorial layer there are aggregative fuzzy AND neurons (denoted by c), each one connected to one or more neurons in the input layer by arcs with adjustable weights. The output layer contains one aggregative fuzzy OR neuron (also called hypothesis and denoted by h) for each possible class, linked to one or more neurons of the combinatorial layer. The synapses may be excitatory or inhibitory and are characterized by a strength value (weight) between zero (not connected) and one (fully connected synapses).

The network is created completely uncommitted, according to the following steps: (a) one neuron in the input layer for each evidence in the training set; (b) one neuron in the output layer for each class in the training set; and (c) for each neuron in the output layer, there is a complete set of hidden neurons in the combinatorial layer which corresponds to all possible combinations of connections with the input layer. Actually, in order to reduce the combinatorial explosion, the author recommends to adopt the magic number of Miller [8] (seven plus or minus two) for the maximum size of the combinations. Moreover, combinations with size one (just one neuron in the input layer) are connected directly to the hypotheses.Thus, the combinatorial layer keeps the connections with length between two and nine.

The learning mechanism works in one iteration, and it is described below. Note that we are applying the learning mechanism to a topology (with the three layers) already built in one previous step.

PUNISHMENT_&_REWARD_LEARNING_RULE

input: set of examples *E* and the network topology

output: trained network

processing:

begin

for each example from *E*, *do*

propagate the evidence beliefs from input neurons until the hypotheses layer;

for each arc reaching a hypothesis neuron, *do*:

if the reached hypothesis neuron corresponds to the correct class of the case

then backpropagate from this neuron until input neurons, increasing the accumulator of each traversed arc by its evidential flow (Reward)

else backpropagate from the hypothesis neuron until input neurons, decreasing the accumulator of each traversed arc by its evidential flow (Punishment).

end_for;

end_begin.

After training, the value of accumulators associated to each arc arriving to the output layer will be between [-T, T], where T is the number of cases present in the training set. The last step is the pruning of the network; and it is performed through the following actions: (a) remove all arcs whose accumulator is lower than a threshold (specified by a specialist); (b) remove all neurons from the input and combinatorial layers that became disconnected from all hypotheses in the output layer; and (c) make weights of the arcs arriving at the output layer equal to the value obtained by dividing the arc accumulators by the largest arc accumulator value in the network. After this pruning, the network becomes operational for classification tasks.

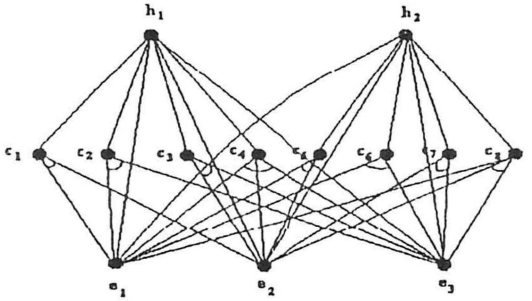

Fig. 2.1. Combinatorial neural model

3 Dealing with Inconsistencies

In real cases, the pure Boolean negation is not adequate to deal with inconsistencies, since a negation of a concept *c* is not accomplished by the simple concatenation of a *not* signal (like ¬, e.g.) before the concept symbol. In the medical domain, where CNM was first applied, it is common to have a set of concepts that are incompatible among them. For example, one physician may conclude by *leichmaniosis* on the basis of some symptoms but another physician may disagree in one symptom. It is necessary to deal with this not only by expressing an overall probability, but also showing to the specialists the disagreement. The model as used does not care about this problem. Actually, by computing the overall confidence (based on beliefs and disbeliefs) the model masks the problem. In order to cope with the contradiction posed in these terms, we define and treat this relation by means of the Set Theory.

3.1 The Basic Process

For terminology convenience, we call both evidences and hypotheses in CNM as *conditions* and define the extended negation as *incompatibility* among conditions. It is justified by the fact that we can have conflicts among any kind of conditions (hypotheses or evidences).

First we define the notion of *incompatibility* with respect to two conditions. Then, based on this notion, it is defined the set of conditions that are incompatible with a specific condition. Finally, we define the set of conditions that are incompatible with another set of conditions. This definition is necessary, since we are interested in identifying inconsistencies in rules, which can involve many conditions.

Definition 1: Incompatibility - A condition y is incompatible with a condition x if they cannot be simultaneously true.

Definition 2: Incompatibility class of a condition x - The incompatibility class of a condition x, denoted by $I(x)$, is composed by all conditions that are incompatible with x.

Definition 3: Extended incompatibility class of the set $X=\{x_1,x_2,...,x_n\}$ – The extended incompatibility class of the set $X=\{x_1,x_2,...,x_n\}$, denoted by $EI(x_1,x_2,...,x_n)$, is defined by the set of conditions $Y=\{y_1, y_2 ..., y_m\}$, where $Y=I(x_1)\cup I(x_2)\cup...\cup I(x_n)$.

Naturally, the specialists must define the incompatibility classes and the system computes the extended incompatibility classes. The following algorithm performs the detection of incompatibilities, on the basis of the extended incompatibility classes:

INCOMPATIBILITIES_DETECTOR_ALGORITHM

input: Set of rules defined by a CNM and the incompatibility classes for all conditions in the rules;

output: Set of rules with incompatibilities;

processing:

begin

for each rule $R(E,h)$, with E the set of evidences and h the consequence, do

if $\{E,h\}\cap EI(E,h)\neq\varnothing$ *report* "inconsistent rule: type 1";

end_for;

for each unordered pair of rules $R_1(E_1,h_1)$ and $R_2(E_2,h_2)$, *do*

if $E_1 \subseteq E_2$ *and* $\{E_1,h_1,E_2,h_2\} \cap EI_1(E_1,h_1,E_2,h_2) \neq \varnothing$ *report*

"inconsistent rules R_1 and R_2: type 2";

end_for;

end_begin.

Basically, the algorithm detects two kinds of inconsistencies. The first one(type 1) occurs inside the conditions of each rule. The second one (type 2) occurs between two rules that eventually are triggered simultaneously.

To illustrate, suppose we have the rules:

$R_1=((e_1,e_2,e_3),h_1)$,

$R_2=((e_4,e_5,e_6),h_2)$, and

$R_3=((e_1,e_2),h_3)$

the incompatibilities:

$I_1(e_1)=\{e_7\}$,

$I_2(e_4)=\{e_8\}$,

$I_3(h_3)=\{e_3\}$, and

$I_4(h_2)=\{e_5\}$

and the extended incompatibility classes of interest for us:

$EI_1(e_1,e_2,e_3,h_1)=\{e_7\}$,

$EI_2(e_4,e_5,e_6,h_2)=\{e_5,e_8\}$,

$EI_3(e_1,e_2,h_3)=\{e_3,e_7\}$, and

$EI_4(e_1,e_2,e_3,h_1,h_3)=\{e_3,e_7\}$

Since e_5 is part of EI_2 and argument for EI_2, R_2 is reported as " inconsistent rule: type 1". The pair R_1 and R_3 is reported as "inconsistent rules R_1 and R_3: type 2". It happens because the evidences of R_3 is a subset of the evidences in R_1, and that e_3 is part of EI_4 and argument for EI_4.

4 Preventing from Knowledge Loss

The pruning process in CNM is based on the accumulator values in the arcs arriving to the hypothesis neurons. By this process, the combinatorial neuron in which the connection to the hypotheses layer has an accumulator value smaller than a pre-defined threshold is discarded. All arcs arriving to this neuron are also discarded. For example, applying the training set in Table 1 to the CNM in Fig. 4.1 (the corresponding rules are shown in Table 2) we obtain the model in Fig. 4.2.

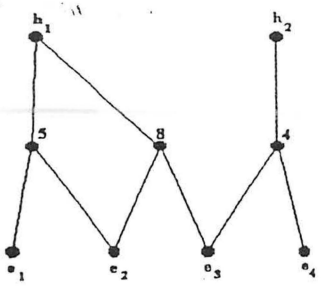

Fig. 4.1. Example of a CNM

Table 4.1. Training set

Case	Symptom	Disease
1	e_2, e_3	
2	e_2, e_3	h_1
3	e_2, e_3	
4	e_2, e_4	
5	e_2, e_4	
6	e_2, e_4	
7	e_2, e_4	
8	e_2, e_4	h_2
9	e_2, e_4	
10	e_2, e_4	
11	e_2, e_4	
12	e_2, e_4	
13	e_2, e_4	

Table 4.2. Rules corresponding to Fig. 4.1

Id.	Rule
1	If e_1 and e_2 Then h_1
2	If e_2 and e_3 Then h_1
3	If e_3 and e_4 Then h_2

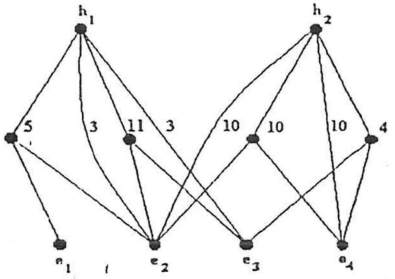

Fig. 4.2. Example after updating

Considering a threshold of 8, which is reasonable in face of the lower and upper bounds of the accumulators, the new shape for the CNM is depicted in Fig. 4.3. The corresponding rules are shown in Table 3.

Table 4.3. Rules corresponding to Fig. 4.3

Id.	Rule
1	If e_2 and e_3 Then h_1
2	If e_2 and e_4 Then h_2
3	If e_4 Then h_2

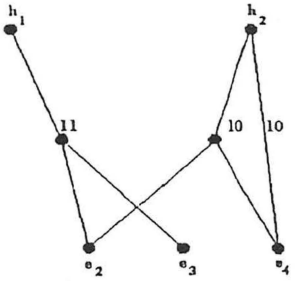

Fig. 4.3. Example pruned after updating

Comparing Tables 2 and 3, we note a drastic change in the knowledge, making established beliefs to disappear completely. This change may represent a valid knowledge update, exhibiting a non-monotonic characteristic of CNM, or maybe an inconsistency. In any case, it is necessary to review all decisions depending on the changed knowledge. Ideally, it would be interesting to avoid pruning, and keep all accumulators, since any of them can play an important rule in next updating. Therefore, in consequence of computational limitations, the pruning is unavoidable. To mitigate the problem, we propose a solution that keeps track of two states s_i and s_{i+1} of the knowledge base, and submit the modifications to the user judgment. It works like an alarm to support the user in reviewing her(his) beliefs.

Our solution for this problem consists in obtaining the set $D = B_{i+1} - B_i$ which contains the modifications that has occurred in knowledge base B from instant $i+1$ to i. This operation is accomplished by the following algorithm:

KNOWLEDGE_CHANGING_ALGORITHM

input: Knowledge bases B_{i+1} and B_i;

output: Set D of differences between B_{i+1} and B_i;

processing:

begin

for each hypothesis h in B_{i+1} and B_i create L_{i+1} and L_i, respectively, lists of all rules in B_{i+1} and B_i *that have* h as consequence;

compare L_{i+1} and L_i, creating D, with the following content:

report new knowledge for the rules in L_{i+1} but not in L_i;

report lost knowledge for the rules in L_i but not in L_{i+1};

end_for;

end_begin.

5 Discussion

CNM has been adopted with success for learning from examples, being able to start from scratch or loaded with background knowledge from specialists. Despite of the practical results, no work had been presented to cope with post-processing of the model. We have identified important problems that can affect the knowledge quality. Namely, we approached the consequences of the pruning process in the incremental nature of the model and the lack of an adequate solution to face the occurrence of conflicts resulting from acquiring knowledge of disparate sources.

The highlighted incremental characteristic of CNM should make it always able to learn from new examples. Actually, when new examples are considered, it would not be necessary to pass the whole data set which originated the model. However, due to the pruning process, part of the structure (with accumulator values smaller than a threshold) is discarded. There is no reason to believe that the knowledge represented by the pruned part will not be important in future to sum with new evidences. It is possible that, with the application of new examples, the accumulators of that part have values greater or equal to the threshold. To be really incremental, CNM should never be pruned, what is unfeasible, considering the requirements of performance. Our solution in controlling the changing in knowledge between two training steps can, at least, reduce the problem.

With respect to the conflicts among conditions in the model, we adopted the notion of extended negation that we regard as more adequate than the simple Boolean negation. By applying this notion, the system can identify conflicts among sets of conditions, instead of only the conflict of a symbol φ and its negated form $\neg\varphi$. The inconsistencies reported by the system can be useful in a

consent phase in which the specialists have to clear up any kind of inconsistency, either in the examples or among themselves.

References

[1] Beckenkamp, F. G.; Feldens, M. A. and Pree, W. Optimizations of the Combinatorial Neural Model. In: BRAZILIAN SYMPOSIUM ON NEURAL NETWORKS, 5., 1998, Belo Horizonte, Brazil. **Proceedings...** Los Alamitos: IEEE Computer Society Press, 1998.

[2] Bigus, J. P. Data Mining with Neural Networks. [S.l.]: McGraw-Hill, 1996. p.3-42.

[3] Michalski, R. S.; Bratko, I.; Kubat, M. Machine Learning and Data Mining: Methods and Applications. New York, NY, 1998. 456pp.

[4] Fayyad, U.; Piatetsky-Shapiro, G; Smyth, P. From Data Mining to Knowledge Discovery: An Overview. In: U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (Eds.) Advances in Knowledge Discovery and Data Mining, Cambridge, MA: AAAI Press, 1996. p.1-34.

[5] Feldens, M.A., Castilho, J.M.V., Data mining with the Combinatorial Rule Model: an application in a health-care relational database. In: CLEI, 23.,1997, Valparaizo, Chile, **Proceedings ...** Valparaízo: CLEI, 1997. V.1, pp.135-145.

[6] Freeman, J. A., Skapura, D. M., Neural Networks, Algorithms, Applications, and Program Techniques. [S.l.]: Addison-Wesley, pp.293-339, 1992.

[7] Haykin, S . Neural Networks, A Comprehensive Foundation. Upper Saddle River, NJ: Prentice Hall, 842pp. 1999.

[8] Machado, R. J. and Rocha, A. F. Handling Knowledge in High Order Neural Networks: The Combinatorial Neural Network. Rio de Janeiro: IBM Rio Scientific Center, 1989. (Technical Report CCR076).

[9] Machado, R. J., Barbosa , V.C., Neves, P. A., Learning in the combinatorial neural model, IEEE Transactions on Neural Networks, v. 9, pp.831-847, 1998.

[10] Prado, H. A.; Frigeri, S. R. and Engel, P. M. A Parsimonious Generation of Combinatorial Neural Model. In: CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACIÓN, 4., 1998, Neuquén, Argentina. **Proceedings...** Neuquén: Universidad Nacional del Comahue, 1998.

[11] Prado, H. A.; Machado, K. F.; Frigeri, S. R.; and Engel, P. M. Accuracy Tuning in Combinatorial Neural Model. In: PACIFIC-ASIA CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 3., 1999, Bejing, China. **Proceedings...** Berlin: Springer-Verlag, 1999. (Lecture Notes in Artificial Intelligence, v. 1574).

[12] Prado, H. A. Explaining Cluster's Structures by Integrating Unsupervised and Supervised Learning Algorithms. In: IEEE WORKSHOP ON DB & IS, 4., 2000, Vilnius, Lithuania. **Proceedings...** Vilnius: Lithuanian Computer Society, 2000.

[13] Prado, H. A.; Hirtle, S. C. and Engel, P. M. Scalable Model for Extensional and Intensional Descriptions of Unclassified Data. In: IPDPS WORKSHOP ON HIGH PERFORMANCE DATA MINING, 3., 2000, Cancún, México. **Proceedings...** Berlin: Springer-Verlag, 2000. (Lecture Notes in Computer Science, v. 1800).

[14] Prado, H. A.; Machado, K. F.; Engel, P. M. Alleviating the complexity of the Combinatorial Neural Model using a committee machine. In: INTERNATIONAL CONFERENCE ON DATA MINING, 2., 2000, Cambridge, UK. **Proceedings...**, Southampton: WIT Press, 2000.