Extraction of crisp logical rules from medical datasets

Włodzisław Duch, Rafał Adamczak, and Krzysztof Grąbczewski
Department of Computer Methods, Nicholas Copernicus University,
Grudziądzka 5, 87-100 Toruń, Poland.
E-mail: duch,raad,kgrabcze@phys.uni.torun.pl

Abstract

Three neural-based methods for extraction of logical rules from data are presented. These methods facilitate conversion of graded response neural networks into networks performing logical functions. MLP2LN method tries to convert a standard MLP into a network performing logical operations. C-MLP2LN is a constructive algorithm creating such MLP networks. Logical interpretation is assured by adding constraints to the cost function, forcing the weights to ± 1 or 0. Skeletal networks emerge ensuring that a minimal number of logical rules are found. In both methods rules covering many training examples are generated before more specific rules covering exceptions. The third method, FSM2LN, is based on the probability density estimation. Results of application of these methods to the hypothyroid and hepar datasets are presented.

I. Introduction

LOGICAL rules should be preferred over other methods of classification, provided that the complexity of the set of rules will not be too large and their accuracy will be sufficiently high. This is especially true in medical applications, where understanding which features contribute to classification and which are irrelevant is of extreme importance. Rules allow for detailed control over complexity of the classification model. Surprisingly, simple logical rules proved to be more accurate and were able to generalize better than many machine and neural learning algorithms [1]. Many methods of logical rule extraction are based on neural networks (for a recent review and extensive references see [2]).

Recently we have introduced several new methods of logical rule extraction and feature selection based on density estimation networks and multi-layered perceptrons (MLPs). Although we concentrate on crisp logical rules these methods can easily find also fuzzy rules. In contrast with the existing neural rule extraction algorithms based on analysis of small connection weights, analysis of sensitivity to changes in input or analysis of the total network function [2] our goal is to create a simplified network with nodes performing logical functions. This goal is achieved either by constraining the multi-layered perceptron (MLP) network that has already been trained, or by constructing the network during training, adding new neurons and immediately simplifying their connections. Third possibility is based on density estimation. Cuboidal density regions are easily interpreted as crisp logic rules, therefore one should select appropriate transfer functions and either convert smooth complicated densities into cuboids or create cuboidal densities directly. These three algorithms are briefly described in the next section. Two applications to the medical data analysis are presented in the third section. The paper is finished with a short discussion.

II. Three rule extraction algorithms

Preliminary: linguistic variables. Logical rules require symbolic inputs (linguistic variables). If the input data components x_i are real numbers finding optimal linguistic variables is a part of the classification problem. Density estimation networks provide linguistic variables from analysis of response of network nodes. The inputs from features giving strong response along the whole range of data are irrelevant and should be removed. The range of values taken by continuous inputs may also be partitioned into distinct (for crisp logic) sets by analysis of class-membership histograms. The initial linguistic variable boundary points are optimized after the rules are found and the whole rule extraction process is repeated with new linguistic variables. MLPs may find linguistic variables using a combination of two neurons, called here an L-unit, performing for each continuous input a "window-type" function:

$$s_*(x;b,b') = \sigma(x-b)(1-\sigma(x-b')); \quad s_+(x;b,b') = \sigma(x-b) - \sigma(x-b') \qquad (1)$$

where the gain of the sigmoidal functions $\sigma(x)$ reaches a very high value during learning, changing $s(x;b,b')$ into a step-like logical function. This function has two biases which may be treated as boundaries defining linguistic variable $(l = \text{true}) \equiv (x \in [b,b'])$ [3].

MLP2LN method. Logical rules giving classification results corresponding to the trained, multi-layered network, are required. MLP network is modified by retraining it while the slopes of sigmoidal functions are gradually increased and the weights simplified. Integer weight values are enforced: 0 for irrelevant inputs, $+1$ for features that must be present and -1 for features that must be absent. This is achieved by modifying the error function:

$$E(W) = E_0(W) + \frac{\lambda_1}{2}\sum_{i,j} W_{ij}^2 + \frac{\lambda_2}{2}\sum_{i,j} W_{ij}^2 (W_{ij}-1)^2 (W_{ij}+1)^2 \qquad (2)$$

where $E_0(W)$ is the standard quadratic error measure, the second term with λ_1 leads to a large number of zero weights, i.e. elimination of irrelevant features, and the third term vanishes for weights equal to 0 or ± 1. Similarly as in the case of weight pruning techniques in the backpropagation algorithm these terms lead to the additional change of weights:

$$\Delta W_{ij} = \lambda_1 W_{ij} + \lambda_2 W_{ij}(W_{ij}^2 - 1)(3W_{ij}^2 - 1) \qquad (3)$$

where λ_1 and λ_2 scale the relative importance of auxiliary conditions. This form of error function has two advantages: independent parameters control enforcing of 0 and ± 1 weights, and an interpretation of this function from the Bayesian point of view [4] is straightforward. It defines our prior knowledge about the probability distribution $P(W|M)$ of the weights in our model M. A network trained on classification tasks should give crisp logical decision "yes", "no" or "irrelevant", therefore *a priori* conditional probability [4] is:

$$P(W|M) = Z(\alpha)^{-1} e^{-\alpha E_a(W|M)} \propto \prod_{ij} e^{-\alpha W_{ij}^2 (W_{ij}-1)^2 (W_{ij}+1)^2} \qquad (4)$$

To find logical rules giving classifications equivalent to those of a trained MLP network new error function Eq. (2) with relatively large values of regularization parameters λ_1, λ_2

is used for retraining. Typical MLP network has many irrelevant connections with small weights and regularization terms initially increase the error. After retraining a skeletal network emerges, with a few connections left. Such network is capable of rough classification performing simple logical operations. This skeletal network is inserted into the original network and its weights and biases are kept frozen. Modified MLP is now retrained again with smaller regularization parameters. After several cycles of retraining and modification the original network is changed into a simplified MLP performing logical operations that approximate original classifications.

C-MLP2LN method. In this approach a single hidden layer skeleton MLP is constructed. One hidden neuron per output class is created and the modified error function minimized during training on all data until convergence is reached. The remaining non-zero weights and the thresholds obtained are then analyzed and the first group of logical rules is found, covering the most common input-output relations. The input data correctly handled by the first group of neurons do not contribute to the error, therefore the weights of these neurons are kept frozen during further training. A second group of neurons is added and trained on the remaining data. This process is repeated until a set of rules $R_1 \vee R_2 ... \vee R_n$ capable of satisfactory classification is found, or until the number of cases correctly classified by a given new neuron drops below certain minimum. Neurons handling only few specific training cases model noise rather than regularities in the data, and thus should be deleted. The output neuron for each class is connected to the hidden neurons created for that class, performing a simple summation of the incoming signals. Logical rules are sometimes simpler if, in a small number of cases, wrong classifications by a newly added neuron is allowed. These cases are treated as exceptions to the rules. The set of extracted logical rules has then *a hierarchical order*. Rules handling exceptional cases, placed at the top of hierarchy, are applied first.

Extraction of rules from skeleton network with small number of integer connections is quite simple. For each neuron non-zero inputs define levels of a search tree, and values of linguistic variables define possible branches of this tree. To each node of the search tree a contribution to the activation of neuron is assigned. Since at each level maximum activation that lower levels may contribute is known, nodes that do not lead to activations larger than threshold are quickly deleted. Rules obtained by both MLP2LN algorithms are ordered, starting with the rules that are used most often and ending with the rules that handle only a few cases. Quality of a set of rules is checked using test data – an optimal balance between the number of rules and the generalization error is usually obtained when only the rules that classify larger number of cases are retained. The final solution may be presented as a set of rules or as a network of nodes performing logical functions, with hidden neurons realizing the rules and the hidden-output neuron weights all set to ± 1.

FSM2LN method. It is well known that RBF networks with Gaussian functions are equivalent to the fuzzy logic systems with Gaussian membership functions [5]. To obtain crisp logic one can either use rectangular basis functions [6] or use transfer functions that may be smoothly changed into functions with cuboidal contour surfaces, for example products of L-units defined in Eq. (1). Such functions are separable, but not radial, therefore we use the Feature Space Mapping (FSM) density estimation constructive network [7] instead of RBF. The slopes of sigmoidal functions are gradually increased during the training process, allowing for smooth transition from fuzzy to crisp rules. Feature selection is performed by adding a penalty term for small dispersions to the error function:

$$E(V) = E_0(V) + \lambda \sum_{i}^{N} 1/(1+\sigma_i^2) \tag{5}$$

where V represents all adaptive parameters, such as positions and dispersions $\sigma_i = |b_i - b_i'|$ of localized units, and the sum runs over all active inputs for the node that is most active upon presentation of a given training vector. The penalty term encourages dispersions to grow – if σ_i includes the whole range of input data the connection to x_i is deleted. After the training each node has class label and represents a cuboid in relevant dimensions of the input subspace, easily interpreted as logical conditions for a given class. An alternative approach is to test how much the nodes may expand without changing the classification error.

III. Results for the hypothyroid and hepar datasets

The hypothyroid dataset was created from real medical tests screening for hypothyroid problems [8]. Since most people were healthy 92.5% of cases belong to the normal group, and 8% of cases belonging to the primary hypothyroid or compensated hypothyroid group. 21 medical tests were made in most cases, with 6 continuous and 15 binary values, and about 10% of values missing. A total of 3772 cases are given for training (results from one year) and 3428 cases for testing (results from the next year). Thus from the classification point of view this is a 3 class problem with 22 attributes.

Many neural classifiers have been tried on this data, giving accuracies up to 98.5% on the test set. Schiffman et.al. [9] optimized about 15 MLPs trained with different variants of backpropagation and cascade correlation algorithms. In addition tedious genetic optimizations have been performed on many MLP network architectures. The best results of this study [9] are included in Table I, together with the results obtained using k-NN, decision trees and statistical classification techniques [1].

We have created an MLP network using C-MLP2LN approach. Since there are 3 classes only two outputs are needed, the third class (normal) is treated as "else" or not the first two. It was relatively easy to find the important features: `FTI, TSH, T3` values, `on thyroxine` and `thyroid surgery`. For the first class one neuron was created, giving 3 rules, but one of these rules classifies only one vector, in addition erroneously. This is quite common situation and therefore rules obtained after the weight analysis should be simplified and checked directly on the data. Initially 4 rules are found, classifying correctly 98.4% of the training cases and 98.0% of the test cases. Linguistic variables were optimized and all conditions checked to find out if they could be dropped without the change of classification error on the test data. Optimization of linguistic variables is easily done by changing the cutoff values until the classification error is minimized. Rule extraction process (network retraining and optimization) is repeated with new linguistic variables, until this iterative process converges. After this iterative optimization the final rules are:

R11: TSH $\geq$ 0.029 $\wedge$ FTI $<$ 0.063
R12: TSH $\in$ [0.0061,0.029) $\wedge$ FTI $<$0.063 $\wedge$ T3 $<$ 0.02
E21: thyroid surgery = yes $\wedge$ TSH $\in$ [0.0061,0.029)
R21: on thyroxine = no $\wedge$ TSH $>$ 0.0061 $\wedge$ FTI $\in$ [0.0644,0.018)

The E21 rule is an exception to the rule R21, which means that it is checked first and if it is true other rules are not checked. With these rules we get 99.68% accuracy on the training set

and 99.07% accuracy on the test set. These results are similar to those found using heuristic version of PVM method by Weiss and Kapouleas [1]. The differences among PVM, CART and MLP2LN are rather small, but other methods, such as well-optimized MLP or cascade correlation classifiers, give significantly worse results.

FSM2LN for this dataset generated 24 rules, but these rules used more features and were not so accurate (98% on the test set) as MLP2LN rules.

TABLE I

Classification results for various classifiers applied to the thyroid dataset.

Method	Training set accuracy %	Test set accuracy %
BP+conjugate gradient	94.6	93.8
Best Backpropagation	99.1	97.6
RPROP	99.6	98.0
Quickprop	99.6	98.3
BP+ genetic optimization	99.4	98.4
Local adaptation rates	99.6	98.5
Cascade correlation	100.0	98.5
k-NN	100.0	95.3
Bayes rule	97.0	96.1
PVM	99.8	99.3
CART (decision tree)	99.8	99.4
FSM2LN rules	99.6	98.0
MLP2LN rules	99.7	99.1

The Hepar dataset was collected by dr. H. Wasyluk and co-workers from the Postgraduate Medical Center in Warsaw, and contains 570 cases described by 119 values of medical tests and other features. These vectors are divided into 16 classes corresponding to different types of liver disease. The data has not yet been analyzed previously. Small number of cases belonging to several classes (only 2 in class 16, or 5 in classes 3 and 9, etc.) does not allow for reliable classification of all data. Moreover, the data is not divided into the training and the test set and since some classes do not have enough samples it is hard to make such partitioning.

Analysis of histograms in one and two dimensions does not yield any hints for good linguistic variables. Therefore we have used FSM2LN approach to find out the relevant features. Selecting a target of 96% accuracy the number of neurons (representing rules) created after training was 170, most of them covering only a few data vectors. Lowering the target accuracy to 90% leads to elimination of only 10 neurons - this shows that complexity of data representation by FSM is appropriate. Rectangular basis functions were used to create FSM clusters and 73 out of 119 features were found relevant.

It is dangerous to select test vectors from Hepar data by random sampling since some of the classes may end up with very few representatives in the training set. To test generalization of the FSM network vectors belonging to five largest classes (more than 50 vectors in each class), comprising 2/3 of all data (a total of 370 vectors), were selected. 50% of these vectors were used for training and the rest for testing. Assuming the target of 94% accuracy on the training part best results of classification on the test set were only 58%, with lower test

results for other target accuracies. Much lower classification accuracies are obtained if the whole data is divided in half or if 20% of all data is randomly selected as a test set.

We have tried to find general trends in the data using MLP2LN, but the number of rules generated in this way was quite large. The most general rule, describing correctly 30 cases (one erroneously), is: IF (F5 ON and F29 OFF and F81 $\in$ [669,3982]) THEN Class 13

IV. Summary

New methods for extraction of logical rules with the help of neural classifiers have been presented and applied to two medical datasets. These methods have many advantages. Crisp and fuzzy rules may be obtained, rules are ordered from the most common to more specialized, hierarchy of rules and exceptions is established, overfitting is avoided by dropping more specialized rules and the accuracy is frequently better than given by other (neural and statistical) approaches. Rules obtained by MLP2LN methods require careful analysis and sometimes additional optimization of linguistic variables. Hierarchy of rules and exceptions gives natural flexibility to the rules derived in our approach, uncommon in other approaches. One concept that is worth more attention is the stability of rules (and other classification systems). Rules are brittle if a small change in the input data leads to a different set of rules or to large errors. The choice of linguistic variables is most important for stability of rules. This is true in the density networks as well as in MLPs. In FSM2LN the penalty Eq. (5) used to eliminate input features is also useful to stabilize the rules. In MLP2LN linguistic variables are optimized using extracted rules and the rule extraction process is iteratively repeated with new, optimized variables.

Results obtained for the hypothyroid dataset are better than those obtained from sophisticated neural and other classifiers and similar to the PVM rules [1] (computationally quite demanding and feasible only if there are a few dominating features) and CART decision tree, which is not so easy to interpret. Results for the Hepar dataset are inconclusive. We would be surprised if a compact representation of this data exists. Probably the tests performed do not contain enough information to allow for reliable classification in this case.

Acknowledgments: Support by the Polish Committee for Scientific Research, grant number 8T11F00308, is gratefully acknowledged.

References

[1] S.M. Weiss, I. Kapouleas, An empirical comparison of pattern recognition, neural nets and machine learning classification methods, in: J.W. Shavlik and T.G. Dietterich, *Readings in Machine Learning*, Morgan Kauffman Publ, CA 1990

[2] R. Andrews, J. Diederich, A.B. Tickle, A Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks, Knowledge-Based Systems 8 (1995) 373–389

[3] W. Duch, R. Adamczak, K. Grąbczewski, Constrained backpropagation for feature selection and extraction of logical rules, *Proc. of Colloquia in AI*, Łódź, Poland, pp. 163-170, 1996

[4] D.J. MacKay, A practical Bayesian framework for backpropagation networks, *Neural Comp.* 4 (1992) 448-472

[5] J-S. R. Jang, C.T. Sun, Functional Equivalence Between Radial Basis Function Neural Networks and Fuzzy Inference Systems, *IEEE Trans. on Neural Networks*, vol. 4, pp. 156-158, 1993.

[6] M. Berthold, K. Huber, Building Precise Classifiers with Automatic rule Extraction, in: Proc. of the IEEE ICNN, Perth, Vol. 3, pp. 1263-1268, 1995.

[7] W. Duch, G.H.F. Diercksen, Feature Space Mapping as a universal adaptive system, *Computer Physics Communications*, 87 (1995) 341–371

[8] C.J. Mertz, P.M. Murphy, UCI repository of machine learning databases, http://www.ics.uci.edu/pub/machine-learning-databases.

[9] W. Schiffmann, M. Joost, R. Werner, Comparison of optimized backpropagation algorithms, *ESANN '93*, Brussels 1993, pp. 97-104; Synthesis and Performance Analysis of Multilayer Neural Network Architectures, Tech. Rep. 15/1992, available in `neuroprose` as schiff.gann.ps.Z