Neurocomputing 207 (2016) 610–622

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Neural network training and rule extraction with augmented discretized input

Yoichi Hayashi [a], Rudy Setiono [b], Arnulfo Azcarraga [c]

[a] Department of Computer Science, Meiji University Tama-ku, Kawasaki 214-8571, Japan
[b] School of Computing, National University of Singapore 13 Computing Drive, Singapore 117417, Republic of Singapore
[c] College of Computer Studies, De La Salle University 2401 Taft Avenue, Manila, Philippines

ARTICLE INFO

Article history:
Received 4 December 2015
Received in revised form
1 March 2016
Accepted 12 May 2016
Communicated by S. Mitra
Available online 2 June 2016

Keywords:
Supervised classification
Network pruning
Discretization
Rule extraction

ABSTRACT

The classification and prediction accuracy of neural networks can be improved when they are trained with discretized continuous attributes as additional inputs. Such input augmentation makes it easier for the network weights to form more accurate decision boundaries when the data samples of different classes in the data set are contained in distinct hyper-rectangular subregions in the original input space. In this paper, we present first how a neural network can be trained with augmented discretized inputs. The additional inputs are obtained by dividing the original interval of each continuous attribute into subintervals of equal length. The network is then pruned to remove most of the discretized inputs as well as the original continuous attributes as long as the network still achieves a minimum preset accuracy requirement. We then discuss how comprehensible classification rules can be extracted from the pruned network by analyzing the activations of the network hidden units and the weights of the network connections that remain in the pruned network. Our experiments on artificial data sets show that the rules extracted from the neural networks can perfectly replicate the class membership rules used to create the data perfectly. On real-life benchmark data sets, neural networks trained with augmented discretized inputs are shown to achieve better accuracy than neural networks trained with the original data.

1. Introduction

Pattern classification is one of the most important tasks in data analysis. Among the many machine learning tools that have been proven to be effective for solving pattern classification problems are artificial neural networks. Successful applications of neural networks in a wide array of domains such as engineering, medical, business and social sciences have been widely reported in the literature [1,22,29,34]. Successful applications of neural networks in business include forecasting cash demand in ATMs [58], predicting financial distress among listed Chinese companies [19], development of an early warning system to predict currency crisis [51] and development of consumer credit scoring models [59]. Neural networks are also found to be very useful tools for time-series forecasting. They are shown to be more accurate than conventional statistical methods when used to predict new technology product demand [5] and foreign exchange rates [35,43].

Trained neural networks are often pruned to improve their generalization capability [40]. Networks with fewer hidden units and connections normally have lower prediction errors than more complex networks [60,61]. By removing redundant and irrelevant network connections, the networks will be less likely to overfit the training data samples. Another benefit of network pruning is the yielding of a less complex network structure which makes the task of extracting comprehensible rules from the network easier. By generating classification rules that are comprehensible to the users, the knowledge embedded in the pruned network can be examined, interpreted and verified by people who are not necessarily machine learning experts. With this goal in mind, algorithms that extract rules from neural networks have been proposed [6,8,15,33,44,57].

Many researchers have reported new development and applications of algorithms that extract rules from trained neural networks recently. Novel applications of neural network rule extraction include solving the problem of identifying factors responsible for air pollution levels [13], determining the quality of cotton yarn [2], detecting fault in a transformer [9], recognizing

E-mail addresses: hayashiy@cs.meiji.jp (Y. Hayashi), rudys@comp.nus.edu.sg (R. Setiono), arnie.azcarraga@delasalle.ph (A. Azcarraga).

http://dx.doi.org/10.1016/j.neucom.2016.05.040
0925-2312/

various hand gestures [28], and predicting derivative use for financial risk hedging [14].

Before rules are extracted, it is crucial to have a network that has been trained to classify the data samples with acceptable accuracy rates. A more accurate network can be expected to yield more accurate classification rules. In order to improve the accuracy of the network and the extracted rules as well as to facilitate the rule extraction process, rule extraction algorithms normally require pruning of redundant network units or connections [6,23,27,48,49,62]. Redundant and irrelevant units and connections are also removed so that the extracted rules are more concise and, hence, easier to analyze and understand.

Hybrid learning methods that combine neural networks with other machine learning tools have also been proposed to obtain more accurate and concise rules. Machine learning techniques that aid in rule extraction from neural networks include artificial immune systems [25], ant colony optimization [36], fuzzy Adaptive Resonance Theory (ART) map [54], genetic algorithm [26,32] and decision trees [16,42]. These hybrid neural network rule extraction methods are applied to solve prediction problems in various domains such as power output generation from wind turbines [26], changes in inflation rate [3] and breast cancer diagnosis [32]. The results obtained from these applications demonstrate the effectiveness of the hybrid approach to neural network rule extraction.

Prediction using an ensemble of models may be more accurate than prediction from just a single model [17,65]. Accordingly, some algorithms have been designed to extract classification rules from a neural network ensemble. The algorithm REFNE [64] utilizes trained neural network ensembles to generate a number of data instances and then extract rules from these instances. On six test data sets, it is shown that the rules extracted by REFNE have lower generalization errors than the neural network ensembles, individual neural networks and C4.5rules. The rules are also shown to be, on the average, fewer in number and in rule antecedents than C4.5rules, a very popular machine learning tool based on decision trees [39]. Rule extraction from neural network ensembles has been applied to predict gene expression in microarray data [4], rainfall runoff [37], protein sequences [11] and medical risks [63].

Discretization of input attributes to make the rule extraction process from neural networks simpler is proposed through a special Interpretable Multi-Layer Perceptron (IMLP) [10]. IMLP has 2 layers of hidden units. Each unit in the first hidden layer is connected to only one input unit and the virtual bias unit. Units in the second hidden layer are fully connected to the first hidden layer units and to the output units. The units in the first hidden layer compute their output according to the threshold function, i.e. the output is 1 if the weighted sum of the input and the bias is positive, and it is 0, otherwise. As only one input is connected to a hidden unit in this layer, a rule condition of the form $a_i < \tau$ will be generated, where a_i is a data attribute and τ is a threshold value obtained by neural network training.

In this paper we propose the method NN-ADV which trains neural networks (NN) with augmented discretized variables (ADV) to improve the network predictive accuracy. The additional discretized inputs are obtained by simply dividing the original interval of each continuous attribute into subintervals of equal length. The thermometer encoding scheme is used to represent the discretized inputs. The network is pruned to remove most of the discretized inputs as well as the original continuous attributes as long as the network still achieves a minimum preset accuracy requirement. Not only the method obtains higher accuracy rates, but it also yields simpler neural network structure with smaller number of connections after pruning. The method then proceeds to extract classification rules from the pruned networks. The rule extraction algorithms that we have developed can generate propositional rules [49], MofN rules [45] or rule sets with disjoint rule conditions for discrete and continuous attributes [48].

The paper is organized as follows. In Section 2 we describe how discrete variables are generated and added to the input data for neural network training and pruning. The new discrete inputs are encoded using the thermometer encoding scheme in order to facilitate rule extraction. We describe how the irrelevant input units, hidden units, and connections of the network are removed by a pruning algorithm. We then present the steps of the rule extraction algorithm that transforms the classification made by the pruned network into a set of comprehensible rules. In Section 3, we present the results of our experiments using both artificial and real-life data sets. From the artificial data sets, the rule extraction algorithms are able to retrieve the exact conditions by which the data samples have been generated. The experiments on six real-life benchmark data sets show that our proposed method yields better accuracy rates than other methods based on support vector machines that have also been trained using discretized input variables. In this section we also present how the proposed approach can be applied to credit card scoring to obtain concise and comprehensible sets of rules that have lower predictive error rates than those generated from other neural network models. Section 4 concludes the paper.

2. Network training and rule extraction with discretized input

A neural network with one hidden layer consisting of a sufficient number of hidden units can be trained to form an arbitrarily complex decision boundary [24]. While the capability to form a very complex boundary gives neural networks competitive advantage over other classification methods such as decision trees and logistic regression, there are situations when simpler decision boundaries are desirable. When classification rules are to be extracted from a trained neural network, rule conditions that are expressed as linear functions of the input data may be preferred over other rule conditions that are more complex. There are also data sets where data samples from different classes can be separated by axis-parallel hyperplanes.

We add discretized values of the original continuous input variables before training our neural networks in order to see if simpler and more accurate decision boundaries can be obtained by the networks. A pruning algorithm is then applied to remove redundant units and connections in the network. Hidden units are removed to reduce data overfitting and improve generalization.

Input units not useful in determining the class boundaries are also removed by pruning. When the decision boundaries are axis parallel, we expect the additional discretized inputs not only to improve the classification accuracy but also to simplify the extracted classification rules. On the other hand, when the decision boundaries are oblique, the discretized inputs should be removed during pruning and the rule conditions would involve weighted sums of the original continuous attributes.

2.1. Input variable discretization

Discretization transforms continuous input variables into discrete ones. Having only discrete variables in the data allows one to apply a wider range of data mining algorithms that can learn only from categorical data [18]. Replacing each original continuous data variable by a set of binary variables has also been suggested as an effective means to improve classification accuracy and to better detect relevant variables and their interactions in the context of support vector machines for supervised classification [12].