Pattern Recognition Letters 68 (2015) 41–47

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

A neural tree for classification using convex objective function☆

Asha Rani[a,*], Gian Luca Foresti[b], Christian Micheloni[b]

[a] CVGIP Lab, Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee 247667, India
[b] AVIRES Lab, Department of Mathematics and Computer Science, University of Udine, Udine 33100, Italy

ARTICLE INFO

Article history:
Received 3 January 2015
Available online 6 September 2015

Keywords:
Neural tree
Artificial neural networks (ANNs)
Mean squared error
Pattern classification
Perceptron
Convex optimization

ABSTRACT

In this paper, we propose a neural tree classifier, called the convex objective function neural tree (COF-NT), which has a specialized perceptron at each node. The specialized perceptron is a single layer feed-forward perceptron which calculates the errors before the neuron's non-linear activation function instead of after them. Thus, the network parameters are independent of non-linear activation functions, and subsequently, the objective function is a convex objective function. The solution can be easily obtained by solving a system of linear equations which will require less computational power than conventional iterative methods. During the training, the proposed neural tree classifier divides the training set into smaller subsets by adding new levels to the tree. Each child perceptron takes forward the task of training done by its parent perceptron on the superset of this subset. Thus, the training is done by a number of single layer perceptrons (each perceptron carrying forward the work done by its ancestors) that reach the global minima in a finite number of steps. The proposed algorithm has been tested on available benchmark datasets and the results are promising in terms of classification accuracy and training time.

1. Introduction

Artificial neural networks (ANNs) have been used in a number of scientific and engineering applications. ANNs [1] have proven to be very powerful for classification and regression tasks. Still, there is a major issue associated with their use-they are very much dependent on architecture. The ANN architecture is not unique for a given problem as there may exist different ways of defining an architecture for a specific problem. Depending on the problem, it may require one or more hidden layers, feed-forward or feedback connections, and there may be direct connections between input and output nodes. Several solutions have been proposed in the literature to tackle this issue. Neural trees are one such solution [2–5].

Neural trees have been used in a broad area of problems. Some of these problems include vowel recognition [6], character recognition [7], face recognition [8], image analysis [9], time series prediction [10], disease classification [11], outlier detection in stereo image matching [12], digital image watermarking [13,14], novelty detection [15], traffic prediction [16], protein structure prediction [17], power signal pattern classification [18], and water stage forecasts in river basin during typhoons [19].

Neural trees are hybrid structures between decision trees and artificial neural networks that were developed to determine the structure of artificial neural networks automatically [4,20–22]. Sethi [23] described a method for converting a univariate decision tree into a neural network and then retraining it, resulting in a tree structured entropy network with sigmoid splits. Guo and Gelfand [24] developed a decision tree with multi-layer perceptrons at each node, giving non-linear and multivariate splits. In the last two decades efforts have been done to optimize the structure of neural trees by punning techniques [25,26]. In [25], the tree is pre-pruned by removing some patterns that lead to over-fitting and add quite a few levels to the tree. In [26], a uniformity factor has been introduced to pre-prune the tree branches. Such a factor stops the tree growing further by accepting some error, bounded by a threshold. Different variants of stochastic search methods have been used for the development of structure and parameter identification [8,10,16]. Based on the kind of optimization strategy different kinds of neural trees have been distinguished in the literature such as the balanced neural tree (BNT) [25], flexible neural tree (FNT) [27], and generalized neural tree (GNT) [28]. Different variants of artificial neural networks, such as high-order perceptrons (HOP) [29], multi-layer perceptrons (MLP) [26], and radial-basis functions (RBF) [30,31] have been used as decision taking units at nodes of the neural tree. Apart from the several kinds of hybrid neural trees, exploiting more than one type of classification unit has also been undertaken [8,32–35].

Although there are several kinds of neural tree classifiers proposed in the literature employing various kinds of learning machines such as MLP, HOP, RBF, etc., the use of a single layer perceptron (SLP) as the decision taking unit has its own advantages. SLPs are simple

☆ This paper has been recommended for acceptance by G. Moser.

* Corresponding author. Tel.: +91 1332 285824.
E-mail address: asha0chaudhary@gmail.com (A. Rani).

http://dx.doi.org/10.1016/j.patrec.2015.08.017
0167-8655/

to implement and computationally cheaper compared to above the mentioned learning machines. They operate with a minimum number of ad-hoc parameters. In the proposed COF-NT, the necessity of ad-hoc parameters such as learning rate, number of iterations, error tolerance threshold, etc. is completely eliminated. Neural trees based on single layer neural networks or multi-layer neural networks use conventional iterative learning schemes to optimize the mean squared error (MSE) objective function. The conventional MSE is a non-convex objective function due to superimposition of several convex functions. As a result, the learning process has a tendency to stick in local minima and does not obtain optimal solution. The consequence is an enlarged tree structure with a poor generalization capability or a non-converging tree building process. Other than this, due to iterative learning schemes, these kinds of neural trees need longer training times. In this paper, we propose a new neural tree classifier, called the Convex Objective Function Neural Tree (COF-NT), that uses a specialized single layer perceptron using a convex objective function. In this objective function, the mean squared error is computed before the non-linear activation function. As the new objective function is a convex function, the optimal solution can be obtained analytically by equating derivatives to zero, giving a system of linear equations. Thus, the weights of the COF-NT network are computed by solving a system of linear equations, which is much faster than iterative schemes.

2. Description of the COF-NT classifier

A single layer perceptron is easy to train compared to a multilayer perceptron. However, although a single layer perceptron uses a non-linear activation function in its output layer, it still has only a linear discrimination capability [36]. A non-linear performance can be obtained by using single layer perceptrons in a non-linear structure, such as in a tree structure. The proposed COF-NT classifier has specialized single layer perceptrons with convex objective functions at each node. We will discuss the perceptron with convex objective functions in later sections. A neural tree learns the training set (TS) by partitioning it into smaller subsets called local training sets (LTS). It has a unique root node, several internal nodes, and several leaf nodes.

2.1. Training phase

The training process starts at the root node with whole training set as input. A single layer feed-forward neural network with convex objective function (perceptron) is trained at the root node. A trained perceptron splits the training set into subsets depending on the generated activation values for each of the outputs. The winner-takes-all rule is applied to classify a pattern, i.e. a pattern belongs to the class the having highest activation value. Thus, all the patterns present in the current LTS are further divided into groups/LTS. Child nodes are added to the tree at the next level corresponding to each LTS. A single layer neural network is trained at the child node for learning the corresponding LTS. The tree keeps on growing by adding child nodes and the training process proceeds until all the LTS become homogeneous. A homogeneous LTS is a set consisting of patterns that belong to a single class. Furthermore, classes are not mixed. When an LTS becomes homogeneous, the node corresponding to this LTS is marked as a leaf node and labeled by the class to which the pattern of this LTS belongs. When all the nodes at the current level (the latest level) become leaf nodes, the tree stops growing and the training process are complete.

Some times the perceptron is unable to divide the LTS in further groups (see discussion in later sections). In such a case, the perceptron is replaced by a binary classifier that divides the LTS based on features having maximum variance among two dominating classes present in the LTS. Such a classifier splits the LTS into two LTSs. Now the tree model is ready for classifying unseen patterns of a similar type. A flowchart describing the training phase of COF-NT is shown in Fig. 1.

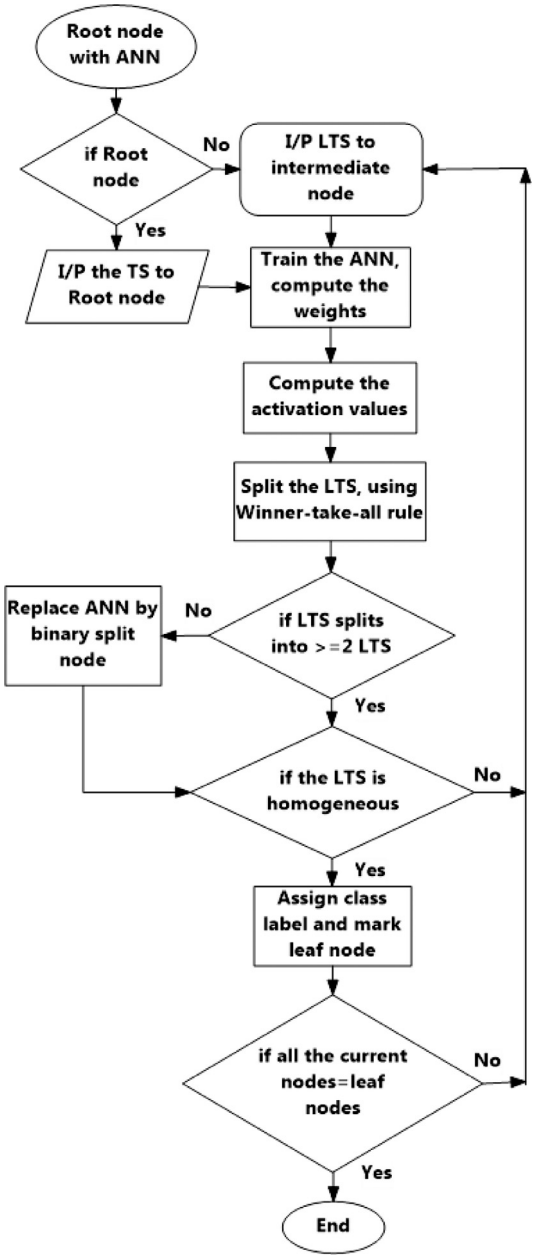

Fig. 1. Flow chart diagram of COF-NT (training phase).

2.2. Classification phase

The already build tree model is traversed in a top-down approach to classify a test pattern. A test pattern starts traversing the tree at the root node and moves down until it reaches a leaf node. The label of the leaf node tells the class of the test pattern. There are several paths in a tree, out of these paths which path a pattern will take, has to be decided by the weights of the single layer perceptron stored at each node during the training process. The perceptron generates activation values for a pattern for each of the possible classes using the weights. The highest activation value tells the class, and in turn, the path, to be taken by the test pattern. In cases, where a test pattern reaches a binary split node, a prefixed threshold decides the path to be taken by the test pattern. A flowchart describing the classification phase of COF-NT is shown in Fig. 2.