Applied Soft Computing xxx (2016) xxx–xxx

Contents lists available at ScienceDirect

Applied Soft Computing

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

Ensemble of heterogeneous flexible neural trees using multiobjective genetic programming

Varun Kumar Ojha[a,*], Ajith Abraham[b], Václav Snášel[a]

[a] IT4Innovations, VŠB-Technical University of Ostrava, Ostrava, Czech Republic
[b] Machine Intelligence Research Labs (MIR Labs), Auburn, WA, USA

ARTICLE INFO

Article history:
Received 14 July 2015
Received in revised form 9 August 2016
Accepted 24 September 2016
Available online xxx

Keywords:
Pareto-based multiobjectives
Flexible neural tree
Ensemble
Approximation
Feature selection

ABSTRACT

Machine learning algorithms are inherently multiobjective in nature, where approximation error minimization and model's complexity simplification are two conflicting objectives. We proposed a multiobjective genetic programming (MOGP) for creating a heterogeneous flexible neural tree (HFNT), tree-like flexible feedforward neural network model. The functional heterogeneity in neural tree nodes was introduced to capture a better insight of data during learning because each input in a dataset possess different features. MOGP guided an initial HFNT population towards Pareto-optimal solutions, where the final population was used for making an ensemble system. A *diversity index* measure along with *approximation error* and *complexity* was introduced to maintain diversity among the candidates in the population. Hence, the ensemble was created by using accurate, structurally simple, and diverse candidates from MOGP final population. Differential evolution algorithm was applied to fine-tune the underlying parameters of the selected candidates. A comprehensive test over classification, regression, and time-series datasets proved the efficiency of the proposed algorithm over other available prediction methods. Moreover, the heterogeneous creation of HFNT proved to be efficient in making ensemble system from the final population.

1. Introduction

Structure optimization of a feedforward neural network (FNN) and its impact on FNN's generalization ability inspired the flexible neural tree (FNT) [1]. FNN components such as weights, structure, and activation function are the potential candidates for the optimization, which improves FNN's generalization ability to a great extent [2]. These efforts are notable because of FNN's ability to solve a large range of real-world problems [3–6]. Followings are the important structure optimization methods: constructive and pruning algorithms [7,8], EPNet [2], NeuroEvolution of Augmenting Topologies [9], sparse neural trees [10], Cooperative co-evolution approach [11], etc.

Similarly, many efforts focus on the optimization of hybrid training of FNN [12–14]. FNT was an additional step into this series of efforts, which was proposed to evolve as a tree-like feed-forward neural network model, where the probabilistic incremental program evolution (PIPE) [15] was applied to optimize the tree structure [1]. The underlying parameter vector of the developed FNT (weights associated with the edges and arguments of the activation functions) was optimized by metaheuristic algorithms, which are nature-inspired parameter optimization algorithms [16]. The evolutionary process allowed FNT to select significant input features from an input feature set.

In the design of FNT, the non-leaf nodes are the computational nodes, which takes an activation function. Hence, rather than relying on a fixed activation function, if the selection of activation function at the computational nodes is allowed to be selected by the evolutionary process. Then, it produces heterogeneous FNTs (HFNT) with the heterogeneity in its structure, computational nodes, and input set. In addition, heterogeneous function allowed HFNT to capture different feature of the datasets efficiently since each input in the datasets posses different features. The evolutionary process provides adaptation in structure, weights, activation functions, and input features. Therefore, an optimum HFNT is the one that offers the lowest approximation error with the simplest tree structure and the smallest input feature set. However, approximation error minimization and structure simplification are two conflicting objectives [17]. Hence, a multiobjective evolutionary approach [18] may offer an optimal solution(s) by maintaining a balance between these objectives.

* Corresponding author.
E-mail addresses: varun.kumar.ojha@vsb.cz (V.K. Ojha), ajith.abraham@ieee.org (A. Abraham), vaclav.snasel@vsb.cz (V. Snášel).

http://dx.doi.org/10.1016/j.asoc.2016.09.035
1568-4946/© 2016 Elsevier B.V. All rights reserved.

Moreover, in the proposed work, an evolutionary process guides a population of HFNTs towards Pareto-optimum solutions. Hence, the final population may contain several solutions that are close to the best solution. Therefore, an ensemble system was constructed by exploiting many candidates of the population (candidate, solution, and model are synonymous in this article). Such ensemble system takes advantage of many solutions including the best solution [19]. Diversity among the chosen candidates holds the key in making a good ensemble system [20]. Therefore, the solutions in a final population should fulfill the following objectives: low approximation error, structural simplicity, and high diversity. However, these objectives are conflicting to each other. A fast elitist nondominated sorting genetic algorithm (NSGA-II)-based multiobjective genetic programming (MOGP) was employed to guide a population of HFNTs [21]. The underlying parameters of selected models were further optimized by using differential evaluation (DE) algorithm [22]. Therefore, we may summarize the key contributions of this work are as follows:

(1) A heterogeneous flexible neural tree (HFNT) for function approximation and feature selection was proposed.
(2) HFNT was studied under an NSGA-II-based multiobjective genetic programming framework. Thus, it was termed HFNT^M.
(3) Alongside *approximation error* and *tree size* (complexity), a *diversity index* was introduced to maintain diversity among the candidates in the population.
(4) HFNT^M was found competitive with other algorithms when compared and cross-validated over classification, regression, and time-series datasets.
(5) The proposed evolutionary weighted ensemble of HFNTs final population further improved its performance.

A detailed literature review provides an overview of FNT usage over the past few years (Section 2). Conclusions derived from literature survey supports our HFNT^M approach, where a Pareto-based multiobjective genetic programming was used for HFNT optimization (Section 3.1). Section 3.2 provides a detailed discussion on the basics of HFNT: MOGP for HFNT structure optimization, and DE for HFNT parameter optimization. The efficiency of the above-mentioned hybrid and complex multiobjective FNT algorithm (HFNT^M) was tested over various prediction problems using a comprehensive experimental set-up (Section 4). The experimental results support the merits of proposed approach (Section 5). Finally, we provide a discussion of experimental outcomes in Section 6 followed by conclusions in Section 7.

2. Literature review

The literature survey describes the following points: basics of FNT, approaches that improvised FNT, and FNTs successful application to various real-life problems. Subsequently, the shortcomings of basic FNT version are concluded that inspired us to propose HFNT^M.

FNT was first proposed by Chen et al. [1], where a tree-like-structure was optimized by using PIPE. Then, its approximation ability was tested for time-series forecasting [1] and intrusion detection [23], where a variant of simulated annealing (called degraded ceiling) [24], and particle swarm optimization (PSO) [25], respectively, were used for FNT parameter optimization. Since FNT is capable of input feature selection, in [26], FNT was applied for selecting input features in several classification tasks, in which FNT structure was optimized by using genetic programming (GP) [27], and the parameter optimization was accomplished by using memetic algorithm [28]. Additionally, they defined five different mutation operators, namely, changing one terminal node, all terminal nodes, growing a randomly selected sub-tree, pruning a randomly selected sub-tree, and pruning redundant terminals.

Li et al. [29] proposed FNT-based construction of decision trees whose nodes were conditionally replaced by neural node (activation node) to deal with continuous attributes when solving classification tasks. In many other FNT based approaches, like in [30], GP was applied to evolve hierarchical radial-basis-function network model, and in [31] a multi-input-multi-output FNT model was evolved. Wu et al. [32] proposed to use grammar guided GP [33] for FNT structure optimization. Similarly, in [34], authors proposed to apply multi-expression programming (MEP) [35] for FNT structure optimization and immune programming algorithm [36] for the parameter vector optimization. To improve classification accuracy of FNT, Yang et al. [37] proposed a hybridization of FNT with a further-division-of-partition-space method. In [38], authors illustrated crossover and mutation operators for evolving FNT using GP and optimized the tree parameters using PSO algorithm.

A model is considered efficient if it has generalization ability. We know that a consensus decision is better than an individual decision. Hence, an ensemble of FNTs may lead to a better-generalized performance than a single FNT. To address this, in [39], authors proposed to make an ensemble of FNTs to predict the chaotic behavior of stock market indices. Similarly, in [40], the proposed FNTs ensemble predicted the breast cancer and network traffic better than individual FNT. In [41], protein dissolution prediction was easier using ensemble than the individual FNT.

To improve the efficiency in terms of computation, Peng et al. [42] proposed a parallel evolving algorithm for FNT, where the parallelization took place in both tree-structure and parameter vector populations. In another parallel approach, Wang et al. [43] used gene expression programming (GEP) [44] for evolving FNT and used PSO for parameter optimization.

A multi-agent system [45] based FNT (MAS-FNT) algorithm was proposed in [46], which used GEP and PSO for the structure and parameter optimization, respectively. The MAS-FNT algorithm relied on the division of the main population into sub-population, where each sub-population offered local solutions and the best local solution was picked-up by analyzing tree complexity and accuracy.

Chen et al. [1,26] referred the arbitrary choice of activation function at non-leaf nodes. However, they were restricted to use only Gaussian functions. A performance analysis of various activation function is available in [47]. Bouaziz et al. [48,49] proposed to use beta-basis function at non-leaf nodes of an FNT. Since beta-basis function has several controlling parameters such as shape, size, and center, they claimed that the beta-basis function has advantages over other two parametric activation functions. Similarly, many other forms of neural tree formation such as balanced neural tree [50], generalized neural tree [51], and convex objective function neural tree [52], were focused on the tree improvement of neural nodes.

FNT was chosen over the conventional neural network based models for various real-world applications related to prediction modeling, pattern recognition, feature selection, etc. Some examples of such applications are cement-decomposing-furnace production-process modeling [53], time-series prediction from gene expression profiling [54]. stock-index modeling [39], anomaly detection in peer-to-peer traffic [55], intrusion detection [56], face identification [57], gesture recognition [58], shareholder's management risk prediction [59], cancer classification [60], somatic mutation, risk prediction in grid computing [61], etc.

The following conclusions can be drawn from the literature survey. First, FNT was successfully used in various real-world applications with better performance than other existing function approximation models. However, it was mostly used in time-series analysis. Second, the lowest approximation error obtained by an