Neurocomputing 174 (2016) 203–210

Contents lists available at ScienceDirect

Neurocomputing

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

Extreme learning machine based transfer learning for data classification

Xiaodong Li [a,*], Weijie Mao [b], Wei Jiang [b]

[a] School of Software Engineering, Hangzhou Dianzi University, 310018 Hangzhou, PR China
[b] State Key Laboratory of Industrial Control Technology Institute of Cyber-Systems and Control Zhejiang University, Yuquan Campus, Hangzhou 310027, PR China

ARTICLE INFO

Article history:
Received 30 September 2014
Received in revised form
5 January 2015
Accepted 6 January 2015
Available online 11 August 2015

Keywords:
Extreme learning machine
Transfer learning (TL)
Classification

ABSTRACT

The extreme learning machine (ELM) is a new method for using Single-hidden Layer Feed-forward Networks (SLFNs) with a much simpler training method. While conventional extreme learning machine are based on the training and test data which should be under the same distribution, in reality it is often desirable to learn an accurate model using only a tiny amount of new data and a large amount of old data. Transfer learning (TL) aims to solve related but different target domain problems by using plenty of labeled source domain data. When the task from one new domain comes, new domain samples are relabeled costly, and it would be a waste to discard all the old domain data. Therefore, an algorithm called TL-ELM based on the ELM algorithm is proposed, which uses a small amount of target domain tag data and a large number of source domain old data to build a high-quality classification model. The method inherits the advantages of ELM and makes up for the defects that traditional ELM cannot transfer knowledge. Experimental results indicate that the performance of the proposed methods is superior to or at least comparable with existing benchmarking methods. In addition, a novel domain adaptation kernel extreme learning machine (TL-DAKELM) based on the kernel extreme learning machine was proposed with respect to the TL-ELM. Experimental results show the effectiveness of the proposed algorithm.

1. Introduction

As a nonlinear model, neural network (NN) has the ability of good generalization and nonlinear mapping, which can be used to solve the dimension curse problem [1]. Combining forward propagation of information with back- propagation of error, typical back-propagation (EBP) neural network plays an important role in neural learning [2]. Besides, support vector machine (SVM) is based on the statistical learning and structural risk minimization principle [3,4]. However, it is known that both BP neural network and SVM have some challenging issues such as: (1) slow learning speed, (2) trivial human intervention, (3) poor computational scalability [37]. A new learning algorithm, i.e., extreme learning machine (ELM) was proposed by Huang et al. [5]. Compared with BP neural network and SVM, the ELM has better generalization performance at a much faster learning speed and with least human intervention.

Although ELM has made some achievements, but there is still space for improvement. Some scholars are engaged in optimizing the learning algorithm of ELM. Han et al. [6] encoded a priori information to improve the function approximation of ELM. Kim et al. [7] introduced a variable projection method to reduce the dimension of the parameter space. Zhu et al. [8] used a differential evolutionary algorithm to select the input weights of ELM. Some other scholars dedicated to optimize the structure of ELM. Wang et al. [9] made a proper selection of the input weights and bias of ELM in order to improve the performance of ELM. Li et al. [10] proposed a structure-adjustable online ELM learning method, which can adjust the number of hidden layer RBF nodes. Huang et al. [11,12] proposed an incremental structure ELM, which increase the hidden nodes gradually. Meanwhile, another incremental approach referred to as error minimized extreme learning machine (EM-ELM) was proposed by Feng et al. [13]. All these incremental ELM start from a small size of ELM hidden layer, and add random hidden node (nodes) to the hidden layer. During the growth of networks, the output weights are updated incrementally. On the other hand, an alternative method to optimize the structure of ELM is to train an ELM that is larger than necessary and then prune the unnecessarily nodes during learning. A pruned ELM (PELM) was proposed by Rong et al. [14,15] for classification problem. Yoan et al. [16] proposed an optimally pruned extreme learning machine (OP-ELM) methodology. Besides, there are still

* Corresponding author.
E-mail addresses: hzxiaodong22@163.com (X. Li), wjmao@iipc.zju.edu.cn (W. Mao), jiangwei@iipc.zju.edu.cn (W. Jiang).

http://dx.doi.org/10.1016/j.neucom.2015.01.096
0925-2312/

other attempts to optimize the structure of ELM, such as CS-ELM [17] proposed by Lan et al., which used a subset model selection method. Zong et al. [18] put forward the weighted extreme learning machine for imbalance learning. The kernel trick applied to ELM was introduced in previous work [19]. While conventional extreme learning machine are based on the training and test data which should be under the same distribution, in reality it is often desirable to learn an accurate model using only a tiny amount of new data and a large amount of old data.

The transfer learning technology can reuse past experience and knowledge to solve current problem. Generally speaking, the goal of transfer learning is to use training data from related tasks to aid learning on a future problem of interest. Transfer learning refers to the problem of retaining and leveraging the knowledge available for one or more tasks, domains, or distributions to efficiently develop a reasonable hypothesis for a new task, domain, or distribution [20].Transfer learning (TL) is a method that aims at reusing knowledge learned in an environment to improve the learning performance in new environments, which can solve the problem of transfer learning from different but similar tasks [21]. According to the relationship between the source and target domains, TL can be divided into inductive transfer [22] and transductive transfer [23]. Dai et al. [24] proposed a boosting algorithm, TrAdaBoost, which is an extension of the AdaBoost algorithm, to address the inductive transfer learning problems. Wu et al. [25] integrated the source domain (auxiliary) data in Support Vector Machine (SVM) framework for improving the classification performance. Pan et al. [29] proposed a Q learning system for continuous spaces which is constructed as a regression problem for an ELM. Instead of involving generalization across problem instances, transfer learning emphasizes the transfer of knowledge across tasks, domains, and distributions that are related but not the same. The default assumption of traditional supervised learning methods is that training and testing data are drawn from the same distribution. When the two distributions do not match, two distinct transfer learning sub-problems can be defined depending on whether the training and testing data refer to the same domain or not [33]. In the framework of domain adaptation, most of the learning methods are inspired by the idea that these two considered domains, although different, are highly correlated [35]. Duan et al. also proposed a domain transfer SVM(DTSVM) and its extended version DTMKL for DAL problems such as cross-domain video concept detection and text classification [36].

In this paper, we would like to investigate these issues. Therefore, an algorithm called transfer learning based on the ELM algorithm (TL-ELM) is proposed, which uses a small number of target tag data and a large number of source domain old data to build a high-quality classification model. The method takes the advantages of the traditional ELM and makes up for the defect that traditional ELM cannot migrate knowledge. In addition, we propose a so-called TL-DAKELM based on the kernel extreme learning machine ELM as an extension to the TL-ELM method for pattern classification problems. Experimental results show the effectiveness of the proposed algorithm.

2. Kernel extreme learning machine

This section we briefly review the ELM proposed in [26]. The essence of ELM is that in ELM the hidden layer need not be tuned. The output function of ELM for generalized SLFNs is

$$f_L(x)=\sum_{i=1}^{L}\beta_i h_i(x_j)=\sum_{i=1}^{L}\beta_i h(w_i\cdot x_j+b_i)=h(x)\beta \quad j=1,\ldots,N \tag{1}$$

where $w_i\in R^n$ is the weight vector connecting the input nodes and the ith hidden node,$b_i\in R$ is the bias of the ith hidden node,$\beta_i\in R$ is the weight connecting the ith hidden node and the output node, and $f_L(x)\in R$ is the output of the SLFN. $w_i\cdot x_j$ denotes the inner product of w_i and $x_j\cdot w_i$ and b_i are the learning parameters of hidden nodes and they are randomly chosen before learning.

If the SLFN with N hidden nodes can approximate theses N samples with zero error, it then means there exists β_i,w_i, and b_i such that

$$\sum_{i=1}^{L}\beta_i h(w_i\cdot x_j+b_i)=t_j,\quad j=1,\ldots,N \tag{2}$$

Eq. (2) can be written compactly as

$$H\boldsymbol{\beta}=\mathbf{T} \tag{3}$$

where

$$\mathbf{H}=\begin{pmatrix}h(x_1)\\ \vdots\\ h(x_N)\end{pmatrix}=\begin{pmatrix}h(w_1,b_1,x_1) & \cdots & h(w_L,b_L,x_1)\\ \vdots & \ddots & \vdots\\ h(w_1,b_N,x_1) & \cdots & h(w_L,b_L,x_N)\end{pmatrix}_{N\times L},$$

$\mathbf{T}=[t_1,\ldots,t_N]^T$, and $\boldsymbol{\beta}=[\beta_1,\beta_2,\cdots,\beta_L]^T$.

$\mathbf{H}$ is called the hidden layer output matrix of the network [4,5]; the ith column of $\mathbf{H}$ is the ith hidden node's output vector with respect to inputs $x_1,x_2,\cdots,x_N$ and the jth row of $\mathbf{H}$ is the output vector of the hidden layer with respect to input x_j.As introduced in [27], one of the methods to calculate Moore–Penrose generalized inverse of a matrix is the orthogonal projection method: $\mathbf{H}^{\dagger}=\mathbf{H}^T(\mathbf{H}\mathbf{H}^T)^{-1}$.

According to the ridge regression theory [27], one can add a positive value to the diagonal of $\mathbf{H}\mathbf{H}^T$ the resultant solution is more stable and tends to have better generalization performance

$$f(x)=h\beta=h(x)H^T\left(\frac{\mathbf{I}}{C}+\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T}, \tag{4}$$

The feature mapping $h(x)$ is usually known to users in ELM. However, if a feature mapping $h(x)$ is unknown to users a kernel matrix for ELM can be defined as follows [6]:

$$\Omega_{ELM}=\mathbf{H}\mathbf{H}^T:\Omega_{ELM_{i,j}}=h(x_i)\cdot h(x_j)=K(x_i,x_j). \tag{5}$$

Thus, the output function of ELM classifier can be written compactly as

$$f(x)=h(x)\mathbf{H}^T\left(\frac{\mathbf{I}}{C}+\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T}=\begin{bmatrix}K(x,x_1)\\ \vdots\\ K(x,x_N)\end{bmatrix}^T\left(\frac{\mathbf{I}}{C}+\Omega_{ELM}\right)^{-1}\mathbf{T}. \tag{6}$$

Algorithm1. Given a training set $\{(x_i,t_i)\}_{i=1}^{N}\subset R^n\times R^n$, activation kernel function $g(\cdot)$, and the hidden node number L:

Step 1: Randomly assign input weight w_i and bias $b_i, i=1,\cdots,L$.
Step 2: Calculate the hidden layer output matrix $\mathbf{H}$.
Step 3: Calculate the output weight $\boldsymbol{\beta}$: $\boldsymbol{\beta}=\mathbf{H}^{\dagger}\mathbf{T}$.

3. ELM in transfer learning

3.1. Minimum norm least-squares (LS) solution of SLFNs

It is very interesting and surprising that unlike the most common understanding that all the parameters of SLFNs need to be adjusted, the input weights w_i and the hidden layer biases b_i are in fact not necessarily tuned and the hidden layer output matrix $\mathbf{H}$ can actually remain unchanged once random values have been assigned to these parameters in the beginning of learning. For fixed input weights w_i and the hidden layer biases b_i, seen from Eq. (7), to train an SLFN is simply equivalent to finding a