Neural Networks 60 (2014) 133–140

Contents lists available at ScienceDirect

Neural Networks

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

Logarithmic learning for generalized classifier neural network

Buse Melis Ozyildirim [a,*], Mutlu Avci [b]

[a] Department of Computer Engineering, Adana Science and Technology University, Adana, Turkey
[b] Department of Biomedical Engineering, University of Cukurova, Adana, Turkey

ARTICLE INFO

Article history:
Received 5 February 2014
Received in revised form 11 July 2014
Accepted 11 August 2014
Available online 19 August 2014

Keywords:
GCNN
Logarithmic cost function
Classification neural networks
Gradient descent learning

ABSTRACT

Generalized classifier neural network is introduced as an efficient classifier among the others. Unless the initial smoothing parameter value is close to the optimal one, generalized classifier neural network suffers from convergence problem and requires quite a long time to converge. In this work, to overcome this problem, a logarithmic learning approach is proposed. The proposed method uses logarithmic cost function instead of squared error. Minimization of this cost function reduces the number of iterations used for reaching the minima. The proposed method is tested on 15 different data sets and performance of logarithmic learning generalized classifier neural network is compared with that of standard one. Thanks to operation range of radial basis function included by generalized classifier neural network, proposed logarithmic approach and its derivative has continuous values. This makes it possible to adopt the advantage of logarithmic fast convergence by the proposed learning method. Due to fast convergence ability of logarithmic cost function, training time is maximally decreased to 99.2%. In addition to decrease in training time, classification performance may also be improved till 60%. According to the test results, while the proposed method provides a solution for time requirement problem of generalized classifier neural network, it may also improve the classification accuracy. The proposed method can be considered as an efficient way for reducing the time requirement problem of generalized classifier neural network.

1. Introduction

Supervised learning neural networks generally learn by updating weight values denoting synapses of biological neural system. Therefore, weights are the key parameters of neural networks. There are several learning algorithms defined for obtaining appropriate weight values. Learning algorithms should be fast while they provide optimized weights. Correlation, outstar, perceptron, Widrow–Hoff (Least Mean Square, LMS), Delta learning, and Levenberg–Marquardt approaches are some of the supervised learning methods (Wilamowski, 2009). In perceptron learning, weights are updated with the multiplication of input and the difference between the desired output and the output of network. LMS learning aims to minimize squared error function. Error function is defined as sum of squared difference between the output of network and the desired output, given in (1) where i denotes ith neuron, R is the number of training data, net_{ir} denotes the output of ith neuron for rth training datum, d_{ir} denotes the target value of ith neuron for rth training datum and e_i represents the error value of ith neuron.

$$e_i = \sum_{r=1}^{R} (net_{ir} - d_{ir})^2 . \tag{1}$$

Change of weight is calculated with the derivative of error function (Widrow & Hoff, 1960). There are two kinds of update named as incremental and batch. In incremental update, weights are changed after each training. On the other hand, in batch training after applying all training samples and obtaining average error, weights are updated (Widrow & Hoff, 1960; Wilamowski, 2009). Incremental learning converges to minimum faster than batch approach, however; performance of incremental learning depends on the order of inputs (Widrow & Hoff, 1960; Wilamowski, 2009). Linear regression provides one step learning while Least Square Error (LSE) requires many iterations. It is also used for training neural networks with linear activation function (2), where X, W and d denote inputs, weights and desired outputs respectively.

$$\begin{aligned} XW &= d \\ W &= \left(X^T X\right)^{-1} X^T d. \end{aligned} \tag{2}$$

Delta learning rule is an improved version of LMS. Difference between LMS and delta learning rule is the activation function of

* Corresponding author. Tel.: +90 05552507373.
E-mail addresses: melis.ozyildirim@gmail.com, bmozyildirim@adanabtu.edu.tr (B.M. Ozyildirim), mavci@cu.edu.tr (M. Avci).

http://dx.doi.org/10.1016/j.neunet.2014.08.004
0893-6080/

network. Both batch and incremental learning approaches can be used in delta learning rule as in LMS. Backpropagation learning is based on delta learning rule and named as gradient descent algorithm (McClelland & Rumelhart, 1988).

Another learning rule known as Levenberg–Marquardt (LM) algorithm uses Jacobian matrix with trust region strategy. It updates weights as given in (3), where w_k denotes weights of kth iteration, J_k denotes the Jacobian matrix, e is calculated error, μ is a positive coefficient and I is the identity matrix. For each step it requires two major processes. One of them is Jacobian matrix calculation and the other is the computation of the inverse of the squared Jacobian matrix. Hence, generally LM is not used for training of large neural networks due to calculation complexity (Levenberg, 1944; Marquardt, 1963).

$$w_{k+1} = w_k - \left(J_k^T J_k + \mu I\right)^{-1} J_k^T e. \tag{3}$$

In Radial Basis Function (RBF) based neural networks such as Generalized Regression Neural Network (GRNN), Probabilistic Neural Network (PNN) and Generalized Classifier Neural Network (GCNN), smoothing parameter determines the performance of methods instead of weights. Therefore, determining optimal smoothing parameter is critical for these approaches. There are studies for optimizing smoothing parameter with gradient descent method (Berthold & Diamond, 1998; Lee, Lim, Yuen, & Lo, 2004; Mao, Tan, & Set, 2000; Masters & Land, 1997).

GCNN is an RBF based neural network for classification. It uses regression based convergence, diverge effect term and gradient descent based smoothing parameter optimization to solve problems encountered by PNN and GRNN such as overfitting, optimal smoothing parameter value and stuck neuron problems. Regression based convergence is provided by assigning a value 0.9 or 0.1 to each datum according to its class. GCNN uses diverge effect term in N neurons of summation layer, which is an exponential form of $y(j, i) - y_{\max}$. Diverge effect term increases the effect of $y(j, i)$ and separates data that belong to different classes. Smoothing parameter is the most important parameter of RBF based neural networks. Smoothing parameter optimization is provided by gradient descent learning in GCNN. Efficiency of GCNN is proved by test results. Although smoothing parameter optimization increases the efficiency, it leads to long training time requirement. Long training time requirement can be considered as a drawback of GCNN (Ozyildirim & Avci, 2013). It is observed that the initial value of smoothing parameter the key component of GCNN, affects the classification performance and determines the required time to converge. The farther the smoothing parameter from the optimal value, the longer convergence time is required.

In this work, a learning method named as Logarithmic Learning GCNN (L-GCNN) is proposed for reducing training time and improving the efficiency of standard GCNN. The approach contains the main idea of maximizing logarithmic likelihood of probabilistic assumption based on the logistic regression model. Since GCNN has regression based classification method, unlike squared error approach, logarithmic function provides continuous optimization within a range. While efficiency of GCNN is improved, the number of iterations are decreased by using the proposed learning method. L-GCNN is tested on 15 different data sets in MATLAB environment with different initial smoothing parameter values. These are glass identification, Haberman's survival, two spiral problem, lenses, Balance Scale, iris, breast-cancer-Wisconsin (Bennett & Mangasarian, 1992; Mangasarian, Setiono, & Wolberg, 1990; Mangasarian & Wolberg, 1990; Wolberg & Mangasarian, 1990), E.coli, yeast, wine, ionosphere, hill-valley, pen-digits, image segmentation, and transfusion (Yeh, Yang, & Ting, 2008) data sets (Frank & Asuncion, 2010). Classification performances, training and test times of L-GCNN are compared with that of standard one. Results are summarized in Table 1.

Table 1
Summary of L-GCNN's results.

Initial σ	Decrease in training time (%)		Increase in classification performance (%)	
	Maximum	Average	Maximum	Average
10	99	51.6	60	7.84
0.67	99	16	57	4
0.3	89	30	0	0

Initial values of smoothing parameters are chosen as 10, 0.67 and 0.3. When the initial smoothing parameter is chosen as 10, maximum 99% and average 51.6% decrease in training time and maximum 60% and average 7.84% increase in classification performance are obtained with L-GCNN. Maximum 99% and average 16% decrease in training time and maximum 57% and average 4% increase in classification performance are obtained when the initial smoothing parameter is 0.67 and maximum 89% and average 30% decrease in training time and the same classification performances are obtained when it is 0.3.

According to the test results, this training method can be considered as a solution for convergence time of standard GCNN.

2. Maximum likelihood estimation for logistic regression

Similarly, least-squares estimation (LSE), and maximum likelihood estimation (MLE) are statistical methods for parameter estimation. Likelihood function $L(w|o)$ is the probability density function that fits the target model best. MLE developed by R.A. Fisher in the 1920s was based on maximization of likelihood function by searching the parameter space. Parameter estimation starts with definition of log-likelihood function ($\log L(w|o)$). Since log-likelihood function is differentiable, if there is an appropriate parameter vector, the derivative of likelihood equation given in (4) will be obtained, where i is an indice of parameter numbers, w and o denote weight and output, respectively. The likelihood equation requires two conditions given in (4) and (5) to guarantee maximization of $L(w|o)$ and existence of such a parameter vector. The shape of function is checked for maximization control. Log-likelihood function should be convex near the estimated parameter vector. This can be controlled with the second derivative of function, given in (5). This second derivative should be negative at estimated parameter vector (Myung, 2003).

$$\frac{\partial \log L(w|o)}{\partial w_i} = 0 \tag{4}$$

$$\frac{\partial^2 \log L(w|o)}{\partial w_i^2} < 0. \tag{5}$$

If probability density function is non-linear and there are many parameters, optimization algorithms will be used for maximizing log-likelihood to obtain optimal parameters. Optimization algorithms use smaller search spaces and iterations. In each iteration, previous iteration result is taken into consideration (Myung, 2003).

An important difference between LSE and MLE is the convergence of different estimated values, especially when data sets are not normally distributed. On the corresponding data sets, if the probability density function can be defined, MLE will provide a better solution. Under the same training conditions the same optimized parameter values are obtained from LSE and MLE. They provide normal distribution with a constant variance, only if data are independent of each other (Myung, 2003).

In Ng (2013), least squares regression is described with the help of maximum likelihood estimators under a set of assumptions. In this description, classification is defined by two assumptions as given in (6). These two assumptions can be combined into