Horse Racing Prediction Using Artificial Neural Networks

ELNAZ DAVOODI, ALI REZA KHANTEYMOORI
Mathematics and Computer science Department
Institute for Advanced Studies in Basic Sciences (IASBS)
Gavazang, Zanjan, Iran
IRAN
E_davoodi@iasbs.ac.ir, Khanteymoori@iasbs.ac.ir

Abstract: - Artificial Neural Networks (ANNs) have been applied to predict many complex problems. In this paper ANNs are applied to horse racing prediction. We employed Back-Propagation, Back-Propagation with Momentum, Quasi-Newton, Levenberg-Marquardt and Conjugate Gradient Descent learning algorithms for real horse racing data and the performances of five supervised NN algorithms were analyzed. Data collected from AQUEDUCT Race Track in NY, include 100 actual races from 1 January to 29 January 2010. The experimental results demonstrate that NNs are appropriate methods in horse racing prediction context. Results show that BP algorithm performs slightly better than other algorithms but it needs a longer training time and more parameter selection. Furthermore, LM algorithm is the fastest.

Key-Words: - Artificial Neural Networks, Time Series Analysis, Horse Racing Prediction, Learning Algorithms, Back-Propagation

1 Introduction

Artificial Neural Networks (ANN) were inspired from brain modeling studies. Implementations in a number of application fields have been presented ample rewards in terms of efficiency and ability to solve complex problems. Some classes of applications that ANNs have been applied to include classification, pattern matching, pattern completion, optimization, control, data mining and time series modeling [1]. Time series analysis is used for many applications such as predictions [2].

Prediction is making claims about something that will happen, often based on information from past and from current state. One possible way for predicting future events is finding approximations, for example regression of a predicted variable on other events that is then extrapolated to the future. But finding such approximation can be difficult, too [3]. Also, the relationship chosen for regression tends to be linear. The other difficulty with regression is that, the regression equation applies across the entire span of input space [4]. So, we use NN for prediction, a general method of prediction which avoids these difficulties.

ANNs have been employed to predict weather forecasting, traveling time, stock market and etc. Also, ANNs have been applied in predicting game results, such as soccer, basketball, animal racing, etc.

Some works were done in predicting animal racings. Chen *et al* [5] investigated greyhound racing, it was a very similar environment to horse racing. In their research, they investigated the use of BPNN in predicting races. After that, Jannet Williams and Yan Li in [6] used BPNN to predict horse racing by a network with one hidden layer. Nowadays, horse racing software products, such as Brain Maker, are very popular [7].

In this paper, ANNs have been applied to predict the horse racing in AQUEDUCT Race Track, USA, and acceptable predictions were made. ANN has been used for each horse to predict the finishing time of each horse participating in a race. The best architecture of NN has been investigated by trial and error and different architectures have been compared according to MSE (Mean Squared Error) measure of ten times of running NN for each horse. Therefore, the best architecture is the one which has the minimum MSE measure.

The rest of the paper is organized as follows: The structure of NN and its training algorithms are presented in Section 2. In Section 3 we briefly presented the concept of prediction and some common methods for prediction. Experimental results are presented in Section 4. Finally the paper is concluded in Section 5.

2 Artificial Neural Networks

The brain consists of a large number of highly connected elements called neurons. These neurons have three principal components: the dendrites, the cell body and the axon.

Artificial Neural Networks (ANNs) were inspired from brain modeling studies. An Artificial Neural Network (ANN) is an adaptive system that learns to perform a function (an input/output map) from data. Adaptive means that the system parameters are changed during operation, normally called the training phase. After the training phase the Artificial Neural Network parameters are fixed and the system is deployed to solve the problem at hand (the testing phase) [8].
A neural network has to be configured such that the application of a set of inputs produces the desired set of outputs. Various methods to set the strengths of the connections exist. One way is to set the weights explicitly, using a priori knowledge. Another way is to train the neural network by feeding it teaching patterns and letting it change its weights according to some learning rules [8]. Some learning rules that we use for training our network is presented in the next chapter.

2.1 Learning Algorithms

There are three main types of learning algorithms for training network: Supervised learning where the NN is provided a training set and a target (desired output); Unsupervised learning where the aim is to discover patterns or features in the input data with no assistance from an external source; and Reinforcement learning where the aim is to reward the neurons for good performance and penalize the neuron for bad performance [1].
Supervised learning requires a training set that consists of input vector and a target vector associated with each input vector. The NN learner uses the target vector to determine how well it has learned, and to guide adjustments to weight values to reduce its overall error. The weight updating is generally described as:

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n) \quad (1)$$

Here $\Delta w_{ij}(n)$ is determined by learning algorithm and $w_{ij}(n)$ is initialized randomly. In this paper, the following five supervised algorithms have been employed and their prediction power and their performances have been compared.

2.1.1 Gradient Descent BP Algorithm

Gradient Descent (GD) optimization has led to one of the most popular learning algorithms, namely back propagation [9]. In this algorithm, learning iteration consists of two phases-forward pass, which simply calculates the output(s) value of the NN for each training pattern; and backward propagation, which propagates from the output layer toward the input layer where weights are adjusted as functions of the back propagation error signal.
In this algorithm, the sum squared error (SSE) is used as the objective function, and the error of output neuron j computes as follows:

$$\varepsilon(n)=\tfrac{1}{2}\left(\sum e_j^2(n)\right),\ e_j(n) = d_j(n)-y_j(n) \quad (2)$$

Where $d_j(n)$ and $y_j(n)$ are respectively the target and the actual values of the j-th output unit.
The total mean squared error is the average of the network errors of the training examples.

$$E_{Av}=1/N\left(\sum_{n=1}^{N} \varepsilon(n)\right) \quad (3)$$

A weighted sum a_j, for the given input x_i, and weights w_{ij}, is computed as follows:

$$a_j=\sum_{i=1}^{n} x_i w_{ij} \quad (4)$$

Here n is the number of inputs to one neuron.
The standard sigmoid activation function with values between 0 and 1 is used to determine the output at neuron j:

$$y_j=f(a_j)= \frac{1}{1+e^{-a_j}} \quad (5)$$

Weights are updated according to following equation:

$$w_{ij}(t)=w_{ij}(t-1)+\Delta w_{ij}(t) \quad (6)$$

$$\Delta w_{ij}(t)=\eta\frac{\partial E}{\partial w_{ij}} \quad (7)$$

η is learning rate.

2.1.2 Gradient Descent BP with momentum Algorithm

The convergence of the network by back propagation is crucial problem because it requires much iteration. To mitigate this problem, a parameter called "Momentum", can be added to BP learning method by making weight changes equal to the sum of fraction of the last weight change and the new change suggested by

$$w_{ij}(t) = w_{ij}(t-1) +\Delta w_{ij}(t)+\alpha\Delta w_{ij}(t-1) \quad (8)$$

Where α is momentum term. The momentum is an effective means not only to accelerate the training but also to allow the network to respond to the (local) gradient [9].

2.1.3 Quasi-Newton BFGS Algorithm

In Newton methods the update step is adjusted as:

$$W(t+1)=w(t)–H_t^{-1}g_t \quad (9)$$

Where H_t is the Hessian matrix (second derivatives) of the performance index at current values of weights and biases. Newton's methods often converge faster than conjugate gradient methods. Unfortunately, they are computationally very expensive, due to the extensive computation of the Hessian matrix *H* coming along with the second-order derivatives. The Quasi-Newton method that has been the most successful in published studies is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update [10].

2.1.4 Levenberg-Marquardt Algorithm

Similarly to Quasi-Newton methods, the Levenberg-Marquardt algorithm was designed to approach second-order training speed without having to compute the Hessian matrix. Under the assumption that the error function is some kind of squared sum, then the Hessian matrix can be approximated as:

$$H=J^TJ \quad (10)$$

And the gradient can be computed as:

$$g=J^Te \quad (11)$$

Where *J* is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases. The Jacobian matrix determination is less computationally expensive than the Hessian matrix; e is a vector of network errors. Then the update can be adjusted as:

$$w(t+1) = w(t) –[J^TJ+\eta I]^{-1}J^Te \quad (12)$$

The parameter η is scalar controlling the behavior of the algorithm. For $\eta = 0$, the algorithm follows Newton's method, using the approximation Hessian matrix. When η is high, this becomes gradient descent with small step size [10].

2.1.5 Conjugate Gradient Descent Algorithm

The standard back propagation algorithm adjusts the weights in the steepest descent direction, which does not necessarily produce the fastest convergence. And it is also very sensitive to the chosen learning rate, which may cause an unstable result or a long-time convergence. As a matter of fact, several conjugate gradient algorithms have recently been introduced as learning algorithms in neural networks. They use at each iteration of the algorithm different search direction in a way which produce generally faster convergence than steepest descent direction. In the conjugate gradient algorithms, the step size is adjusted in each iteration. The conjugate gradient used here is proposed by Fletcher and Reeves.

All conjugate gradient algorithms start out by searching in the steepest descent direction on the first iteration.

$$P_0= - g_0 \quad (13)$$

The Search direction at each iteration is determined by updating the weight vector as:

$$w(t+1) = w(t) + \eta_t p_t \quad (14)$$

where:

$$p_t = - g_t + \beta_t p_{t-1} \quad (15)$$

For the Fletcher-Reeves update, the constant β_t is computed by:

$$\beta_t= \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}} \quad (16)$$

This is the ration of the norm squared of the current gradient to the norm squared of the previous gradient [1, 10].

3 Prediction

Predicting is making claims about something that will happen, often based on information from past and from current state. There are many methods for prediction that we consider some of them, for example Time Series Analysis Methods, Regression and Neural Networks. Each approach has its own advantages and limitations.

Time Series (TS) is a sequence of observations ordered in time. Mostly these observations are collected at equally spaced, discrete time intervals [11].

A basic assumption in any time series analysis is that some aspects of the past patterns will continue to remain in the future. So, time series analysis is used for predicting future events according to past events. There are some models in time series analysis that is used for prediction. A new methodology was found called ARIMA (AutoRegressive Integrated Moving Average) [12]. But ARIMA methodology has some disadvantages like needing minimum 40 data point for forecasting, so it's not suitable for horse racing prediction.

Another method is used for prediction is regression. Regression analysis is a statistical technique for determining the relationship between a single dependent (criterion) variable and one or more independent (predictor) variables. The analysis yields a predicted value for the criterion resulting from a linear combination of the predictors [13]. Regression analysis has some advantages and disadvantages. One of the advantages of this method is that when relationships between the independent variable and dependent variable are almost linear, produce optimal results, and one of the disadvantages is that, in this method, all history is created equal, the other disadvantages is that one bad data point in this method can greatly affect forecast [12].

The last method we describe for prediction in NNs. NNs can be used for prediction with various levels of success. The advantages of them include automatic learning of dependencies only from measured data without any need to add further information (such as type of dependency like with the regression). The neural networks are trained from the historical data with the hope that they will discover hidden dependencies and be able to use them for predicting future. Also, NNs are able to generalize and are resistant to noise. From the other point of view, neural Networks have some disadvantages like high processing time for large networks and needing training to operate [3]. So, we use neural networks for our purpose, because of its advantages and excel to other methods.

4 Experimental Results

Nowadays, lots of people follow horse racings and some of this sport's fans bet on the results and who wins the bet, get lots of money. As we mentioned in previous sections, one of NN's applications is prediction and in this paper we use NN for predicting horse racing results.

4.1 Data Base

One of the most popular horse racings which take place in the United States, hold in Aqueduct Race track in New York [14]. In this paper horse racings in January 2010 have been used for predicting the results.

Each horse race has a form which contains information about the race and horses participating in the race [14].

In this paper one neural network have been used for each horse, NN's output is finishing time of the horse. Then, for all horses participating in a race, we have been predicted finishing times and horses' ranks have been got by sorting finishing times in increasing order.

In this paper, 8 features have been used for each horse in train and test phases. Some of these features have numerical values and some others have symbolic values, that symbolic data have been encoded into continues ones. Also, since inputs might have different ranges of values which affect the training process, so we normalized input data as follows:

$$x_i = \frac{x_i - mean\ (x_i)}{var\ (x_i)} \tag{17}$$

We use eight features as inputs of each NN which are listed as: Horse weight, type of race, horse's trainer, horse's jockey, number of horses in the race, race distance, track condition and weather.

Each horse weight is denoted below Wgt column and weight scale is pound (lb.). There are different types of race, that each type has specific distance and conditions.

Horses' trainers are denoted below the past performance table and each horse trainer is specified by his/her horse number. The next feature we need is horse's jockey that is denoted next to horse name in parenthesis.

Race distance depends on type of race and track. In horse race form, race distance is specified at the top of form and under type of race. Race distance scales have been used in these horse race forms are Furlong and Mile and distance scales have been changed into Meter, and changed scaled have been used as input data.

Two other features have been used are track and weather condition which are denoted in the form and we encode them into numerical values for using as input data.

The last point about race information form is the column called Last Raced which contains date and track name that each horse raced last before this race.

4.2 Neural Networks Architecture

In general, there are three fundamentally different classes of network architectures in ANNs- Single-Layer Feedforward Networks (SLFF) which have an input layer of source nodes that projects onto an output layer of neurons; Multilayer Feedforward Networks (MLFF) which have some hidden layers between input layer and output layer; and the last class of network architecture is Recurrent Networks (RN) which have at least one feedback loop that feeding some neuron's output signal back to inputs [9].

There are two types of adaptive algorithms that help us to determine which structure is better for this application: network pruning: start from a large network and successively remove some neurons and links until network performance degraded, network growing: begin

with a small network and introduce new neurons until performance is satisfactory [9].

In this paper, multilayer feedforward neural network have been used and the best architecture that minimizes mean square error (MSE) of the have been reached by network growing method. This network consists of four layers: an input layer that each neuron in this layer corresponds to one input signal; two hidden layers of neurons that adjust in order to represent a relationship; and an output layer that each neuron in this layer corresponds to one output signal. Also, our network is fully connected in the sense that every node in each layer of the network is connected to every other node in the adjacent forward layer.

For gaining the best structure by growing method, at first a multilayer perceptron with one hidden layer and different numbers of neurons were tested (8-neurons-1 structures). After that a layer was added to the network and predicted results of these two-hidden layers networks with different numbers of neurons in both layers were calculated (for example 8-5-neurons-1 structure) . The best structure which has minimum MSE measure was gained with five neurons in the first hidden layer and seven neurons in the second hidden layer (8-5-7-1 structure). Figure 1 shows some of structures that we tested for gaining the best structure, and their MSE measures for a sample race. The best structure is the minimum of black plot in Figure 1.

4.3 Train and Test

Five training algorithms are applied to the horse racing data collected from actual races and their prediction performances are compared. In this paper, a total of 100 races are considered which took place between January 1, 2010 and January 29 in AQUEDUCT Race Track in NY.

In this paper, one neural network is used to represent one horse. Each NN has eight inputs and one output that is horse finishing time. Experimental results are computed by Matlab 7. 6. 0(R2008b). We employ Gradient Descent BP (BP), Gradient Descent with Momentum (BPM), Quasi-Newton BFGS (BFG), Levenberg-Marquadt (LM), and Conjugate Gradient Descent (CGD) and then these learning algorithms results are compared. Table 1 shows the predicting results using all five algorithms in one actual race (Race 7, January 18) and its actual racing results. In the race specified in Table 1, BP, BFG, LM and CGD algorithms determine the horse which ranks first correctly.

In predicting horse racing results, sometimes it is good to know the horse in the last position and help horse racing fans not to bet on that. Experimental results show that CGD algorithm is quite better for predicting the last horse than the other training algorithms we use. Also Experimental results demonstrate that BPM (with momentum value 0.7) and BP algorithms are better than other algorithms for predicting the first horse in the race. Table 2 shows the summary of Experimental results. It is remarkable that BP algorithm produces acceptable predictions with accuracy of 77%. Some parameters influence performance of algorithms and we investigate the influence of these parameters on performance of algorithms.

Epochs: The epochs for training the algorithms are set to 400 as it can provide good results for all.

Hidden units: Number of hidden layers and number of neurons in each hidden layer affect the performance of algorithms. One of the architectures we use in order to obtain the best architecture was used in [5], but predicted results with 8-5-7-1 architecture were better than the results in [5] with 8-13-1 architecture.

Momentum factor: The momentum factor improves the speed of learning and the best value would avoid the side effect of skipping over the local minima [15]. It is found that the momentum value of 0.7 produces optimal results.

In this paper, we had some problems with data set. One of the problems we had was about past performance of horses. Some horses didn't have enough past performances as much as we need. For example, some horses had only one or two past races. This history of a horse was not adequate to predict the next race. So we had to delete these horses from our data set, and sometimes we had to shuffle past races of horses to obtain good results.

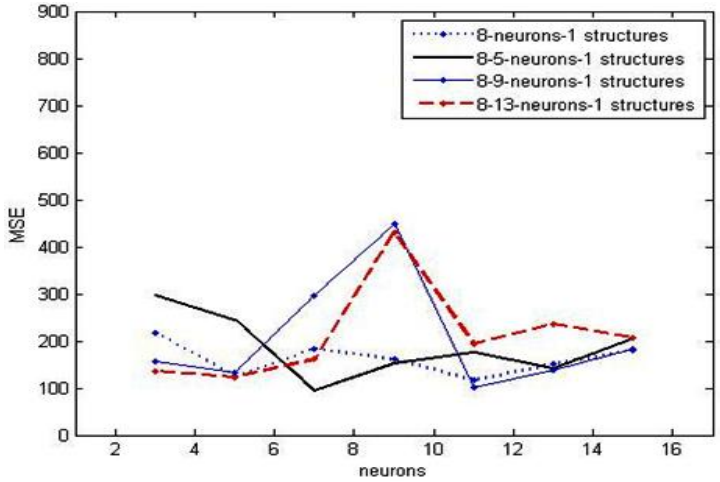

Fig 1: Comparison Neural Networks structures

Table 1: Race Showing Some Prediction (Race 7, January 18)

Show placement	Actual	BP	BPM	BFG	LM	CGD
1st	Out of Respect	Out of Respect	Dahlgren chapel	Out of Respect	Out of Respect	Out of Respect
2nd	Dahlgren chapel	Dahlgren chapel	Jet to Classics	Dahlgren chapel	Persian Ruler	Persian Ruler
3rd	Persian Ruler	Jet to Classics	Out of Respect	Jet to Classics	Jet to Classics	Jet to Classics
4th	Jet to Classics	Persian Ruler	Persian Ruler	Persian Ruler	Dahlgren chapel	Dahlgren chapel

Table 2: Experimental Results

	BP	BPM	BFG	LM	CGD
First Position in actual race results predicted correctly	39	39	35	29	32
Last Position in actual race results predicted correctly	30	29	27	22	37
1 horse in actual results predicted correctly	43	41	47	44	33
2 and more than 2 horses in actual results predicted correctly	33	23	25	23	37
No horse in actual results predicted correctly	24	31	28	33	30

5 Conclusion

In this paper, we applied Neural Networks approach to horse racing prediction context, which consist in building a Neural Network model and analyzing the performance of learning algorithms. We investigated the performances of five supervised neural network algorithms include Back-Propagation, Back-Propagation with Momentum, Quasi-Newton, Levenberg-Marquardt and Conjugate Gradient Descent. The experimental results demonstrate that NNs are appropriate methods for horse racing prediction. All algorithms can produce acceptable prediction with and accuracy of 77% by average. The performance of each model has been influenced by the Neural Network structure and learning algorithm. As a result, it was shown that BP algorithm slightly performs better than other algorithms but need a longer training time. Furthermore, LM is the fastest.

Refrences

[1] Engelbrecht, A. P., Computational Intelligence, University of Pretoria, 2007.

[2] Brockwell, P.J., Davis, R.A.: Introduction to time-series and forecasting, Springer-Verlan, New York, 1996.

[3] Pattreson D., Artificial Neural networks - Theory and Applications, Prentice Hall, 1996.

[4] Bhadeshia, H. K. D. H., and Sormail, T., Neural Networks.

[5] Chen H., Buntin, P., *et .al:* Expert Prediction, Symbolic Learning and Neural Network. An Experiment on Greyhound Racing, IEEE, EXPERT Vol. 9 No. 6, (1994) 21-27

[6] Williams, J ., and Li, Y .,"A case study using neural network algorithms: Horse racing prediction in Jamaica",2008.

[7] Sydenham P. H. and Thorn R. ,Handbook of measuring system and design, Published by Wiley, 2005.

[8] Hagan, M. T., H. B. Demuth, and M. H. Beale, Neural Network Design, Boston, MA: PWS Publishing 1996.

[9] Haykin, S., Neural Networks, Published by Printice-Hall , 1999.

[10] Zayani, R., Bouallegue, R., Raviras, D., "Levenberg-Marquardt learning neural network for adaptive pre-distortion for time varying HPA with memory in OFDM systems", EUSIPCO, 2008.

[11] Box G E P and Jenkins GM., Time series analysis: forecasting and control, Holden-Day, 1976.

[12] Pedhazur EJ. Multiple Regression in Behavioral Research. 3rd ed. Fort Worth, TX: Harcourt Brace College Publishers; 1997.

[13]Palmer, Phillip B.,O'Connell, Dennis G., "Regression Analysis For Prediction: Understanding the process", Cardiopulmonary Physical Therapy Journal,2009.

[14] www.equibase.com. EQUIBASE company.

[15]Smith, L: An Introduction to Neural Network. www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.htm