Ivan Lirkov
Svetozar Margenov (Eds.)

LNCS 10665

Large-Scale Scientific Computing

11th International Conference, LSSC 2017
Sozopol, Bulgaria, June 5–9, 2017
Revised Selected Papers

Editors
Ivan Lirkov
Institute of Information and Communication Technologies
Bulgarian Academy of Sciences
Sofia
Bulgaria

Svetozar Margenov
Institute of Information and Communication Technologies
Bulgarian Academy of Sciences
Sofia
Bulgaria

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-319-73440-8 ISBN 978-3-319-73441-5 (eBook)
https://doi.org/10.1007/978-3-319-73441-5

Library of Congress Control Number: 2017962887

LNCS Sublibrary: SL1 – Theoretical Computer Science and General Issues

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Hybrid Approach Based on Combination of Backpropagation and Evolutionary Algorithms for Artificial Neural Networks Training by Using Mobile Devices in Distributed Computing Environment

Iliyan Zankinski(✉), Maria Barova, and Petar Tomov

Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Acad. G. Bonchev Str., Block 2, 1113 Sofia, Bulgaria
iliyan@hsi.iccs.bas.bg

Abstract. When Evolutionary Algorithms (EAs) are used for Artificial Neural Networks (ANNs) training, the most valuable advantage is the potential for this training to be done in parallel or even using distributed computing. With the capabilities of modern mobile devices, for example their use for distributed computations, they can be used much more extensively for scientific calculations. It is well known that distributed computing systems are limited by their communication bandwidth, because of network latency. In such environment some EAs are pretty suitable for distributed implementation. This is because of their high level of parallelism and relatively less intensive network communication needs. Subset of distributed computing is volunteer computing where users donate some of the computing power provided by devices under their control. This research proposes Android Live Wallpaper volunteer computing implementation of a system used for financial time series prediction. The forecasting module is organized as ANN, which is trained by hybrid combination of Backpropagation and EAs.

Keywords: Artificial Neural Networks · Evolutionary Algorithms
Distributed computing

1 Introduction

ANNs are very common in the field of machine learning. In its nature ANN training is an optimization problem. When the searching space is too big, global optimization EAs as Genetic Algorithms (GAs) can be very suitable. In the last two decades there are numerous attempts to combine EAs based optimization with ANNs training. This combination shows to be much more promising when it is implemented as distributed computing system [1–3]. With the rising popularity of mobile devices a lot of new possibilities for distributed computing can be investigated. Successful desktop distributed computing projects, as described

I. Lirkov and S. Margenov (Eds.): LSSC 2017, LNCS 10665, pp. 425–432, 2018.
https://doi.org/10.1007/978-3-319-73441-5_46

in [4,5] can be efficiently implemented on mobile devices. This idea can be applied to EA based ANN training into a mobile distributed computing environment [6]. In this paper a genetic algorithm in combination with backpropagation artificial neural network training is described. The training is done on a mobile devices as active Android wallpaper application. In addition, the traditional ANN's sigmoid function is replaced by fading sinusoidal function. The successful application of this hybrid approach is documented by series of experiments.

This paper is organized as follows. Section 1 gives an overview of the problem with a special emphasis on ANNs/GAs and their strengths/weaknesses when applied for the problem's solution. Section 2 introduces a distributed computing system based on mobile devices. Experiments and results are presented in Sect. 3. The final Sect. 4 concludes and some further work suggestions are provided.

1.1 Financial Time Series Forecasting

A time series is a collection of measurements made in a sequence through time. There are many examples for time series like the sales of the specific stock in successive months, the average daily temperature at a particular location, the electricity consumption in a particular areal for successive seasons and many other. In the field of the finance, decision makers are in the position of taking very responsible steps during formation of investment strategy. To invest it means to take acceptable risk with expectation of certain amount of profit. The most important aspect of investment is the balance between risk taken and expected profit. On the currency market (FOREX) the main trading is done by exchanging currencies. Currency is the most volatile in price changing object of trading. During the process of trading on a market as FOREX decision makers needs to take three important decisions: 1. Price will go up or down; 2. What volume to buy or sell; 3. How long to keep the opened position. Even if it sounds simple in fact it is very difficult to estimate price changing direction, because of the huge number of factors influencing it. The order volume is directly related to the amount of risk taken. High volume order can lead to high profit if price changing direction is well estimated, but it can lead to high loss in the other case. How long to keep the opened position is related to making the profit even bigger or making the loss as small as possible. Financial forecasting is most important for the traders on the currency market, because of the high price dynamics [2].

1.2 Artificial Neural Networks

Artificial Neural Network is a mathematical model inspired by research into the nature of the human brain. ANNs consists of five components: 1. Network topology which is represented by a directed graph where arcs are referred as links; 2. A variable which represents the state of each node; 3. For each link a real value variable represents its weight; 4. A real value bias which is supplied to each node; 5. Node transfer function which determines the state of the node. The transfer function consists of an activation function (in most cases sigmoid

or hyperbolic tangent). The activation function accepts the sum of the inputs multiplied by the weights of the links between the node and the input nodes.

The input nodes in feed-forward network do not have input arcs. The input nodes should be supplied with values. After that the input information can be spread across the network. The other nodes are changing their state variable according to the propagation rules. In multilayer perceptron (MLP) any path from an input node to an output node traverses the same number of arcs. A hidden layer is a group of nodes which are neither input nor output and are on the same number of arcs distance from the input layer and the output layer. MLP is fully connected if each node in a particular layer is connected to each node in the previous layer.

Research shows that such MLP generalize well in practical problems. When trained on a relatively sparse set of data points, they often provide the right output for an input not presented in the training set. Secondly, gradient based training algorithms can be successfully applied in order to find a good set of weights in acceptable amount of time. The advantage is in calculating the gradient of the error according to the weights for a given input by back propagating the error through the network. Gradient based training works well on simple training problems, but when the problem complexity increases (most commonly because of the increased dimensionality and/or greater data complexity) the performance of the gradient based training falls off rapidly [7].

The performance slow down seems to come from the fact that complex spaces have nearly global optimum around the local optimum. Gradient search techniques tend to get trapped at the local optimum. With a high enough gain (or momentum), backpropagation can escape these local optima. However, it leaves them without knowing whether the next one it finds will be better or worse. When the nearly global optima are well hidden among the local optimum, backpropagation can end up bouncing between local optima without much overall improvement, thus making for very slow training. Another drawback of the gradient based training is the requirement for the activation function differentiability. Backpropagation can not handle discontinuous functions or discontinuous node activation functions.

1.3 Genetic Algorithms

Genetic Algorithms are meta heuristics for global optimization inspired from the ideas of biological evolution. They have five components: 1. Approach for encoding the problem in terms of chromosomes (population of individuals); 2. An evaluation function which is used to determine survival capabilities of each chromosome; 3. A strategy for initial population initialization; 4. Set of operator applied over parent chromosomes in order reproduction to appear (in most cases—selection, crossover, mutation and/or domain specific genetic material recombination); 5. Set of parameters applied over the population and the operators.

GA applies these components and operates in the following steps:

1. Initialization of the population (random or some prior information).
2. Chromosomes evaluation (relative ranking as result in this step).
3. Epochs of recombinations are executed until stopping criteria is met.
   3.1 Stochastic parents selection (parents with better fitness are preferred).
   3.2 Children are produced by recombination operators over the parents.
   3.3 Evaluation of the children is done to keep some of them in the population. Some times the elitism rule is applied by keeping the best found individuals to survive until the end of the evolution.

If appropriate problem encoding and recombination operators are selected the algorithm is capable of producing better and better solutions, converging finally on results close to a global optimum. In most of the real life problems (as the problem presented in this research) standard operators as crossover and mutation are sufficient. In this case GA can be used as black-box optimizer. It means that no specific knowledge of the problem domain is needed. As it is shown in this research by involving problem specific operators (crossover and mutation), the optimization efficiency can be improved.

GA does not have a scaling problem as it is the case with ANN [8]. Even more, GAs are very suitable for parallel computing and even for distributed computing, because GA improves the current best candidate monotonically. It is done by keeping the best found candidates as part of the population while searching for better candidates continues. GAs are not threatened by getting stuck in a local optimum. The mutation and crossover operators can step from a valley across a hill to an even lower valley with no more difficulty than descending directly into a valley.

1.4 Neuron Activation Function Proposal

The most often used activation functions in MLPs are the sigmoid function and the hyperbolic tangent function [9]. In this research sin function with exponent fading effect is proposed as neurons activation function (Fig. 1). Because of its fading effect neurons are more active in a specific input range. If the input is highly positive or highly negative the neuron stops acting as effect of over saturation.

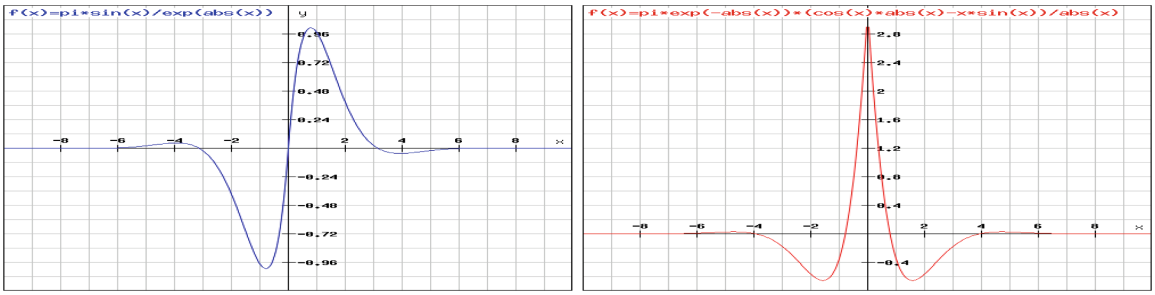
$$f(x) = \frac{\pi \sin(x)}{e^{|x|}} \tag{1}$$

The proposed function (Eq. 1) is differentiable (Eqs. 2 and 3) which is one of the common requirements in gradient based training of ANNs.

$$\frac{d}{dx} f(x) = \frac{\pi(|x| \cos x - x \sin x)}{|x| e^{|x|}} \tag{2}$$

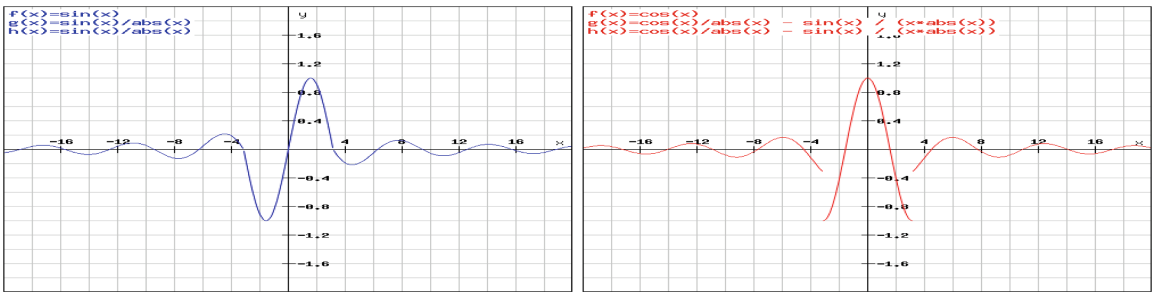
Function derivative can be expressed by the function itself as it is shown in Eq. 3.

$$\frac{d}{dx} f(x) = \begin{cases} f(x+\pi) - f(x), & x > 0 \\ f(x+\pi) + f(x), & x < 0 \\ +\infty, & x = 0 \end{cases} \tag{3}$$



(a) Function (b) Derivative

Fig. 1. Exponent regulated sin activation function and its derivative.



(a) Function (b) Derivative

Fig. 2. Fading sin activation function and its derivative.

The advantage of such activation function is that each neuron has loading level which it is capable to handle.

If periodic processes should be modeled with ANN it is possible to apply sin fading function (Fig. 2).

$$f(x) = \begin{cases} \sin(x), & -\pi <= x <= +\pi \\ \frac{\sin(x)}{|x|}, & +\pi < x < -\pi \end{cases} \tag{4}$$

The problem with the sin fading function is that it has break points at $-\pi$ and $+\pi$.

$$\frac{d}{dx} f(x) = \begin{cases} \cos(x), & -\pi <= x <= +\pi \\ \frac{\cos(x)}{|x|} - \frac{\sin(x)}{x|x|}, & +\pi < x < -\pi \end{cases} \tag{5}$$

The components of the function (Eq. 4) are differentiable functions (Eq. 5). This small inconvenience is not a problem when calculations are done with discrete number calculating computers. Representing real numbers in the computer memory is less accurate than the problems of analytic differentiability of the sin fading function.

2 Mobile Devices Distributed Computing

The latest developments in mobile devices technology have made smartphones as the future computing and service access devices. Users expect to run

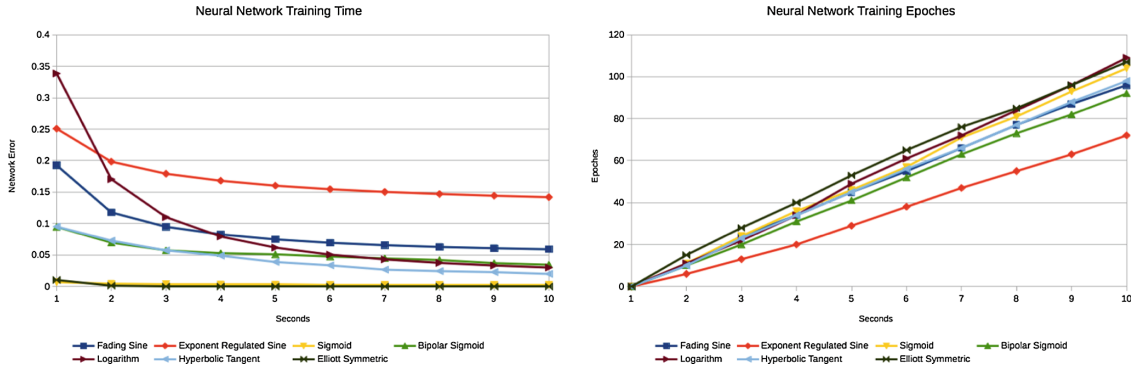
computationally intensive applications on Smart Mobile Devices (SMDs) in the same way as powerful stationary computers [10]. Mobile Cloud Computing (MCC) is the latest practical solution for alleviating this incapability by extending the services and resources of computational clouds to SMDs on demand basis. In MCC, application offloading is ascertained as a software level solution for augmenting application processing capabilities of SMDs [10].

The proposed mobile client application is based on the capabilities of the Android Live Wallpaper [6]. On a regular basis a wallpaper service wakes up. At each wake up a single forecast is done. This forecast is then visualized as part of a wallpaper image. The wake up finishes a single ANN training epoche. The forecasting model is organized as a three-layers MLP instantiated as Encog framework Java object. Local ANN gradient based training is organized as resilient backpropagation which is provided by Encog framework. Training examples are generated from time series by dividing the values in a lag frame (values in the past) and lead frame (values in the future). In this way ANN tries to learn lag and lead patterns [11]. All input data are normalized in the range of the used activation function (in this case -1 and $+1$, as shown in Fig. 1). Different instances of the ANN weights are presented as GA chromosomes. The error of ANN forecasts is used as fitness function in the local GA. On the side of the mobile client GA is represented as Java objects provided by Apache Commons Math Genetic Algorithms framework. In order for ANN training to be continuous and to be independent from Internet network connection, all ANN training examples are stored in a local SQLite database. The communication with the remote server is done only when there is better local solution. The information exchange between the client and the server is organized as HTTP sessions. All data exchanges are packaged as JSON messages. On the server side there is a PHP/MySQL based application which is described in [2].

As summary the weights of an ANN (as Encog Java objects) are optimized with a GA (as Apache Framework Java objects) and this optimization is done on the Android mobile device. Optimization results are collected on the remote PHP/MySQL server.

3 Experiments and Results

All experiments are done with Encog framework by using Java programming language [12]. MLP neural network (256-64-10 topology) is trained with examples of handwritten digits. Data set consists of 1593 handwritten digits from around 80 persons were scanned, stretched in a rectangular box 16×16 in a gray scale of 256 values [13]. Experiments are done on a single CPU core with the following parameters of the machine: Intel Core i7-4790 - 3.6 GHz - 4 cores - 8 threads, 8 GB RAM, Microsoft Windows 10, Encog Core v3.3.0 - Java Version, Java 8 Update 112 (64 bit).



(a) MLP Error (b) Training Epochs

Fig. 3. Activation functions efficiency.

Comparison is done between seven activation functions - Fading Sin, Exponent Regulated Sin, Sigmoid, Bipolar Sigmoid, Logarithm, Hyperbolic Tangent, Elliott Symmetric. As it is shown in Fig. 3, between 70 and 110 training epoche are done for 10 s of training. The performance of exponent regulated sine is slower than the performance of fading sine. The network error for both functions decreases slower than the others, which is very logical because each neuron is limited in the activation signal which can be handled. This limitation gives better responsibilities separation between the neurons in each layer.

4 Conclusions

Even though GA based ANN training is slower than back propagation, when it is implemented as simultaneous computations in a distributed environment it can be efficient enough for practical use. Because modern mobile devices are used on a 24/7 basis to use them as distributed computing network is very cost effective if the approach is based on donated computational power. As further development it will be interesting if mobile devices distributed computing is combined with supercomputers. The central node will be able to provide access to a super computer. Such hybrid infrastructure can provide interesting research and industrial challenges. Another very interesting direction of further development is the capabilities of Android widgets. The client side calculations are possible to be done in a widget instead of a wallpaper. Android widgets are interactive components in the graphical user interface. Such computational widget can be used for user voting or even user's guess for the future forecast. The collection of human opinion will transfer this software solution into the human-computer based distributed computing.

Acknowledgements. This work was supported by private funding of Velbazhd Software LLC.

References

1. Balabanov, T., Zankinski, I., Barova, M.: Strategy for individuals distribution by incident nodes participation in star topology of distributed evolutionary algorithms. Cybern. Inf. Technol. **16**(1), 80–88 (2016). Sofia, Bulgaria
2. Balabanov, T., Zankinski, I., Dobrinkova, N.: Time series prediction by artificial neural networks and differential evolution in distributed environment. In: Lirkov, I., Margenov, S., Waśniewski, J. (eds.) LSSC 2011. LNCS, vol. 7116, pp. 198–205. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29843-1_22
3. Balabanov, T.: Heuristic forecasting approaches in distributed environment (in Bulgarian). In: Proceedings of Anniversary Scientific Conference 40 Years Department of Industrial Automation, UCTM, Sofia, Bulgaria, pp. 163–166 (2011)
4. Balabanov, T.: Distributed evolutional model for music composition by human-computer interaction. In: Proceedings of International Scientific Conference UniTech15. University publishing house "V. Aprilov", Gabrovo, Bulgaria, vol. 2, pp. 389–392 (2015)
5. Balabanov, T.: Avoiding local optimums in distributed population based heuristic algorithms (in Bulgarian). In: Proceedings of XXIII International Symposium Management of Energy, Industrial and Environmental Systems, pp. 83–86. John Atanasoff Union of Automation and Informatics, Sofia, Bulgaria (2015)
6. Balabanov, T., Zankinski, I., Barova, M.: VitoshaTrade Distributed Computing Android Wallpaper, Sofia, Bulgaria (2017). https://github.com/TodorBalabanov/VitDisComp
7. Bas, E.: The training of multiplicative neuron model based artificial neural networks with differential evolution algorithm for forecasting. J. Artif. Intell. Soft Comput. Res. **6**(1), 5–11 (2016)
8. Magnusson, K., Olsson, T.: Training artificial neural networks with genetic algorithms for stock forecasting. A comparative study between genetic algorithms and the backpropagation of errors algorithms for predicting stock prices. Dissertation (2016). http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-186447
9. Karlik, B., Vehbi, A.: Performance analysis of various activation functions in generalized MLP architectures of neural networks. Int. J. Artif. Intell. Expert Syst. (IJAE) **1**(4), 111–122 (2011)
10. Shiraz, M., Gani, A., Khokhar, R., Buyya, R.: A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing. IEEE Commun. Surv. Tutorials **15**(3), 1294–1313 (2013)
11. Qiu, M., Song, Y.: Predicting the direction of stock market index movement using an optimized artificial neural network model. PLoS ONE **11**(5), e0155133 (2016). https://doi.org/10.1371/journal.pone.0155133
12. Zankinski, I.: Encog digits classification resilient training example with sin activation functions, Sofia, Bulgaria (2017). https://github.com/iliyanzan/DigitsResilient
13. Buscema, M., Terzi, S.: Semeion Handwritten Digit Data Set. Center for Machine Learning and Intelligent Systems, California, USA (2009). http://archive.ics.uci.edu/ml/datasets/Semeion+Handwritten+Digit