Prediction of Cluster System Load Using Artificial Neural Networks

Y.S. Artamonov[1]

[1]*Samara National Research University, 34 Moskovskoe Shosse, 443086, Samara, Russia*

Abstract

Currently, a wide range of high-performance environments is available for a researcher to perform computations. It is a really difficult task to select an environment in which the computations will be completed as soon as possible. To solve this, you need to analyze the load of the environment computing resources, and also to predict their availability in the future.
In this paper we describe a solution for prediction of computing resources load in a cluster environment using neural network models. We considered a process of configuring the neural network architecture: selection of activation functions, algorithms of initialization and updating of the weights of neurons. Training and testing was performed on a set of data for the load of the cluster "Sergey Korolev" for the period from November 2013 to December 2016.

Keywords: load prediction; cluster; neural network; model

1. Introduction

Recently, many studies have been devoted to forecasting the load of various computational resources, such as CPU cores [1], individual nodes of a cluster or clouds [2]. Load prediction in cloud and cluster environments is a critical problem that needs to be solved to achieve high performance, since a lot of processes depend on its effective solution, such as resources planning, maintenance periods and modernization of machines and even whole data centers. For instance, without prediction of the availability of shared resources it is impossible to effectively use classic cluster environments where users use shared nodes with different performance and features taking them partially or completely by their computations.

In the previous paper [3] we solved the task of forecasting the load of computational resources using the EMMSP model and examined the applicability of this model. As we noticed, the model is well suited for predictions only on specific data and load history points, but also we showed that it can be effectively used as a component of a simple model mixture: adaptive selection and adaptive composition. This paper considers the use of neural network models to predict the load of cluster resources and compares this approach with that demonstrated previously.

2. Neural network prediction models

Neural network prediction models are based on the use of neural networks that can be trained in regression problems and produce the output value, based on some input parameters, approximating the unknown functional dependencies of the output data on the input.

Neural network models were used in papers [4] and [5] to predict the load of resources with different nature: CPU servers and electrical networks. In both problems, neural network models showed good results and were recognized as effective and adequate to the prediction problem. Given these results, let's look at how well neural network models are suitable for predicting the number of loaded cluster nodes.

Neural network models were chosen for this study because of peculiarities of the task and historical data collected by us. We took into account the following aspects:

- time series of resources load are non-stationary,
- there are templates and periodic components in historical data, as well as segments with low and high load, corresponding to weekends, holidays and work days,
- time series have known minimal and maximum value.

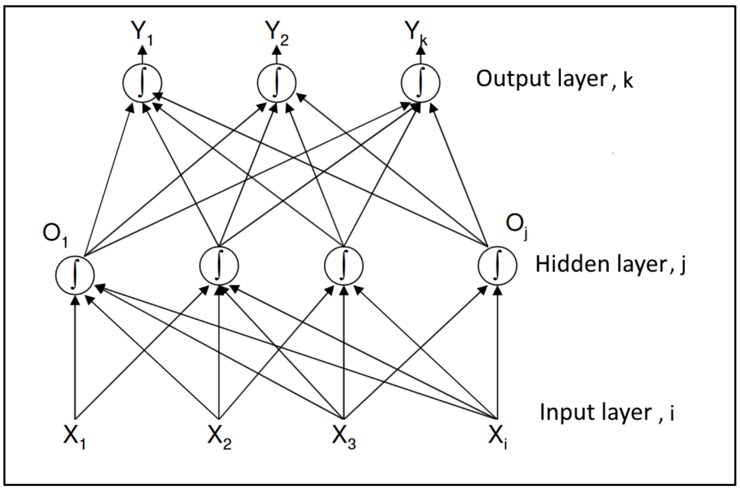
To predict values of time series of this nature we can use neural networks, in fact solving the approximation problem of an unknown function. Taking into account the papers [6] and [7] that apply neural networks for solving forecast problems of similar in nature time series (load of computational resources of cluster / cloud environments), we chose to study the model of a multilayer perceptron (MLP) with single (SL MLP) and two hidden layers (DL MLP).

MLP consist of neurons and connections between them (fig. 1). Neurons have a special transformation function – activation function, each connection has characteristic called weight. Output signal of a neuron in a layer Z of MLP is determined using equation 1[8]:

$$z_j = f(\sum_{i=1}^{N} w_{ij} u_i) \qquad (1)$$

where u_i – output signals of the layer Z, w_{ij} – weights of connections between *i* neuron of the previous layer and *j* neuron of the layer Z, *f* – activation function, z_j – output signal of a neuron. In this paper, we used hyperbolic tangent function as an activation function for neurons of hidden layers.

The training of a neural network is a process of changing the weights of the neuron connections. The main goal of a learning algorithm is to find a configuration of the weights of all the links where the error function is minimized. In the task that we solve we use MSE (Mean Squared Error) as a criterion for model training using gradient descent method, as a final benchmark of a model we use MAE (Mean Average Error) since time series contains a lot of segments with zero value or sequential equal



values, that is why we cannot use MASE (Mean Average Scaled Error) and MAPE (Mean Absolute Percentage Error).

Fig. 1. Structure of MLP with one hidden layer.

We use DeepLearning4j library for training and testing models based on MLP. This library provides battle proven tools and algorithms for training and usage of various artificial neural networks, including the most popular neural network architectures, learning and optimization algorithms. The library is written in Java and uses native extensions for computations on the CPU and GPU to provide the required performance [9]. The DeepLearning4j library is licensed under the Apache License 2.0, this enables us to use it in any applications including commercial, the open source development approach attracts a large number of researchers and improves the quality of the library.

3. Configuring network architecture and learning parameters

To train neural MLP networks the method of back propagation of the error is used with various modifications. The method is an iterative gradient algorithm that is used to minimize the MLP error and to obtain the desired output values. The essence of the method consists in propagation of error signals from the outputs of the network to its inputs, back to direct propagation of signals in the usual mode of operation [10].

Primary parameters of the method and its modifications are:

- learning epochs count,
- learning rate,
- weights initialization algorithm,
- weights update algorithm,
- optimization algorithm,
- learning momentum.

In the task, we need to predict the number of occupied cluster nodes in several of the most intensively used groups of nodes. Target prediction interval – 12 hours, we need to predict 12 points of a time series, one mean value of cluster group load per one hour. We chose qdr_tmp and ddr_tmp cluster groups for training and prediction of a group load. Their load is of the greatest interest because of a large regular load.

To compare the learning methods with different modifications we chose the training parameters presented in Table 1. We compared the training of SL MLP and DL MLP models in the prediction task with 12 points of cluster load (each point – mean load of a cluster group for 1 hour). In training and forecasting, only time series data were considered and passed to neural network inputs. The optimal number of inputs, selected experimentally, is 6.

In the process of training, we fed to the input of the neural network various sets of consecutive 6 values of the series; we used random order of data sets. For each test set, 12 values were generated at the output of the neural network, which were compared with 12 values from the test set. Parameters $i = 6$, $k = 12$.

Table 1. Experimental learning parameters.

Model	Learning epochs	Learning rate	Momentum	Inputs count	Hidden layer neurons count
SL MLP	300	0.01	0.9	6	15
DL MLP	400	0.01	0.9	6	1st: 20 2nd: 10

The backward propagation of errors method is subject to the following problems:

- slow convergence,
- convergence to local minima,
- overfitting.

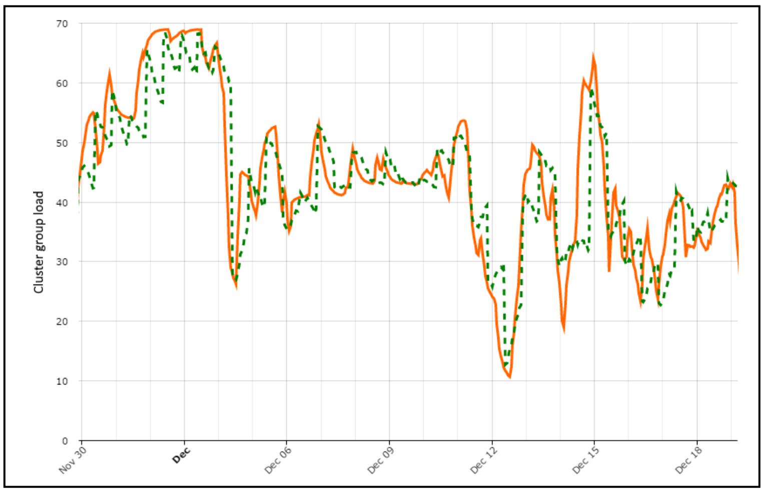


Fig. 2. The forecast of resources load of the cluster "Sergey Korolev" using SL MLP model.

Modifications to the method of back propagation of errors with momentum and various updating algorithms of the link weights, such as Adadelta, enable us to fix or partially fix the above problems, accelerate training and reduce the error of MLP based prediction models.

In the study of neural network models we considered various configurations of training neural networks by the method of back propagation of errors. As parameters of the configuration in the training we used: the algorithm for initializing the weights of neurons, the algorithm for updating the weights of neurons and the optimization algorithm.

We tested 2 options for initializing the balance: uniform distribution (Uniform) and using the Xavier method. The following algorithms for updating the weights of neurons were tested: Nesterov Accelerated Gradient (Nesterovs), adaptive gradient descent (Adagrad), Adaptive learning rate (Adadelta), adaptive momentum estimation (Adam). Two optimization algorithms were tested: linear gradient descent (LGD) and stochastic gradient descent (SGD). These optimizations and the parameters of the gradient descent method and the back propagation of errors algorithm are described in paper [11].

The test used a training sample of length 6000 points and a test sample with a length of 1000 points, the sample data were obtained for the period from January 1, 2015 to January 1, 2016. The results of testing models with different learning parameters for solving the task of forecasting the cluster load are presented in Table 2, the RMSE (Root Mean Square Error) error values are given to estimate the dispersion of the forecast values.

Table 2. RMSE and MAE error values depending on neural network training configuration.

Weights initialization	Weights updater	Optimization algorithm	SL MAE	DL MAE	SL RMSE	DL RMSE
UNIFORM	NESTEROVS	LGD	8,03	7,99	9,28	9,24
XAVIER	NESTEROVS	LGD	8,05	8,20	9,30	9,38
UNIFORM	NESTEROVS	SGD	8,02	7,64	9,28	8,94
XAVIER	NESTEROVS	SGD	8,06	7,70	9,32	8,97
UNIFORM	ADADELTA	LGD	9,71	11,15	10,99	12,11
XAVIER	ADADELTA	LGD	9,62	11,77	10,78	12,75
UNIFORM	ADADELTA	SGD	15,09	8,33	16,08	9,86
XAVIER	ADADELTA	SGD	17,44	12,52	18,61	14,64
UNIFORM	ADAGRAD	LGD	13,24	11,52	13,61	12,64
XAVIER	ADAGRAD	LGD	15,24	9,70	16,36	12,13
UNIFORM	ADAGRAD	SGD	19,04	10,60	21,36	19,23
XAVIER	ADAGRAD	SGD	17,21	11,21	18,12	17,22
UNIFORM	ADAM	LGD	8,10	7,83	9,34	9,13
XAVIER	ADAM	LGD	8,14	7,91	9,38	9,18
UNIFORM	ADAM	SGD	7,99	7,54	9,26	8,84
XAVIER	ADAM	SGD	8,09	7,56	9,34	8,85

Table 2 shows the results of testing the modifications of the method of back propagation of errors, the 3 best results for each model are highlighted with underscores. From these results, we can conclude that the most effective modifications of the back propagation of errors method for the task of forecasting the cluster load are:

1. stochastic gradient descent with initialization of weights using uniform distribution and updating of weights using the ADAM algorithm – for SL MLP and DL MLP models,
2. linear gradient descent with initialization of weights using uniform distribution and updating of weights using the Nesterov method with momentum – for SL MLP model,
3. stochastic gradient descent with initialization of weights using the Xavier method and updating of weights using the ADAM algorithm – for DL MLP model.

The results of DL and SL MLP models differ slightly, which is probably due to the peculiarity of the test data.

4. Comparison of model errors

An example of forecasting cluster load data for a neural network with a single hidden layer is shown in fig. 2, a neural network with two hidden layers - in fig. 3. The dashed line shows the forecast values of the series. The graphs of the forecast values were obtained by calculating the forecast every 12 points.

As the final error metric, the mean absolute error (MAE) is selected, because the relative forecast error (MAPE) can not be used in series that include values close to or equal to zero. The distribution of MAE errors in the SL MLP and DL MLP models is shown in fig. 4, the distribution of errors of both models is close to normal.

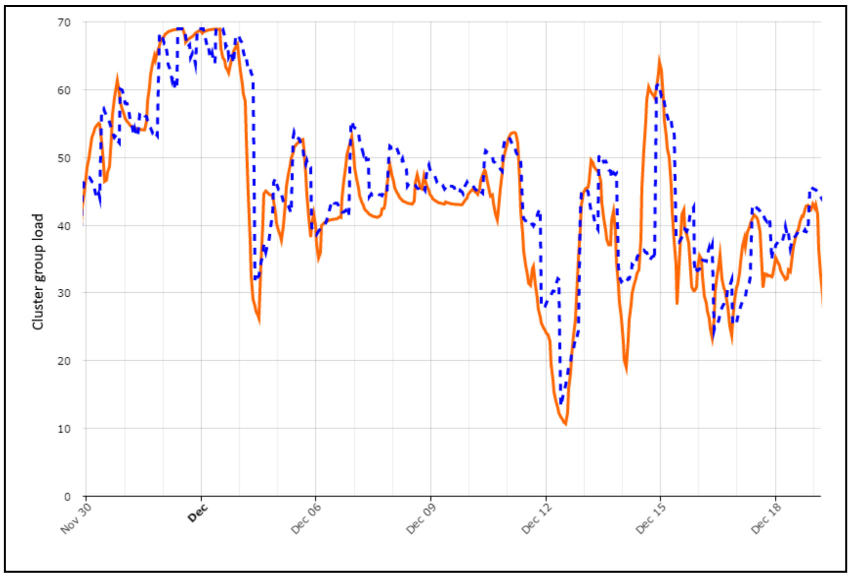


Fig. 3. The forecast of resources load of the cluster "Sergey Korolev" using DL MLP model.

Previously, the task of forecasting 12 cluster load points was solved by the time series prediction method using the maximum resemblance sample (EMMSP) [3]. The MAE prediction errors for method comparison are given in Table 3.

In addition to direct comparison of models, we tried to use all three models (EMMSP, SL MLP, DL MLP) together. In order to do this, we put forward a hypothesis: Each of the models is the best (shows the smallest MAE error) in a certain length of data L > M, where M is the number of prediction points. We tested this hypothesis for the data on which MLP models were tested. Each of the models retains its leadership at the average on a section of 24 to 36 points in length, which corresponds to a time interval of 1 to 1.5 days.

Table 3. MAE errors of different prediction models.

Model	EMMSP	SL MLP	DL MLP	Simple adaptive selection
MAE	8.7	8.02	7.54	6.8

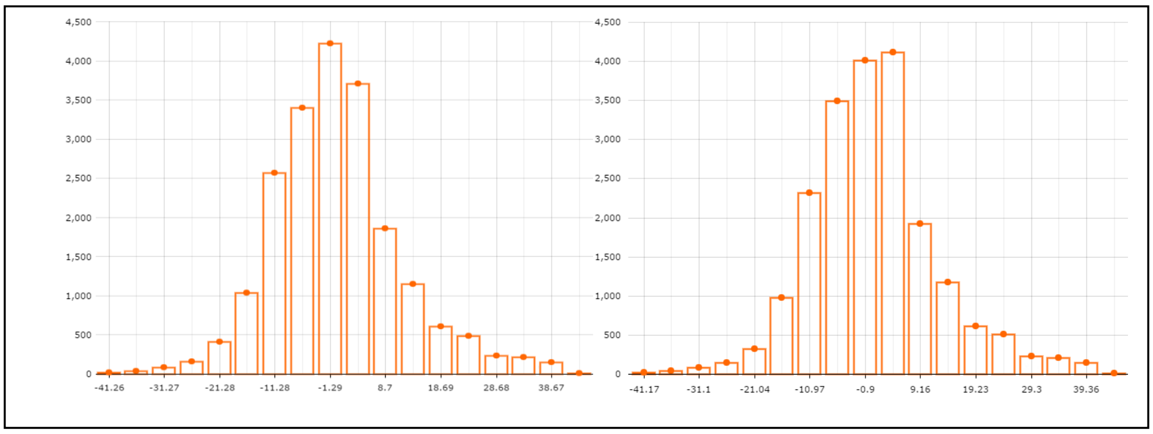


Fig. 4. The distribution of the mean absolute error of models with one hidden layer (on the left) and two hidden layers (right).

The error value for a simple adaptive selective model [12] was obtained for a model that selects the best model for predicting future values by a simple heuristic rule: If one of the models was better in the previous section of the data, then it should be used to predict again. Data for testing were collected between November 2013 and December 2016. The open load monitoring data of the "Sergey Korolev" cluster is available in JSON machine-readable format at: http://templet.ssau.ru/wiki/открытые_данные.

5. Conclusion

The prediction algorithms based on neural network models with one and two hidden layers are integrated into the Templet Web service, which enables users to estimate the task launch time. The forecast graphs and cluster load history are available to registered users of the system. In the future, we plan to provide users with an interactive hint about the number of available resources and the estimated time to start the task based on the task requirements (nodes, groups, software licenses) specified at the time of adding task to a batch queue.

The results of the cluster load forecasting can be applied to solve several types of tasks:

- increase the efficiency of cluster use (energy efficiency, load efficiency),
- selection of optimal environments and parameters for computations,
- planning of cluster growth and maintenance periods.

Methods of forecasting the loading of computing resources are most in demand now in cloud environments where they can enable commercial companies to reduce server maintenance costs or, on the contrary, to effectively adapt to the growing demands of customers.

Acknowledgements

This work is partially supported by the Russian Foundation for Basic Research (RFBR#15-08-05934-A), and by the Ministry of Education and Science of the Russian Federation within the framework of the State Assignments program (№ 9.1616.2017/ПЧ).

References

[1] Naseera S, Rajini GK, Sunil Kumar Reddy P. Host CPU Load Prediction Using Statistical Algorithms a comparative study. International Journal of Computer Technology and Applications 2016; 9(12): 5577–5582.
[2] Di S, Kondo D, Cirne W. Host load prediction in a Google compute cloud with a Bayesian model. Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. IEEE Computer Society Press, 2012; 21 p.
[3] Artamonov YS. Application of the EMMSP model to predict available computing resources in cluster systems. Bulleten of the Samara Scientific Center RAS 2016; 18(4): 681–687. (in Russian)
[4] Naseera S, Rajini GK, Amutha Prabha N, Abhishek G. A comparative study on CPU load predictions in a computational grid using artificial neural network algorithms. Indian Journal of Science and Technology 2015; 8(35).
[5] Kalaitzakis K, Stavrakakis G, Anagnostakis EM. Short-term load forecasting based on artificial neural networks parallel implementation. Electric Power Systems Research 2002; 63(3): 185–196.
[6] Chandini M, Pushpalatha R, Boraia R. A Brief study on Prediction of load in Cloud Environment. International Journal of Advanced Research in Computer and Communication Engineering 2016; 5(5): 157–162.
[7] Engelbrecht HA, van Greunen M. Forecasting methods for cloud hosted resources, a comparison. Network and Service Management (CNSM). 11th International Conference on IEEE 2015; 29–35.
[8] Hajkin S. Nejronnye seti. M.: Vil'jams, 2006; 1104 p.
[9] Deeplearning4j: Open-source distributed deep learning for the JVM. URL: http://deeplearning4j.org (01.01.2017).
[10] Osovskij S. Nejronnye seti dlja obrabotki informacii. M.: Finansy i statistika, 2002; 344 p.
[11] Ruder S. An overview of gradient descent optimization algorithms, 2016. ArXiv preprint arXiv: 1609.04747.
[12] Lukashin JuP. Adaptive methods of short-term forecasting of time series. M.: Finansy i statistika, 2003; 415 p.