BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **20**, No 4
Sofia • 2020 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2020-0047

HPCWMF: A Hybrid Predictive Cloud Workload Management Framework Using Improved LSTM Neural Network

K. Dinesh Kumar, E. Umamaheswari

School of Computer Science and Engineering, VIT University, Chennai 600127, India
E-mails: kdinesh.kumar2015@vit.ac.in umamaheswari.e@vit.ac.in

***Abstract*:** *For cloud providers, workload prediction is a challenging task due to irregular incoming workloads from users. Accurate workload prediction is essential for scheduling the resources to the cloud applications. Thus, in this paper, the authors propose a predictive cloud workload management framework to estimate the needed resources in advance based on a hybrid approach, which is a combination of an improved Long Short-Term Memory* (*LSTM*) *network and a multilayer perceptron network. By improving the traditional LSTM architecture by using opposition-based differential evolution algorithm and dropout technique on recurrent connection without memory loss, the proposed approach has the ability to perform a better prediction process. A novel hybrid predictive approach is aiming at enhancing the prediction performance of the cloud workload. Finally, the authors measure the proposed approach's effectiveness under benchmark data sets of NASA and Saskatchewan servers. The experimental results proved that the proposed approach outperforms the other conventional methods.*

***Keywords*:** *Cloud computing, Improved LSTM neural network, Multilayer perceptron network, Opposition-based differential evolution, Workload prediction.*

1. Introduction

Nowadays, cloud computing has become one of the major technologies in the research community, educational institutions, entertainment, business, and has become a primary part for users and organizations that are completely dependent on cloud based applications. Cloud technology provides computation-processing resources to processing the data aimed to achieve high-performance computing. In cloud computing, mainly three kinds of services are available such as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). However, in present days cloud technology extends its services, called Everything-as-a-Service (XaaS). Virtualization is the primary feature of cloud computing technology, enabling the physical data center as a dynamic virtual resources such as combining network resources, storage resources, and heterogeneous computing

resources to be distributed. These advantages of virtualization technology can provide different kinds of vigorous expansion, resources automatic deployment, resource provisioning according to needs. The resource allocation process is a major part of the cloud data center, which has significant practical value. This process can save energy consumption, reduce computing costs, and enhance resource utilization efficiency. With the help of virtualization technology, it can create multiple or the same configuration of Virtual Machines (VMs) on a single cloud data center. And also, with the help of the VM migration process effectively, we can reduce the computing cost and power consumption. The primary objective of cloud computing technology is to provide on-demand services while meeting economically to both cloud consumers and cloud providers.

In cloud computing technology, elasticity is one of the main components to provisioning resources dynamically based on the workload. With the help of this feature, the cloud resource management framework can perform scaling operations to satisfy the demand for cloud applications. In the cloud service business, several services providers like Amazon EC2, Microsoft Azure, Google App Engine, Salesforce Developers, and IBM bluemix, etc. provide flexible scaling resources. The dynamic allocation of resources can be performed in two ways, like a reactive approach and proactive approach [1]. In the process of a reactive approach, cloud users can fix the threshold values for under-utilization and over-utilization of resources. Whenever the workload value meets the threshold value, then the auto-scaling process performs the action according to the current state of resources, like remove the virtual machines from cloud services for an under-utilization state or adding the virtual machines to cloud services for the over-utilization state. The main drawback of this process is that the auto-scaling process faces difficulties in performing the scaling operations in the situation of sudden fluctuations in workload. On another side, a proactive approach performs resource scaling operations in advance [2]. The cloud resource management forecasts each cloud service's future workload and allocates the resources to their cloud services according to the predicted value. Currently, different kinds of prediction approaches are available to predict the future workload in the cloud environment. Few types of machine learning approaches are being used for estimating future workload such as ARIMA method, Bayesian method, Support vector machine, Random forest, and Artificial neural networks, etc.

However, deciding the exact amount of resources with proactive approaches during execution time for cloud services is a challenging task and also not trivial. Due to the irregular access to cloud service offered by a SaaS provider, the cloud services get the fluctuations in workload. This kind of situation can lead to either over-provisioning or under-provisioning of resources. In an over-provisioning state, more resources will be allocated to cloud applications than needed resources. In the view of Service Level Agreements (SLAs), it is an advantage for cloud users but for SaaS providers, it is an unnecessary cost and leads to high power consumption and CO_2 emission. In the under-provisioning state, fewer resources will be allocated to cloud applications than needed resources, which lead to SLA violations, dropping the Quality of Services (QoS), and finally, the loss of consumers and revenues. So, an

efficient, proactive approach must predict the accurate future needed resources to reach the QoS.

In order to deal with the issues mentioned above, this paper proposes the development of a hybrid predictive approach to predict the future workload with a high accuracy rate, which is a combination of an improved LSTM network and a multilayer perceptron network. Instead of using a single prediction model to handle the irregular workloads, the proposed model uses a hybrid approach with higher prediction performance over conventional deep learning models. In machine learning methods, deep learning methods have outstanding potential ability in the prediction domain. The LSTM network is one of the sophisticated methods among deep learning methods to effectively deal with time series due to the ability to manage long-term dependency sequences. As a result, the LSTM network has been successfully adopted into many domains, such as speech recognition, artificial intelligence, handwriting recognition, and disease diagnosis, etc. Thus, this paper mainly focuses on the LSTM network's ability to extract the nonlinear workload patterns in the cloud workload prediction domain. The main differences of this proposed work from conventional deep learning models is that: (a) the improved LSTM network which can extract the nonlinear workload patterns effectively while other predictors are conventional neural networks; (b) the proposed framework adopted an Opposition-Based Differential Evolution (OBDE) algorithm in which the Opposition-Based Learning (OBL) method is used to identify the opposition point of each population of Differential Evolution (DE) algorithm. This OBDE algorithm identifies the optimized neuron count in each hidden layer and batch size, which can influence the prediction ability of the LSTM network. Compared with the trial-and-error approach, the OBDE algorithm is able to provide more suitable solutions to enhance the performance of LSTM network; (c): In the view of regularization of LSTM networks, dropout has been applied directly on recurrent connections, which does not lead to loss of long-term memory; (d): Instead of using single prediction model, the proposed hybrid predictive model has more capability to get higher accuracy results.

The main contribution of this paper are presented as follows:

1. To deal with the issues of conventional deep learning models, a novel hybrid predictive approach is developed to handle irregular incoming workloads, which is a combination of improved LSTM network and multilayer perceptron network.

2. To tackle the optimization issue in neural networks, the proposed approach has taken the advantages of the Opposition-Based Differential Evolution (OBDE) algorithm, which is a combination of Opposition-Based Learning (OBL) and Differential Evolution (DE) algorithms. The adopted algorithm helps to set the number of neurons in each hidden layer and batch size so that LSTM network will be optimized in efficient way.

3. To enhance the robustness and generalization performance of the LSTM network for improving the predicting ability, the proposed LSTM network used the dropout method directly on recurrent connections without losing long-term memory.

4. Finally, to evaluate the proposed approach's performance experiments have been conducted under benchmark data sets of NASA and Saskatchewan servers HTTP request arrival rates for multiple prediction intervals. The proposed prediction

approach results are compared to different prediction algorithms. The experimental results proved that the proposed prediction approach can learn the nonlinear workload structure, along with a good accuracy rate, and outperforms the other prediction approaches.

The rest of the paper structured as follows. In Section 2, related work is presented. The methodology of algorithms is presented in Section 3. A proposed method is presented in Section 4 with architecture. Section 5 is presents results and experimental analysis. Finally, conclusion of the paper are explained in Section 6.

2. Related work

In cloud computing, a workload prediction is a challenging process, and many of the researchers addressed the different solutions in two ways, such as traditional statistics and history based forecasts. In traditional statistics, the future value is defined based on average methods. In another way, machine learning algorithms analyze the complete data to determine future value. Many conventional models have been used for workload prediction in cloud computing. Advanced proposed approaches are based on machine learning algorithms, able to analyze a large amount of data in cloud environments [2]. Few algorithms are AutoRegressive Integrating Moving Average model (ARIMA), Artificial Neural Networks (ANN), Bayesian model, Markov models, Random forest method, Support Vector Machine (SVM) model, K-Nearest Neighbor (KNN) model, and reinforcement learning model, etc.

In the view of the machine learning domain, in [3], the authors proposed an improved Linear Regression (LR) to estimate the number of request arrival rate for each cloud application. The proposed linear regression model works based on the self-adaptive parameters. Whenever workload fluctuations occur, the proposed model sets the new computation parameters for the linear regression model. In [4], the authors proposed a bin-packing algorithm to handle the VMs. At the first stage, the proposed approach finds a suitable physical machine to deploy the VM based on the predicted memory usage of VM. In the next stage, all physical machines will be minimized based on the future memory usage of each VM to provide sufficient memory for a new VM. To predict the future memory usage of VM, the authors used the AutoRegression (AR) model and to deploy the VMs into a physical machine used bin-packing algorithm. The main feature of this model is VM parameters updated frequently for each new VM. In [5] the prediction algorithm using neural networks is proposed. The method being proposed estimates the future needed CPU utilization for each cloud service. So that, SaaS providers are able to allocate the correct amount of resources to each cloud service in advance. In [6] a framework to estimate the resource demand is proposed. This framework includes different prediction algorithms such as Exponential Moving Average model (EMA), Second Moving Average model (SMA), Trend Seasonality model (TS), AR model, and Fuzzy Neural Networks (FNN). In the first stage, resources demand is predicted using the EMA model, SMA model, TS model, and AR model. Finally, all predictor values are sent to an FNN model. The FNN model is optimized by clustering algorithms. The authors propose the queuing network models to predict the performance of the cloud

applications in [7]. The proposed model uses request rate and average resource requirements for requests as parameters. The queuing model being proposed is mainly used to predict these parameters. Authors in [8] propose a resource-provisioning framework for cloud services. To avoid SLA violations, the authors present a framework based on neutrosophic soft-set and fuzzy neutrosophic soft-set methods. The authors propose a framework for provisioning the VMs in [9]. In the framework being proposed, a workload analyzer estimates the request arrival rate using previous historical workload data. It also predicts the request arrival rate reject ratio and response time. If the predicted parameters lead to SLA violations, the proposed framework updates the allocated VMs to a cloud service.

The authors propose a prediction approach in [10] to predict the server workload based on the autoregression model, but this approach being proposed is only suitable for the linear structure of workloads. In [11] a prediction model using the ARIMA model is proposed. In this framework, the authors mainly focus on the scaling process of resources. To perform the scaling operations in advance, authors use the ARIMA model to predict the cloud service workload. In [12], authors propose a prediction model based on the regression technique to predict broadcast service workloads for a live sports event. The approach proposed uses a simple linear regression model that might not handle the complex irregular incoming workloads. The authors propose a workload prediction model based on the two-stage neural networks in [13]. Later, authors use the neural network regression model to predict the future utilization of resources. Finally, the VM manager performs the scaling operations based on future utilization value. In [14] a prediction model for multiple time series models is proposed. The authors use the Hidden Markov Model (HMM) as a predictor, and this model also does a clustering of similar time series to optimize the execution time. In [15] a framework based on the support vector machine and self-organizing map methods is presented. At the first stage, the self-organizing map method is used to group similar data from different regions. In the next stage, the support vector machine method is used as a predictor to estimate future data. However, this approach maintains the threshold values to determine the data points in different regions, which may lead to an inconsistent accuracy rate. In [16] authors use K-nearest neightbors method as a final predictor, to predict the time series of different workloads. The method proposed is a two-tier architecture to analyze the incoming workloads, but it needs high computational resources due to lazy learners.

Authors propose an ensemble algorithm for workload prediction in [17]. In this ensemble model, authors use five different prediction algorithms to predict each cloud service's workload. Each prediction model contributes the future values based on the assigned weights. In [18] a prediction framework is proposed to perform the scaling operations in advance. The framework proposed also focuses on to avoid SLA violations. Authors propose a prediction model using a sliding window, linear regression, and artificial neural networks in [19]. The main aim of this approach is to predict the future CPU usage of demand traces. In [20] a model is proposed, based on improved genetic algorithm to improve virtual machine selection performance while achieving the load balancing in the cloud data center. Authors propose a prediction algorithm to predict the future usage of resources based on the

backpropagation neural networks in [21]. This model proposed uses another stochastic model to improve the performance of the predictions. In [22], authors propose an enhanced evolutionary algorithm to avoid SLA violations. The framework proposed is a combination of adaptive resampling genetic algorithm and stochastic prediction model. In [23] a workload prediction framework is presented, using the steepest descent learning algorithm and artificial neural networks. The model proposed addresses the solution for the time delay in neural networks. In [24] a workload prediction model is proposed, based on a backpropagation learning algorithm with neural networks. The backpropagation algorithm helps to reduce prediction errors while adjusting the weights of the network. Authors propose a hybrid approach for effective resource provisioning in cloud environments in [25]. This hybrid resource provisioning approach is a combination of autonomic computing and reinforcement learning. The proposed approach is used to estimate the request arrival rate for each cloud service.

Machine learning algorithms are more efficient for long-term predictions, and the integration of different learning algorithms can provide significant workload predictions. The main drawback of conventional and existing solutions is the lack of the ability for long-term predictions. This ability would significantly improve the business of cloud environments. For this purpose, we present a novel Hybrid Predictive Cloud Workload Management Framework (HPCWMF) using an improved LSTM network.

3. Methodology

In this section, the associated theories and developed methodologies are presented as follows: data preprocessing method, improved LSTM network, opposition-based differential evolution algorithm to optimize the LSTM network, and dropout mechanism for recurrent connections, and multilayer perceptron network.

3.1. Data preprocessing method

The data preprocessing method is the initial step in the process of prediction. In the first stage, extract the HTTP requests from the two data sets, and aggregate them as a time interval. To conduct the various experiments, work with different time interval units such as 5, 10, 15, 20, 30, 45, and 60 minutes intervals. To normalize the data in the range of (0, 1), we used the min-max normalization equation, as shown in (1). In (1), $X_{\min}$ and $X_{\max}$ refer to the dataset's minimum value and maximum value. After the normalization process, normalized data are considered as input for the network as shown in (2). Further, the input data is divide into two data sets, such as training and testing datasets,

$$\overline{X_i} = \frac{X_i - X_{\min}}{X_{\max} - X_{\min}}, \tag{1}$$

(2)
$$\begin{bmatrix} 0.93 & 0.84 & 0.75 & 0.62 & 0.75 & \cdots \\ 0.84 & 0.75 & 0.62 & 0.75 & 0.71 & \cdots \\ 0.75 & 0.62 & 0.75 & 0.71 & 0.69 & \cdots \\ 0.62 & 0.75 & 0.71 & 0.69 & 0.68 & \cdots \\ 0.75 & 0.71 & 0.69 & 0.68 & 0.57 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots \end{bmatrix}.$$

3.2. The improved LSTM neural network

In ANNs, many networks have the capability to learn only limited patterns from independent variables such as inputs and outputs. The Recurrent Neural Network (RNN) are able to solve this issue with memory features. With this significant feature, RNN networks can elicit required patterns in data. RNN has the capability to store memory since its current output is dependent on the former computations. However, RNNs handle only a few previous timestamps due to exploding and vanishing gradient issues, which means RNN faces difficulty learning long-term dependencies. The past information exponentially disappears from memory while increasing time steps. The Long-Short Term (LSTM) network is proposed to solve the exploding and vanishing gradient issues in RNNs while handling with long-range dependency data. The LSTM network is one kind of deep RNN model which is composed of LSTM units with memory cells. The LSTM network structure is presented in Fig. 1. In LSTM network, memory Cells (C) are additional features to handle long-range dependency data. So that in LSTM network can be performed read, write, and reset data operations for memory cells. To perform these operations, LSTM has special kinds of gates such as Input gate (I_t), Forget gate (F_t), and Output gate (O_t). These three gates decide which information flows into and out of the memory cell. If the gate value equals to 0, then the signal is stopped by the gate. These computation steps are as follows:

Forget gate decides what information will be removed from the cell state, where: x_t is current input and h_{t-1} represents the previous output of cell state at time t; W_f is weight, and b_f refers to a bias of the forget gate; σ represents the sigmoid function which gives either 0 or 1 as an output. Here, if F_t equals 0, which means completely get rid of the data, otherwise completely keep the data,

(3) $$F_t = \sigma\left(W_f \cdot [x_t, h_{t-1}] + b_f\right).$$

Likewise, the input gate (I_t) decides what new information from the input will be updated or added to the cell state by using Equation (4) to compute new candidate values ($\tilde{C}_t$) for memory cell. Hyperbolic tangent function as shown in Equation (5) is used, to update new cell state (C_t) from the old cell state (C_{t-1}) "×" is used, which means point wise multiplication as shown in Equation (6):

(4) $$I_t = \sigma\left(W_i \cdot [x_t, h_{t-1}] + b_i\right),$$

(5) $$\tilde{C}_t = \tanh\left(W_C \cdot [x_t, h_{t-1}] + b_C\right),$$

(6) $$C_t = F_t \times C_{t-1} + I_t \times \tilde{C}_t.$$

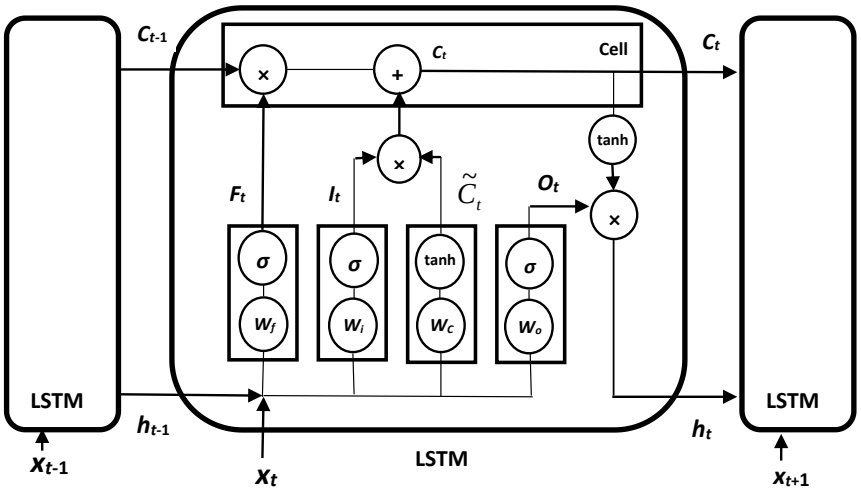

Fig. 1. The LSTM neural network structure

Output gate (O_t) controls the cell state information to flow into the outputs, as shown in Equation (7), which means O_t decides which part of the cell state information to flow into the outputs. Finally, the output of the LSTM cell (h_t) uses a hyperbolic tangent function to calculate the final output, as shown in Equation (8):

(7) $$O_t = \sigma\left(W_o \cdot [x_t, h_{t-1}] + b_o\right),$$

(8) $$h_t = O_t \times \tanh\left(C_t\right).$$

For the next time step, a new cell state (C_t) and hidden state (h_t) will be transferred to the next LSTM cell, as shown in Fig. 1. The network modifies weights and biases values by minimizing the errors between actual and predicted values.

3.2.1. OBDE based optimization of LSTM network

LSTM is a sophisticated network of RNN networks that solve the exploding and vanishing gradient issues. LSTM network contains a memory block that can deal with long and short-term time series. LSTM networks provide a clearer structure for data analysis than conventional deep learning methods. However, they need proper training. Different hyperparameters of the LSTM network can influence the prediction performance, so the selection of these hyperparameters should be based on an intelligence algorithm to improve the prediction performance. Among many parameters of the LSTM network, the number of neurons in the hidden layer(s) and batch size plays an important role. The number of neurons in the hidden layer will determine the result of data fitting. The batch size number will influence the effect of either over-fitting or under-fitting of training data. If the batch size is small, then the training data face difficulty in the convergence of data, which leads to under-fitting, and on the other side, if the batch size is large, then the size of memory will increase. For example, the number of neurons in the hidden layer is within the range of (1, 200), and the batch size range is (1, 200) then a total of 40,000 computations will be done. To deal with this issue, an evolutionary algorithm along with opposition-

based learning is adopted to select optimal parameters for the LSTM network to reduce the computational time and improve the prediction performance.

The Opposition-Based Learning (OBL) is a significant new searching technique in the optimization process, proposed by T i z h o o s h [26]. In the optimization process, the OBL technique finds the best candidate solution by comparing the present candidates and corresponding their opposition candidates. With this technique, instead of starting with one random guess solution, the OBL algorithm also simultaneously considers its opposite solution. By doing this, the closer one to solution (say guess or opposite guess) can be chosen as initial solution [27]. It also helps to reach a global optimum solution. The mathematical definition of the OBL technique is as follows:

(9) $$\breve{x} = \text{lb} + \text{ub} - x,$$

(10) $$\breve{x}_i = \text{lb}_i + \text{ub}_i - x_i.$$

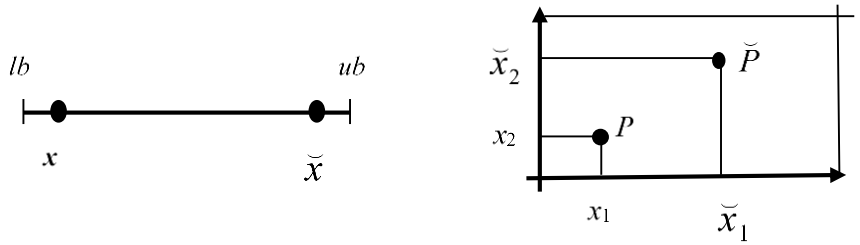

Fig. 2. Opposition theory for one and two dimensional space

Let x be a real number in [lb, ub], then its opposite is $\breve{x}$, set as in (9) where lb and ub are lower bound and upper bound respectively. Similarly, this method can be applied to evolutionary dimension space also as in (10). Let $P(x_1, x_2, x_3, \ldots, x_M)$ be a point in M dimensional space, where $x_1, x_2, x_3, \ldots, x_M$ are real numbers and $x_i \in [\text{lb}_i, \text{ub}_i]$. Then, the opposition point is $\breve{x}_i$, set as in (10). Fig. 2 represents the point and its opposite in one and two dimensional space.

In our approach proposed, two dimensional space is considered, so point P is defined as P(number of neurons in hidden layers, batch size). Instead of providing one random point to the differential evolution algorithm, the OBL technique provides its opposite point also simultaneously to reach a global optimum solution. Fig. 3 represents the process of the opposition-based initial population for the differential evolution algorithm. The point P and its opposite point $\breve{P}$, both are evaluated simultaneously using fitness function and the fittest point will be selected for the next process. For example, $f(x)$ is the objective function and $g(x)$ is the fitness function. For point $P(x_1, x_2)$ and its opposite point $\breve{P}(\breve{x}_1, \breve{x}_2)$, compare the fitness value between $g(P)$ and $g(\breve{P})$. The point which has smallest error value will be considered for the production of new candidates. The fitness value is measured and evaluated using Mean Squared Error (MSE) as shown in the next equation, where y_i and $\hat{y}_i$ are actual and predicted values at time i and N is the total number of samples:

(11) $$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 .$$

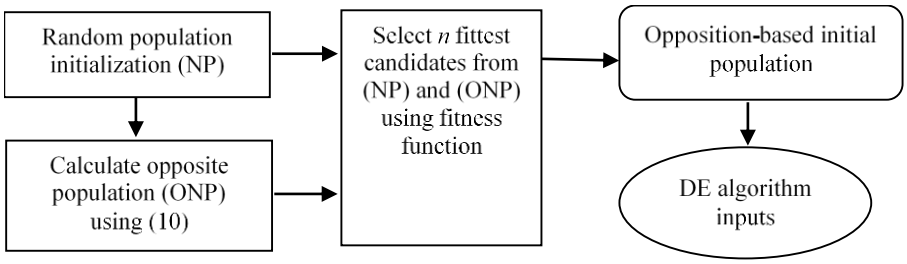

Fig. 3. Opposition-based population initialization for DE Algorithm

In Evolutionary Algorithms (EA), candidates are initially generated randomly, and later algorithms produce a new best candidate as a solution. With this principle, different metaheuristic algorithms have been developed, such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Differential Evolution (DE), etc. In this category, the DE Algorithm is a reliable and intelligent algorithm to train the neural networks. Many researchers adopted a DE algorithm to train the neural networks in different domains. Unlike other evolutionary algorithms, the DE algorithm has the capability to learn better mutation strategies and optimal crossover rates. In the approach proposed, DE algorithm is used to set the number of neurons in hidden layers and batch size of the LSTM network rather than set them by trial and error, which may lead to expensive computation process. By conducting numerous experiments, the count of the neurons considered in hidden layers is in the range of [10, 50], and batch size is in the range of [32, 120] in this paper. The process of DE optimization algorithm is elaborated as follows:

Input: OBL based iNitial Population (NP), mutation (F) and Crossover Rate (CR), and lower and upper bound of search space (ub, lb).

Output: The optimized neurons count of hidden layers and batch size for LSTM network.

Step 1. Initialization: The opposition-based initial population X_t (t=0) presenting different neurons count of hidden layer and batch size. $X_t = \{X_{1,t}, X_{2,t}, \ldots, X_{\text{NP},t}\}$ where NP (i = 1, 2, 3,…, NP) is the population size and t is the generation number. Here i-th individual in X_t represents $X_{i,t} = \{x_{i,1,t}, x_{i,2,t}, \ldots, x_{i,D,t}\}$ where D (j = 1, 2, 3,…, D) is the dimension of search space.

Step 2. Calculate the fitness value (MSE error) of each individual in X_0 using fitness function (using (11)).

Step 3. Mutation: Generate mutated individuals using (12) on X_t, mutated vectors $V_t = \{V_{1,t}, V_{2,t}, \ldots, V_{\text{NP},t}\}$; "rand/1" mutation strategy used to generate mutated vectors,

(12) $$V_{i,t} = X_{1,t} + F * (X_{2,t} - X_{3,t}),$$

where $X_{1,t}$, $X_{2,t}$ and $X_{3,t}$ are different vectors randomly selected in the range of (1, NP) and F is the mutation factor.

Step 4. Crossover: Generate the trail vector $P_{\text{trail}} = \{P_{1,t}, P_{2,t}, \ldots, P_{\text{NP},t}\}$ based on binomial crossover operation on $X_{i,t}$ and $V_{i,t}$:

$$p_{i,j,t} = \begin{cases} v_{i,j,t} & \text{if} \quad \text{rand}_j(0,1) \le \text{Cr or } j = j_{\text{rand}}, \\ x_{i,j,t} & \text{otherwise,} \end{cases} \tag{13}$$

where rand_j is the random value in the range of (0, 1) for j, Cr is the crossover rate and j_{rand} is the random value in the range of (1, D).

Step 5. Calculate the fitness value (MSE error) of each individual in P_{trail} using fitness function (11).

Step 6. Selection: Compare MSE value of $X_{i,t}$ and $P_{i,t}$, then selection operation choose the best individual for the next generation which has lowest error rate:

$$X_{i,t+1} = \begin{cases} P_{i,t} & \text{if} \quad \text{fit}(P_{i,t}) \le \text{fit}(X_{i,t}), \\ X_{i,t} & \text{otherwise.} \end{cases} \tag{14}$$

Step 7. $t = t+1$.

Step 8. If stopping criteria is satisfied then get the optimal number of neurons count of hidden layers and batch size of LSTM network, otherwise go to the Step 3.

3.2.2. Applying dropout without memory loss in LSTM network

Deep neural network structures contain a large number of parameters with multiple hidden layers so that networks can analyze a large amount of data within a short time and also to be able to learn different relationships between input data and output data. Sometimes, these computation relationships can lead to noise issues in the training period, which may lead to overfitting problem. In neural networks, overfitting is a complicated issue, and several approaches have been proposed to avoid it. One of the traditional methods is stopping the training process of the model as the model starts to get worse on a validation set. Few more approaches, such as the soft weight sharing method and L1 and L2 regularization methods, help to avoid overfitting issues. However, training architectures with different hyperparameters is challenging due to different hyperparameters for each architecture, and these architectures take more time for computation. Even large networks have the capability to train network, but at test time, networks may find it difficult to respond quickly.

The dropout method helps to avoid the issues mentioned above. The dropout technique provides a way of combining different types of neural network architectures effectively with different hyperparameters while avoiding overfitting issues. During the training of the network, dropout technique drop units (neurons) along with neuron connections. The main idea of the dropout technique is to remove the neurons and their incoming and outgoing connections duration of training period. The selection of neurons to remove is random. In other words, each neuron is retained with a probability ***p*** where ***p*** can be set as in the range of (0.1, 1). Applying dropout method can prevent co-adjustment among hidden units of deep neural networks by removing out randomly selected hidden units. In the duration of the training period, by removing the hidden units randomly, surviving unit weights are updated by backpropagation method. With the help of this process, co-adaption among hidden units is reduced so that each and every hidden unit becomes more robust. After the training period, in the testing period, fully network model weights are rescaled by

multiplying with (1 – ***p***) as a mean of different network models trained with omitted units in the training period. Authors in [28] have proven that applying the dropout technique to neural networks can achieve effective generalization capabilities with high performance.

However, using the dropout technique in LSTM networks is not a trivial and challenging task due to the LSTM network's recurrent connections. Few researchers have applied the dropout technique on both recurrent connections and non-recurrent connections, but it may lead to difficulty for LSTM networks to learn from temporal dependencies. Particularly long-term dependencies are very important for LSTM networks to update the memory cells. To solve this issue a study is presented in [29] about where the dropout technique works effectively, such as input-to-hidden connections or hidden-to-output connections. With the help of this study, we have concluded that applying the dropout technique on proper connection is more beneficial than applying on every connection. In our method a dropout technique is introduced for LSTM networks to solve the issues of memory loss while applying on recurrent connections. To achieve this, the dropout technique is applied to the candidate values vector as

$$C_t = F_t \times C_{t-1} + I_t \times d\left(\tilde{C}_t\right). \tag{15}$$

In LSTM network architecture, new candidate values ($\tilde{C}_t$) is computed for memory cell by using hyperbolic tangent function. Applying the dropout technique only on candidate values can avoid the memory loss of recurrent connections while providing short and long-term dependencies. In LSTM networks, the hidden state is considered as an input to the subnetworks to compute all gate values and memory cell updates. Applying the dropout on a hidden state may lead to memory loss of recurrent connections. To avoid this issue, in the approach proposed, a hidden state is considered a key part and applied dropout only on the candidate values vector.

3.3. Multilayer perceptron network

In Artificial Neural Networks (ANNs in general), MultiLayer Perceptron Networks (MLPNs in particular) are effective computing network models to analyze a large amount of non-linear data sets. Compared to other machine learning algorithms, MLPNs have the advantages of high accuracy results. This accuracy result comes not only with advanced features of neural network characteristics, but also can bale the data to be analyzed in parallel. MLPNs are used for both classification applications as well as regression applications. These networks include three layers, such as input layer, hidden layer(s), and output layer. In every layer, all neurons will be communicated with acyclic links. In the proposed approach, MLPN has three hidden layers.

The proposed approach has two types of network models, which give better results than a single prediction model. After performing the prediction process the network models' results are aggregated by a new combined mechanism, which is based on the LSTM network.

4. The proposed hybrid predictive framework: HPCWMF

The proposed hybrid approach is able to achieve highly accurate results in the prediction process based on the improved LSTM network and multilayer perceptron network. The proposed architecture is presented in Fig. 4.

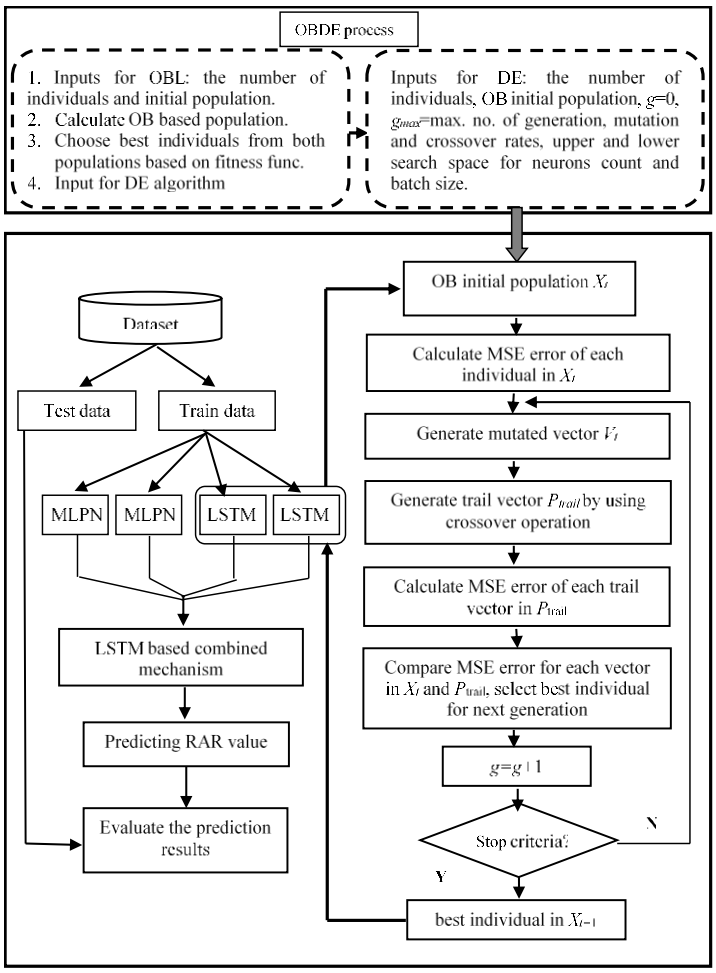

Fig. 4. A proposed hybrid predictive cloud workload management framework

By improving the traditional LSTM architecture by using opposition-based differential evolution algorithm and dropout technique on recurrent connection without memory loss, the proposed approach has the ability to perform better prediction process. In the proposed hybrid model, two LSTM networks and two multilayer perceptron networks are used. In the first stage, the opposition-based differential evolution algorithm are used to find out the optimized neurons count in hidden layers and batch size of the LSTM network. In the second stage, the dropout

technique is applied to the candidate vector of the LSTM network rather than applying on the whole recurrent connection to avoid the memory loss issue in the LSTM network. Finally, four prediction models are evaluated by a new mechanism, which is based on the LSTM network to produce the final results.

Fig. 4 represents all steps of the proposed hybrid prediction model to predict the request arrival rate for a cloud application. A quality training data is needed for the prediction model to complete the successful learning process. To achieve better results in the prediction process, we have improved the traditional LSTM network by using OBDE optimization and dropout techniques. The improved LSTM network identifies optimized parameters for the training process and significantly improves the accuracy rate of prediction, which we prove by various experiments in the next section.

5. Experimental analysis

The proposed approach has been developed and implemented in the system, which has Intel core I7 processor with 8 GB of RAM. The proposed work has been implemented in the Jupiter notebook. To evaluate the efficiency of our approach, we considered two real world workload traces that contain request arrival rate of servers. Two data sets are NASA HTTP traces [30] and Saskatchewan HTTP traces [31] used in our experiments. The NASA Data Set (DS1) contains two months of HTTP requests to the NASA Kennedy Space Center WWW server in Florida. This data set contains five attributes: hostname, timestamp, requests to the server, HTTP reply code, and bytes in the reply. These HTTP logs are considered as one line per request. The second data set Saskatchewan Data Set (DS2) contains seven months of HTTP requests to the University of Saskatchewan's WWW server in Canada. This data set also has the same attributes as the NASA data set, and logs are also considered as one line per request. We performed various experiments with different parameters for each model. Different prediction intervals were considered, such as 5, 10, 15, 20, 30, 45, and 60 min intervals for various experiments. For the experiments, 75% of the data is considered for training, and 25% of the data is considered for testing of the models.

Derived results of the proposed approach are measured by computing the Root Mean Squared Error (RMSE) and *R*-squared (R^2) of the actual workload of the server and the output of the proposed hybrid approach. The RMSE and R^2 are presented in Equations (16) and (17), respectively, where $\bar{y}_i$, y_i, and $\hat{y}_i$ are average of the predicted value, actual and predicted workload values at time i and N is the total number of samples, respectively. These measured metrics has been used in many research fields and give the means to present the efficiency of the proposed models for prediction applications. The results of the proposed hybrid model (HPCWMF) are compared to different prediction techniques such as MLPN, RNN, LSTM, and LSTM-DE. The experimental results proved that the proposed approach achieves efficient results while minimizing the error in between actual workload and predicted workload,

(16) $$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}\ ,$$

(17) $$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y}_i)^2}\ .$$

Figs 5a and b represent the actual and predicted workload for two data sets NASA and Saskatchewan, respectively, for 15 minutes interval. The presented results prove the very close relation between actual and predicted workload values. The error rate is also very low in between two workloads, as shown in Figs 7a and b. Figs 6a and b show the workload values for 60 min interval. The error rate for 60 min interval is presented in Figs 8a and b. The calculated RMSE errors and R^2 for both data sets using comparative models and proposed model are presented in Table 1 and Table 2, for different prediction intervals.

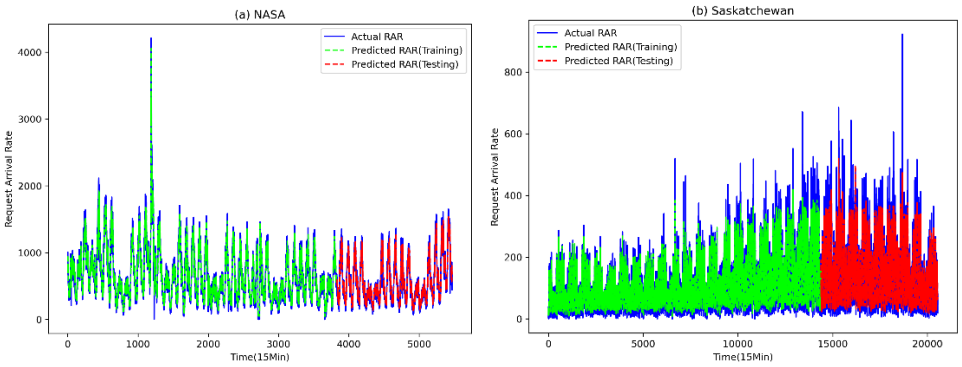

Fig. 5. Actual RAR vs predicted RAR for 15 min interval

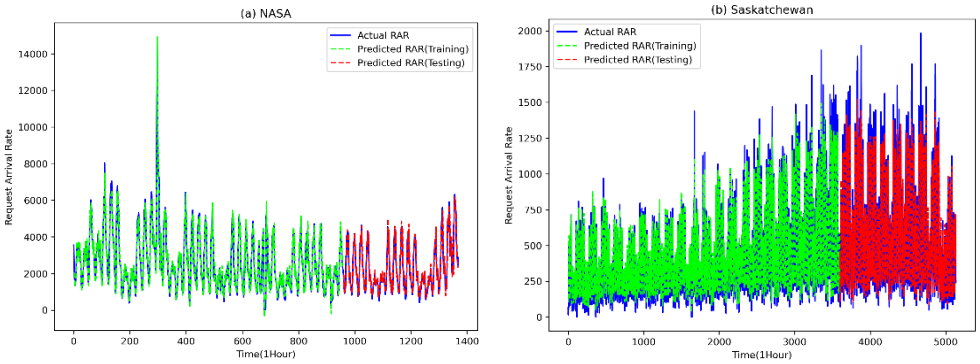

Fig. 6. Actual RAR vs predicted RAR for 60 min interval

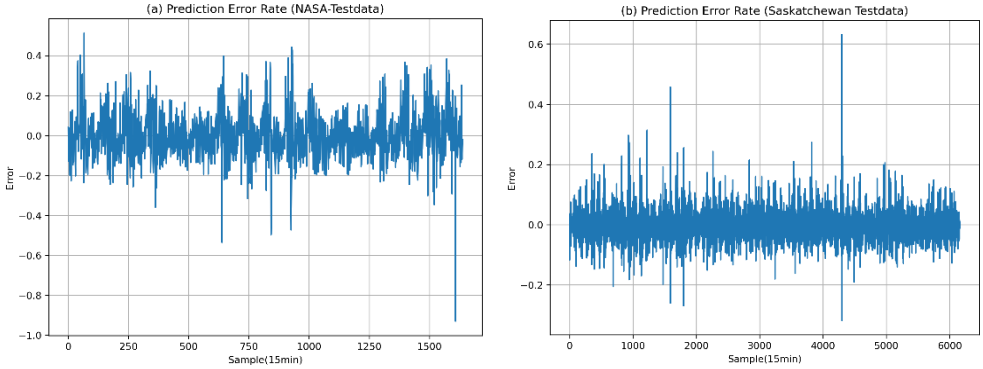

Fig. 7. Error rate for two test datasets (15 min interval)

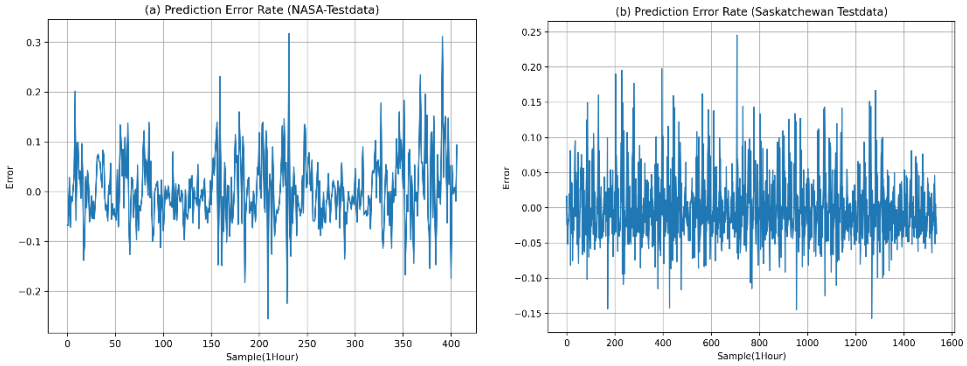

Fig. 8. Error rate for two test datasets (60 min interval)

Table 1. Comparison of various prediction model's RMSE error for two data sets

Time interval (min)	MLPN		RNN		LSTM		LSTM-DE		**HPCWMF (proposed)**	
	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2
5	1.723	1.895	0.975	1.023	0.696	0.833	0.286	0.323	0.156	0.183
10	1.508	1.629	0.962	1.009	0.651	0.809	0.202	0.309	0.132	0.179
15	1.421	1.508	0.899	0.928	0.643	0.771	0.162	0.228	0.119	0.154
20	1.019	0.925	0.875	0.924	0.575	0.778	0.155	0.204	0.085	0.113
30	0.988	0.936	0.883	0.983	0.518	0.724	0.113	0.163	0.083	0.091
45	0.909	0.927	0.791	0.887	0.601	0.705	0.091	0.127	0.061	0.088
60	0.906	0.928	0.788	0.867	0.597	0.698	0.067	0.088	0.019	0.053

Table 2. Comparison of various prediction model's R^2 value for two data sets

Time interval (min)	MLPN		RNN		LSTM		LSTM-DE		**HPCWMF (proposed)**	
	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2	DS1	DS2
5	0.523	0.486	0.589	0.491	0.684	0.613	0.818	0.803	0.865	0.832
10	0.529	0.503	0.597	0.511	0.714	0.656	0.832	0.824	0.894	0.859
15	0.566	0.533	0.623	0.523	0.758	0.696	0.874	0.846	0.918	0.881
20	0.591	0.569	0.647	0.563	0.783	0.727	0.896	0.879	0.923	0.897
30	0.639	0.591	0.685	0.574	0.801	0.763	0.909	0.895	0.944	0.919
45	0.681	0.633	0.703	0.646	0.821	0.795	0.923	0.903	0.969	0.947
60	0.702	0.661	0.718	0.693	0.834	0.811	0.931	0.917	0.988	0.979

We have performed experiments 12 times for each model and the average of the results are presented in Table 1 and Table 2. These obtained results prove that enhancement in RMSE error and R^2 for the proposed hybrid approach. They also prove that for workload prediction in the cloud management framework, the proposed hybrid approach outperforms the other four models in prediction accuracy.

6. Conclusion

The accurate workload prediction for cloud applications is an important task to perform resource scaling operations in advance while avoiding SLA violations and reducing infrastructure's computational cost. This task is not trivial for a cloud provider and challenging due to the irregular incoming workloads to the cloud server from users. To enhance the cloud resource management framework's workload prediction process, the authors propose a novel hybrid predictive approach to predict the future workload with a high accuracy rate, which is a combination of improved LSTM network and multilayer perceptron network. The proposed framework adopts an opposition-based differential evolutionary algorithm to identify the optimized neuron count in each hidden layer and batch size, which can influence the prediction ability of the LSTM network. Dropout technique is applied directly on candidate values in recurrent connections, which does not lead to long-term memory loss. The proposed approach performance is evaluated by using two real-world workload traces data sets such as NASA and Saskatchewan. The experimental results prove that the proposed hybrid approach outperforms the conventional deep learning models.

References

1. Amiri, M., L. Mohammad-Khanli. Survey on Prediction Models of Applications for Resources Provisioning in Cloud. – Journal of Network and Computer Applications, Vol. **82**, 2017, pp. 93-113.
2. Kumar, K. D., E. Umamaheswari. Prediction Methods for Effective Resource Provisioning in Cloud Computing: A Survey. – Multiagent and Grid Systems, Vol. **14**, 2018, No 3, pp. 283-305.
3. Yang, J., C. Liu, Y. Shang, B. Cheng, Z. Mao, C. Liu, L. Niu, J. Chen. A Cost-Aware Auto-Scaling Approach Using the Workload Prediction in Service Clouds. – Information Systems Frontiers, Vol. **16**, 2014, No 1, pp. 7-18.

4. T a n g, Z., Y. M o, K. L i, K. L i, Dynamic Forecast Scheduling Algorithm for Virtual Machine Placement in Cloud Computing Environment. – The Journal of Supercomputing, Vol. **70**, 2014, No 3, pp. 1279-1296.
5. G a r g, S. K., A. N. T o o s i, S. K. G o p a l a i y e n g a r, R. B u y y a. SLA-Based Virtual Machine Management for Heterogeneous Workloads in a Cloud Datacenter. – Journal of Network and Computer Applications, Vol. **45**, 2014, pp. 108-120.
6. C h e n, Z., Y. Z h u, Y. D i, S. F e n g. Self-Adaptive Prediction of Cloud Resource Demands Using Ensemble Model and Subtractive-Fuzzy Clustering Based Fuzzy Neural Network. – Computational Intelligence and Neuroscience, 2015.
7. U r g a o n k a r, B., P. S h e n o y, A. C h a n d r a, P. G o y a l, T. W o o d. Agile Dynamic Provisioning of Multi-Tier Internet Applications. – ACM Transactions on Autonomous and Adaptive Systems (TAAS), Vol. **3**, 2008, No 1, pp. 1-39.
8. B h a r g a v i, K., B. S. B a b u. Uncertainty Aware Resource Provisioning Framework for Cloud Using Expected 3-SARSA Learning Agent: NSS and FNSS Based Approach. – Cybernetics and Information Technologies, Vol. **19**, 2019, No 3, pp. 94-117.
9. C a l h e i r o s, R. N., R. R a n j a n, A. B e l o g l a z o v, C. A. D e R o s e, R. B u y y a. CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms. – Software: Practice and Experience, Vol. **41**, 2011, No 1, pp. 23-50.
10. L i, T. H. A Hierarchical Framework for Modeling and Forecasting Web Server Workload. – Journal of the American Statistical Association, Vol. **100**, 2005, No 471, pp. 748-763.
11. R o y, N., A. D u b e y, A. G o k h a l e. Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting. – In: 4th IEEE International Conference on Cloud Computing, 2011, pp. 500-507.
12. S u n, Y. S., Y. F. C h e n, M. C. C h e n. A Workload Analysis of Live Event Broadcast Service in Cloud. – Procedia Computer Science, Vol. **19**, 2013, pp. 1028-1033.
13. K u m a r, K. D., E. U m a m a h e s w a r i. EWPTNN: An Efficient Workload Prediction Model in Cloud Computing Using Two-Stage Neural Networks. – Procedia Computer Science, Vol. **165**, 2019, pp. 151-157.
14. K h a n, A., X. Y a n, S. T a o, N. A n e r o u s i s. Workload Characterization and Prediction in the Cloud: A Multiple Time Series Approach. – In: IEEE Network Operations and Management Symposium, 2012, pp. 1287-1294.
15. C a o, L. Support Vector Machines Experts for Time Series Forecasting. – Neurocomputing, Vol. **51**, 2003, pp. 321-339.
16. B a n, T., R. Z h a n g, S. P a n g, A. S a r r a f z a d e h, D. I n o u e. Referential kNN Regression for Financial Time Series Forecasting. – In: International Conference on Neural Information Processing, 2013, pp. 601-608.
17. M e s s i a s, V. R., J. C. E s t r e l l a, R. E h l e r s, M. J. S a n t a n a, R. C. S a n t a n a, S. R e i f f-M a r g a n i e c. Combining Time Series Prediction Models Using Genetic Algorithm to Autoscaling Web Applications Hosted in the Cloud Infrastructure. – Neural Computing and Applications, Vol. **27**, 2016, No 8, pp. 2383-2406.
18. K u m a r, K. D., E. U m a m a h e s w a r i. Efficient Cloud Resource Scaling Based on Prediction Approaches. – International Journal of Engineering & Technology (UAE), Vol. **7**, 2018, No 4.10, pp. 413-416.
19. I s l a m, S., J. K e u n g, K. L e e, A. L i u. Empirical Prediction Models for Adaptive Resource Provisioning in the Cloud. – Future Generation Computer Systems, Vol. **28**, 2012, No 1, pp. 155-162.
20. N a i k, K. B., G. M. G a n d h i, S. H. P a t i l. Pareto Based Virtual Machine Selection with Load Balancing in Cloud Data Centre. – Cybernetics and Information Technologies, Vol. **18**, 2018, No 3, pp. 23-36.
21. P r e v o s t, J. J., K. N a g o t h u, B. K e l l e y, M. J a m s h i d i. Prediction of Cloud Data Center Networks Loads Using Stochastic and Neural Models. – In: 6th International Conference on System of Systems Engineering, 2011, pp. 276-281.
22. G o v a r d h a n, P., P. S r i n i v a s a n. Enhanced Evolutionary Computing Assisted Robust SLA-Centric Load Balancing System for Mega Cloud Data Centers. – Cybernetics and Information Technologies, Vol. **19**, 2019, No 3, pp. 74-93.

23. C h a n g, Y. C., R. S. C h a n g, F. W. C h u a n g. A Predictive Method for Workload Forecasting in the Cloud Environment. – In: Advanced Technologies, Embedded and Multimedia for Human-Centric Computing, 2014, pp. 577-585.
24. L u, Y., J. P a n n e e r s e l v a m, L. L i u, Y. W u. RVLBPNN: A Workload Forecasting Model for Smart Cloud Computing. – In: Scientific Programming, toward Smart World. Vol. **2016**. 2016.
25. G h o b a e i-A r a n i, M., S. J a b b e h d a r i, M. A. P o u r m i n a. An Autonomic Resource Provisioning Approach for Service-Based Cloud Applications: A Hybrid Approach. – Future Generation Computer Systems, Vol. **78**, 2018, pp. 191-210.
26. T i z h o o s h, H. R. Opposition-Based Learning: A New Scheme for Machine Intelligence. – In: International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, Vol. **1**, 2005, pp. 695-701.
27. R a h n a m a y a n, S., H. R. T i z h o o s h, M. M. S a l a m a. Opposition-Based Differential Evolution Algorithms. – In: IEEE International Conference on Evolutionary Computation, 2006, pp. 2010-2017.
28. H i n t o n, G. E., N. S r i v a s t a v a, A. K r i z h e v s k y, I. S u t s k e v e r, R. R. S a l a k h u t d i n o v. Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors. – arXiv preprint arXiv:1207.0580, 2012.
29. B l u c h e, T., C. K e r m o r v a n t, J. L o u r a d o u r. Where to Apply Dropout in Recurrent Neural Networks for Handwriting Recognition? – In: 13th International Conference on Document Analysis and Recognition, 2015, pp. 681-685.
30. NASA http Two Months of http Logs from the NASA www Server. **ftp://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html**
31. Saskatchewan-http- Seven Months of http Logs from the Saskatchewan www Server. **ftp://ita.ee.lbl.gov/html/contrib/Sask-HTTP.html**

Received: 28.09.2020; Second Version: 18.10.2020; Accepted: 29.10.2020 (fast track)