Engineering, 2021, 13, 135-157
https://www.scirp.org/journal/eng
ISSN Online: 1947-394X
ISSN Print: 1947-3931

Fractional Rider Deep Long Short Term Memory Network for Workload Prediction-Based Distributed Resource Allocation Using Spark in Cloud Gaming

Koné Kigninman Désiré[1,2], Kouassi Adlès Francis[1], Konan Hyacinthe Kouassi[1], Eya Dhib[3], Nabil Tabbane[3], Olivier Asseu[1,2*]

[1]Ecole Supérieure Africaine des TIC, LASTIC, Abidjan, Côte d'Ivoire
[2]Institut National Polytechnique Felix H. Boigny (INPHB), Yamoussoukro, Côte d'Ivoire
[3]MEDIATRON Laboratory, Higher School of Communication of Tunis, Tunis, Tunisia
Email: *oasseu@yahoo.fr

How to cite this paper: Désiré, K.K., Francis, K.A., Kouassi, K.H., Dhib, E., Tabbane, N. and Asseu, O. (2021) Fractional Rider Deep Long Short Term Memory Network for Workload Prediction-Based Distributed Resource Allocation Using Spark in Cloud Gaming. *Engineering*, **13**, 135-157.
https://doi.org/10.4236/eng.2021.133011

Received: January 26, 2021
Accepted: March 15, 2021
Published: March 18, 2021

Abstract

The modern development in cloud technologies has turned the idea of cloud gaming into sensible behaviour. The cloud gaming provides an interactive gaming application, which remotely processed in a cloud system, and it streamed the scenes as video series to play through network. Therefore, cloud gaming is a capable approach, which quickly increases the cloud computing platform. Obtaining enhanced user experience in cloud gaming structure is not insignificant task because user anticipates less response delay and high quality videos. To achieve this, cloud providers need to be able to accurately predict irregular player workloads in order to schedule the necessary resources. In this paper, an effective technique, named as Fractional Rider Deep Long Short Term Memory (LSTM) network is developed for workload prediction in cloud gaming. The workload of each resource is computed based on developed Fractional Rider Deep LSTM network. Moreover, resource allocation is performed by fractional Rider-based Harmony Search Algorithm (Rider-based HSA). This Fractional Rider-based HSA is developed by combining Fractional calculus (FC), Rider optimization algorithm (ROA) and Harmony search algorithm (HSA). Moreover, the developed Fractional Rider Deep LSTM is developed by integrating FC and Rider Deep LSTM. In addition, the multi-objective parameters, namely gaming experience loss QE, Mean Opinion Score (MOS), Fairness, energy, network parameters, and predictive load are considered for efficient resource allocation and workload prediction. Additionally, the developed workload prediction model achieved better performance using various parameters, like fairness, MOS, QE, energy and delay.

Hence, the developed Fractional Rider Deep LSTM model showed enhanced results with maximum fairness, MOS, QE of 0.999, 0.921, 0.999 and less energy and delay of 0.322 and 0.456.

Keywords

Cloud Computing, Rider Deep LSTM, Fractional Calculus, Workload Prediction, Resource Allocation

1. Introduction

Cloud computing is a developing computing architecture, and it provides various computing resources as general utilities for end user through Internet. Cloud computing allows on-demand access to a shared set of resources, such as services, servers, storage space and networks [1]. However, today, cloud technology is expanding its services, called Everything-as-a-Service (XaaS). Cloud gaming enables a game content on non-specialized devices, like mobile phones, tablets, smart televisions and so on. The cloud gaming provides on-demand manner interactive gaming application, which is remotely processed in cloud and pictures are streamed as video series to play by Internet [2] [3]. The entire processing operations associated with game scene frames are performed on server Virtual Machines (VMs) in cloud gaming.

Virtualisation is the main feature of cloud computing technology, allowing the physical data center to be distributed as dynamic virtual resources. The process of resource allocation is an important part of the cloud data center. This process can save energy consumption, reduce computing costs, and enhance resource utilization efficiency. In resource allocation, the game is performed at cloud server or client side based on the present resources in network and client. Besides, the cloud computing system handles the games computational operation using cognitive capacities. The massive development of cloud computing is resource allocation, which reduces the operating price and improves the resource consumption. Generally, virtualization approach attains the flexibility, and it includes hardware virtualization, such as Central processing Unit (CPU), storage, network and memory [4] [5]. Normally, response delay includes playout delay, processing delay and network delay. Typically, playout delay is considered insignificant, and it does not have an important factor in player's game experience [6] [7]. A huge amount of games are delivered by service providers and simulated in numerous instances at various data-centers in cloud, and it is geographically distributed for reducing network delay. Likewise, processing delay relies on accessible processing power in cloud server. Processing power is identified by server resources, like CPU, Graphics processing Unit (GPU), storage and workload of game sessions, which run on cloud server. Meanwhile, virtualization system is employed for decreasing cost of cloud service providers [8] [9].

Dynamic allocation of resources can be done in two ways, such as a reactive approach and a proactive approach [10]. As part of a reactive approach, cloud users can set thresholds for resource underutilization and overuse. When the workload reaches the threshold value, the automatic resizing process takes the action based on the current state of resources, such as removing virtual machines from cloud services for an underutilized state or adding from virtual machines to cloud services for a state of overuse. The main disadvantage of this process is that the automatic resizing process has difficulty performing the resizing operations in the event of sudden fluctuations in the workload. A proactive approach allows resizing operations to be carried out in advance [11]. Cloud resource management forecasts the future workload of each cloud service and allocates the resources to their cloud services based on the expected value.

Currently, there are many techniques and methods applied to the prediction of the workload of computer systems, such as the e-learning ensemble approach [12], ARIMAR (Auto Regressive and Moving Average) models [13], Recurrent Neural Networks (RNN) [14], Long and Short Memory Networks (LSTM) [15]. However, deciding on the exact amount of resources with proactive approaches during the execution time of cloud services is a difficult and not insignificant task. Due to irregular access, cloud services are subject to fluctuations in workload. This can lead to over- or under-utilization of resources. In a state of over-provisioning, more resources will be allocated to applications in the cloud than necessary. According to service level agreements (SLAs), this is a benefit for cloud users, but for providers, it is an unnecessary cost that results in high energy consumption. In the state of under-provisioning, fewer resources will be allocated to cloud applications than are needed, leading to SLA violations, lower QoE, and ultimately loss of consumers and revenue. An effective and proactive approach must therefore accurately predict the future resources needed to achieve QoE. The most important measure of system workload prediction models is accuracy, which is measured by the difference between predicted and actual results [16]. In general, the closer the predicted value is to the actual value, the better the model.

In recent days, Artificial Intelligence (AI) technologies are introduced for computing amount of resources consumed through clients. In addition, the techniques, termed as GAugur, which identifies whether a co-located game assembles quality of service needs with pre-determined error rate [2]. Specially, Evolutionary computation (EC) approaches, such as Genetic Techniques (GA) is devised to enhance resource utilization and reduce energy consumption. The customized GA with fuzzy multi objective computation is developed for VM in [17]. Meta-heuristic-based approaches are also introduced by offering near optimal solution in sufficient period using particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), GA etc. [18] [19]. Moreover, meta-heuristic method, termed Grey Wolf optimizer (GWO) is developed, which is inspired by grey wolves [20]. However, integration performance of cloud computing is mainly based on matching rank between system representation and meta-heuristic model [21].

In order to deal with the issues mentioned above, this paper proposes the development of a new method for workload prediction method by developed algorithm using spark architecture in cloud gaming.

Motivation

The cloud gaming provides interactive gaming application, which is remotely processed in cloud and pictures are streamed as video series to play by internet. The main challenge is the interactive and real time behaviour of multiplayer cloud gaming produces a response delay in end user quality of experience [7] [22] [23]. The adaptive optimization system for cognitive resource allocation is challenging one, due to load of requested resources and real time service in gaming [7]. Also, the energy consumption is an important challenge in the cloud gaming. These challenges in the existing workload prediction and resource allocation are considered as a motivation, a new method named Fractional Rider-based HSA is developed in this work. In the proposed method, to solve these problems, the workload prediction and resource allocation are carried out using the developed Fractional Rider Deep LSTM. Also, the multi-objective parameters, such as energy, gaming experience loss, fairness, MOS, network parameters, and predictive load are considered for computing the optimal solution in cloud gaming.

The major purpose of this research is to developed a new method for workload prediction method by developed algorithm using spark architecture in cloud gaming. In this research, workload of each resource is computed by developed Fractional Rider Deep LSTM architecture. The developed Fractional Rider Deep LSTM system is the integration of FC [24] and Rider Deep LSTM [25]. Here, resource allocation process is performed based on Fractional Rider-based HSA, which is the combination of HSA [26] and ROA [27] and fractional calculus. Furthermore, fairness, MOS, network parameters, gaming experience loss, energy and predictive error is considered in multi-objective method for efficient resource allocation.

The major contribution of this research is enlisted below:

- Developed Fractional Rider Deep LSTM for effective workload prediction: An efficient technique is devised using Fractional Rider Deep LSTM for workload prediction in cloud gaming. The developed Fractional Rider Deep LSTM is the combination of FC and Deep Rider LSTM network. Moreover, multi-objective parameters, namely energy, gaming experience loss, fairness, MOS, network parameters, and predictive load are considered for computing the optimal solution in cloud gaming.

The remaining parts of the paper are listed as follows: Section 2 illustrates the existing approaches of resource allocation and workload prediction in cloud gaming and Section 3 presents the system model of cloud computing system. Section 4 explains the developed Fractional Rider Deep LSTM for workload prediction and resource allocation system. The results of developed Fractional Rider Deep LSTM are portrayed in Section 5 and Section 6 concludes the paper.

2. Literature Survey

In this section, existing workload prediction and resource allocation are explained with their disadvantages. Yiwen Han *et al.* [9] developed a distributed technique for optimizing VM position in mobile cloud gaming. Here, mobile cloud gaming system was employed with resource optimization, and NP-hard is used for identifying optimal solutions to overcome this optimization problem. Moreover, potential game theory was introduced for determining the Nash Equilibrium in multi-player competition game. This technique obtained enhanced performance than other scales and policies with increasing number of players. However, this approach did not consider the multidimensional parameters in cloud gaming for better optimization. To overcome this problem, the multi-objective method is estimated to find optimal solution in the proposed method. Hossein Ebrahimi Dinaki *et al.* [7] introduced two effective techniques for graphics processing unit-based server selection in cloud gaming. This approach is a developed version of PSO and GA, named boosted PSO and boosted GA. Additionally, service providers and players experience profits were considered for enhancing the quality of experience. This approach obtained better effectiveness in terms of player's quality of energy, and capacity wastage. Even though, this model not considered extra network parameters, and quality metrics for obtaining better inclusive solution. This problem is overcome by using several parameters, such as network definition factor, energy, gaming experience loss, fairness, predictive load, load and MOS in the proposed method. Damian Fernández-Cerero *et al.* [28] devised a GAME-SCORE simulation model in cloud gaming platform. This developed model performed various scheduling method based on Stackelberg game. In this system, two major players, named as energy efficient agent and scheduler were included to analyse the effectiveness. This model achieved light balance among make span and less energy consumption, but this method not explored extra sophisticated energy rules for better performance. In the proposed method, the total energy relating to the execution of application includes energy dissipation on both mobile server and device. Seyed Javad Seyed Aboutorabi and Mohammad Hossein Rezvani [2] modelled Bees technique to tackle players frame rate allocation problem in cloud gaming. This approach effectively enhanced cloud providers and reduced server side expenditure. Moreover, this technique is robustness in terms of frame quality, run time, bandwidth loss, and acceptance ratio, although processing power of this method was the main challenge. In the proposed method, the processing power is identified by server resources, like CPU, GPU and RAM.

Mohammad Sadegh Aslanpour *et al.* [29] presented learning automata-based resource provisioning technique for extremely multiplayer online games in cloud system. Here, an autonomic system was introduced for dynamic prerequisite of VM in cloud-based gaming system. Moreover, Auto Regressive Integrated Moving Average (ARIMA) prediction technique was employed with workload fluctuations for obtaining enhanced prediction accuracy. Furthermore, learning automata-based technique was employed as decision maker for identifying the suitable auto-scaling decision in planning stage. This system easily reduces the response time and cost, even though this method failed to examine the effect of resource auto-scaling and optimization. To overcome this problem, the developed Fractional Rider Deep LSTM predicts the load based on distributed resource allocation in cloud gaming by spark architecture. Yusen Li *et al.* [30] developed machine learning-based performance technique for resource allocation in cloud gaming. In this method machine learning approach was devised for capturing difficult relationship between performance interference. In addition, efficient techniques were devised for resource allocation circumstances in cloud gaming. This approach enhanced the resource utilization, even though the performance of this algorithm is not analysed through more server for better performance. In the proposed method, the total energy relating to the execution of application includes energy dissipation on both mobile server and device. Mostafa Ghobaei-Arani *et al.* [31] introduced autonomous resource provisioning approach for multiplayer online games in cloud structure. At first, load prediction service predicts game entity distribution based on Adaptive Neuro Fuzzy Interference System (ANFIS) from historical trace data. Moreover, fuzzy decision tree technique was employed for estimating appropriate amount of resources based on predicted workload and user Service Level Agreements (SLA). Anyhow, this technique extremely increases the delay. In the proposed method, delay is one of the parameters used for the fitness calculation for optimal solution. Anand Bhojan *et al.* [32] devised new software architecture, termed CloudyGame in cloud game system. Besides, accepted game engine was considered on resource usage in game cloud. After that, synergy and dynamic asset streaming among shared game instance and asset streaming were combined for high resolution. This model achieved high resolution and minimum content set, but this method failed to reduce the computational complexity. In the proposed method, the FC is used to reduce the overall computational time.

3. System Model

Cloud gaming allows the user with short processing capacity to play qualitative games using high quality link connection. The games can be played without installing or downloading other game software's. Moreover, the game service provider utilized a distributed data center for presenting their services to users. The user request is transmitted to particular storage space and VM is allocated to execute every user requests after receiving a request by cloud gaming architecture. Thus, VM utilizes a streaming encoded game to a user. In addition, a cloud model allocated the resources to user task for specific period such that the tasks get finished before the deadline. The resource allocator provides synchronization between cloud service provider and user. Furthermore, VM resource utilizes various configurations with storage, memory and power. The minute degradation creates the cloud infrastructure ineffective, because of resource allocation element poses a total control of cloud functions. Thus, resource allocation representation is more important for cloud gaming infrastructure. The tasks are processed manually and the system senses an overloaded condition, while VM load position is in common circumstances. Additionally, the resource allocation is developed for allocating tasks from overloaded VMs to under-loaded VMs. The network demonstration for allocating a resource in cloud is displayed in Figure 1.

4. Proposed Method for Workload Prediction

This section presents the developed Fractional Rider Deep LSTM for workload prediction in cloud gaming. Here, the developed Fractional Rider Deep LSTM is developed for work load prediction based distributed resource allocation in cloud gaming based on spark architecture. In this method, the resource allocation is performed using Fractional Rider HSA model. Moreover, the work load of each resource allocation is predicted by developed Fractional Rider Deep LSTM. The developed Fractional Rider Deep LSTM is developed by combining FC model and Rider Deep LSTM network. Moreover, gaming experience loss, MOS, fairness, energy and network parameters, predictive load is included into multi-objective model for effective resource allocation. The schematic diagram of developed Fractional Rider Deep LSTM for workload prediction is portrayed in Figure 2.

The major intention of the developed approach is to find the optimal resources in workload prediction and resource allocation to all games demanded by user. Let us assume a cloud structure with g^{th} PMs, and it is expressed as $F = \{F_1, F_2, \cdots, F_r, \cdots, F_g\}; 1 \le r \le g$, and all PM includes dissimilar VMs. Let us consider VMs available in r^{th} PM, which is represented by,

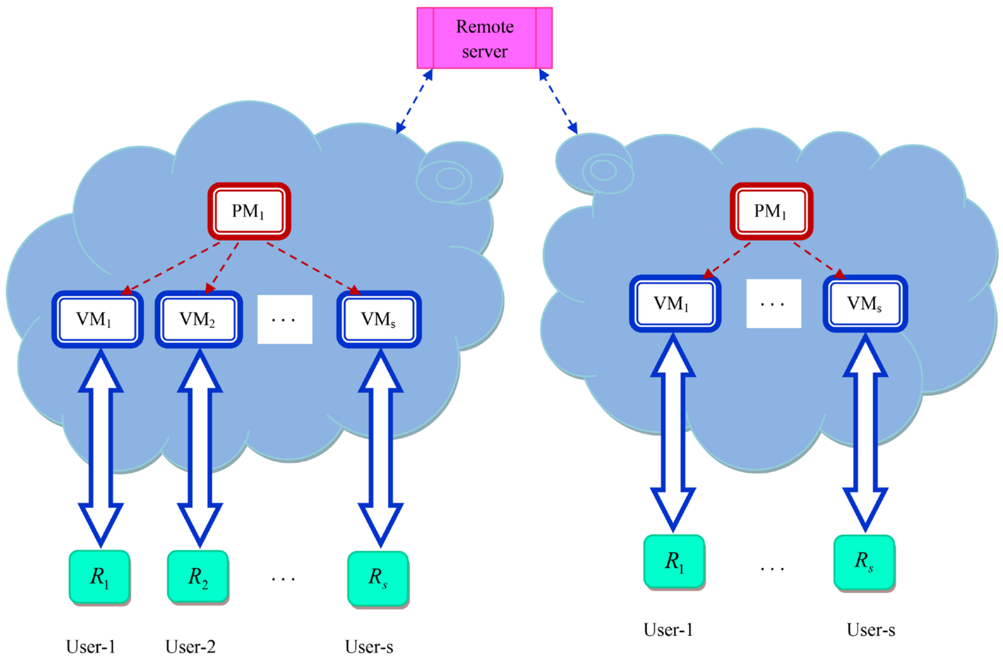


Figure 1. Network model of cloud computing.

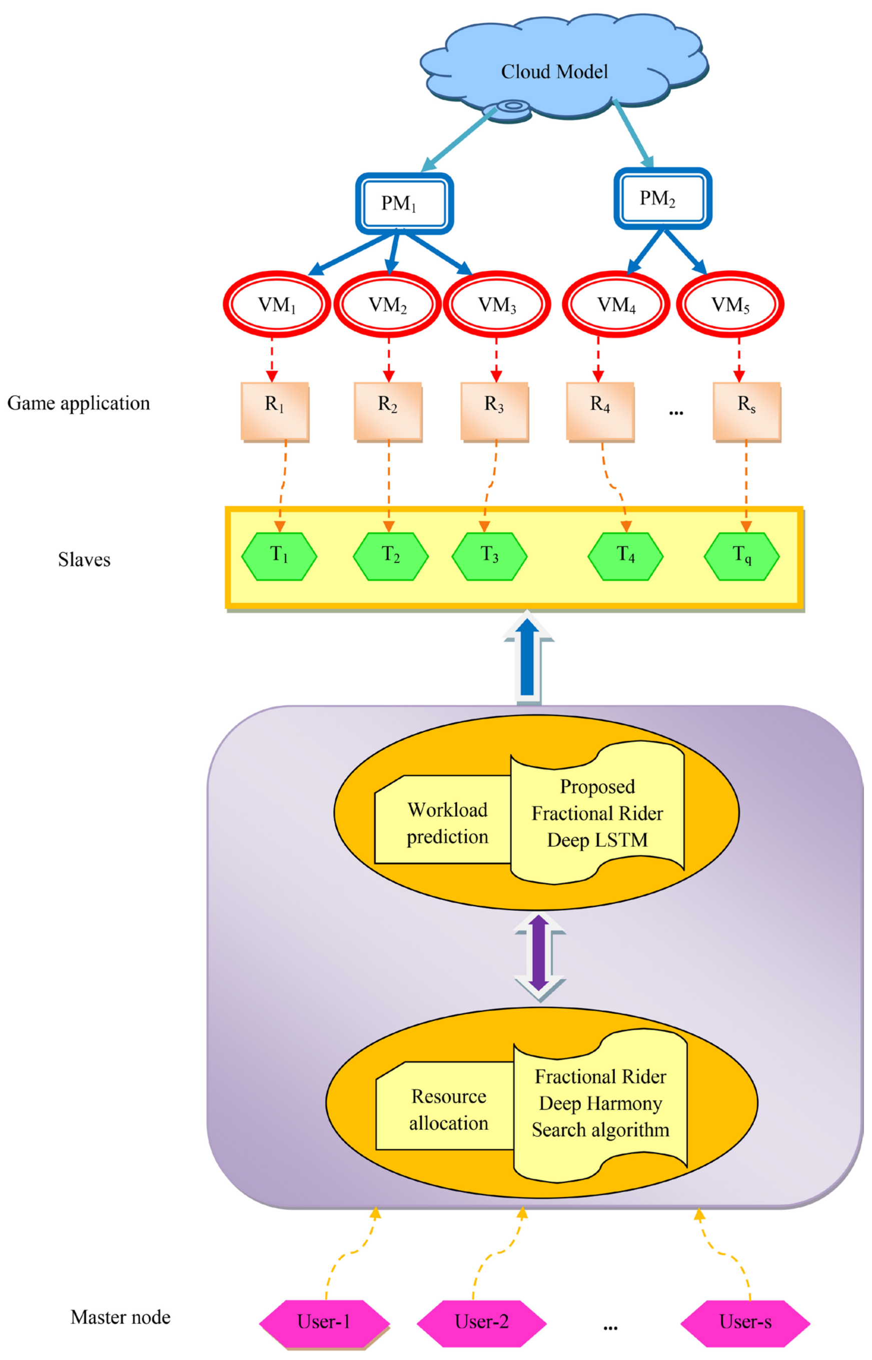


Figure 2. Block diagram of proposed Fractional Rider deep LSTM for workload prediction.

$P = \{P_1, P_2, \cdots, P_i, \cdots, P_j\}; 1 \le i \le j$, in which j denotes the total VMs in r^{th} PM. The requested game by each user is allocated to VM in round robin fashion, which is specified as, $R = \{R_1, R_2, \cdots, R_e, \cdots, R_f\}$. Here, f signified the distributed total games to VM. The user has capability to play games, and it is represented as, $D = \{D_1, D_2, \cdots, D_e, \cdots, D_f\}$. The video resolution is essential measure, when moderating cloud gaming quality. Moreover, the bit rate of each frame is indicated as, in cloud gaming, 3-dimensional video game is provided in distant data centers, and game screen depicted to users in real time by internet connectivity. $E=\{E_1,E_2,\cdots,E_e,\cdots,E_f\}$. The video frame rate for each game is depicted as $K=\{K_1,K_2,\cdots,K_e,\cdots,K_f\}$. The QoE of played game through uses history are illustrated as $Y^h=\{Y_1^h,Y_2^h,\cdots,Y_w^h,\cdots,Y_x^h\}$.

Every selected VM to distribute the resource is configured by various parameters, such as memory, bandwidth, Million Instructions Per Second (MIPS) and processors, and it is formulated by,

$$P_{r,i}=\{B_{r,i},Y_{r,i},Z_{r,i},H_{r,i},T_{r,i}\} \tag{1}$$

where, $B_{r,i}$ represent the processors of i^{th} VM in r^{th} PM, $H_{r,i}$ denotes bandwidth in i^{th} VM in r^{th} PM, $Y_{r,i}$ is memory of i^{th} VM in r^{th} PM, $Z_{r,i}$ signifies the MIPS of i^{th} VM in r^{th} PM and $K_{r,i}$ is frequency scaling of i^{th} VM in r^{th} PM. In addition, the parameters, like $B_{r,i},Y_{r,i},Z_{r,i},H_{r,i},T_{r,i}$ has the value ranging from 1 to constant value o.

Furthermore, resource units of each game are illustrated as,

$$T=\{T_1,T_2,\cdots,T_e,\cdots,T_f\} \tag{2}$$

where, user preference level lies among 0 and 1, where 1 indicates more preference and 0 denotes not preference.

4.1. Multi Objective Model

The multi-objective method is estimated to find optimal solution using solution set. The multi objective function produced for developed Fractional Rider Deep LSTM is using several parameters, like network definition factor, energy, gaming experience loss, fairness, predictive load, load and *MOS*, and it is formulated as,

$$G=L+L_p+L_m+MOS+E_f+N+(1-E_y) \tag{3}$$

where, E_y denotes energy, N is the network definition factor, MOS indicates mean opinion score, L_m specifies gaming experience loss, E_f refers fairness, L signifies load and L_p is a predictive load. The load of VM equation is represented as,

$$L_{r,i}=\sum_{e=1}^{n}\frac{(V_p+V_x+V_a+V_g+V_f)*A}{\max(V_p+V_x+V_a+V_g+V_f)*n}*\frac{1}{N} \tag{4}$$

where, n specified the number of games, N is normalizing factor, V_p denotes number of processing elements in i^{th} VM, V_x indicates the memory units in i^{th} VM, V_a is a bandwidth component in i^{th} VM, V_g represents the MIPS element in i^{th} VM and V_f specifies the frequency component in i^{th} VM.

$$A=\begin{cases}1; & \text{if } e^{th} \text{ game is run by } i^{th} \text{ VM}\\ 0; & \text{otherwise}\end{cases} \tag{5}$$

The load of PM equation is formulated by,

$$L_r=\sum_{i=1}^{f}L_{r,i} \tag{6}$$

The *MOS* equation [33] equation is expressed as,

$$MOS = \frac{1}{j}\sum_{i=1}^{j}\left(E_i * B_i + K_i * B_i\right) \tag{7}$$

where, E_i denotes the game bit rate run in i^{th} VM, K_i indicates video frame rate of game running in i^{th} VM and B_i is a resource parameters. Here, the resource parameter is because of QoS is formulated by,

$$B_i = \frac{1}{5}\left(\frac{V_p + V_x + V_a + V_g + V_f}{\max\left(V_p, V_x, V_a, V_g, V_f\right)}\right) \tag{8}$$

By integrating Frame Per Second and resolution, experience of gaming E [34] of the player is formulated, and it is the target of each player, which is expressed as,

$$E = \alpha_1 D - \alpha_2 Q - \alpha_3 P \tag{9}$$

where, D denotes delay, Q signifies experienced Frames Per Second (FPS), P indicates gaming video quality and $\alpha_1, \alpha_2, \alpha_3$ represents constant parameters.

Clone delay between user e may present, because VM with games is created and ruined dynamically, which indicates that the delay in initializing service. The writing speed in hard disk is represented as K_w through storing games in repository. If a player selects a game using file size t_i, then the delay [34] is formulated using initialization period of H^{th} VM,

$$D = \sum\nolimits_{i=1}^{j} \frac{t_i}{K_w} + H_i \tag{10}$$

where, t_i indicates file size of game in i^{th} VM and H_i is an initialization period of VM.

Moreover, Frame Per Seconde (FPS) practiced by gaming users is a key experience measure. The users follow a gaming with key experience metric, like FPS for dealing with cloud environment. FPS is exposed with Random Access Memory (RAM) GPU and CPU considers the physical server. In cloud gaming, FPS [34] is formulated by,

$$Q = \sum_{i=1}^{j} \frac{\sigma_1}{1 + e^{\sigma_2}\left(\frac{1}{5}\sum_{i=1}^{j}\beta_i + B_i\right) + \sigma_3 B_i} \tag{11}$$

where, σ_1, σ_2 and σ_3 is a approximation parameter. Additionally, game video quality [34] is expressed as,

$$p = \sum_{i=1}^{j}\left[\frac{\sigma}{d}\log_2\left(1 + \frac{cP_i}{\sigma_0 + \sum_{i=1}^{j} cP_i}\right)\right] \tag{12}$$

where, σ, d, c and σ_0 is constant value and P_i represents the video resolution of game in i^{th} VM.

The fairness is formulated as,

$$E_f = \frac{1}{k \times l} \sum_{i=1}^{k} \left(\sum_{j=1}^{l} Y^h \right) * Z \tag{13}$$

where, Z is a user preference level. The network definition factor is expressed as,

$$N = \frac{1}{2} \left[L + (1 - O) \right] \tag{14}$$

where, L is a bandwidth and O indicates delay. The bandwidth is estimated by,

$$L = \frac{1}{a \times b} \sum_{r=1}^{g} \sum_{i=1}^{j} y_{ri} \tag{15}$$

The delay parameter is calculated by,

$$O = \alpha_1 * D \tag{16}$$

where, α_1 defines the constant parameter.

The total energy [35] is expressed as,

$$W = \sum_{y=1}^{f} W_{tot}(y) + \sum_{z=1}^{f} W_{static}(z) \tag{17}$$

The total energy relating to execution of I_y application includes energy dissipation on both mobile server and device, which is illustrated as,

$$W_{tot}(y) = W_{loc,y} + W_{server,z}^{y} \tag{18}$$

where, z indicates to server index in which application is mapped. Let us assume the server follows a time out management of power policies such that, low power mode is maintained after the particular time of idle. The static energy consumption of y^{th} server, while it is idle formulated as,

$$W_{ststic}(z) = X_{ststic.z} * X \tag{19}$$

where, X be the time out threshold and $X_{ststic,z}$ indicates static power server z in idle time.

4.2. Proposed Fractional Rider Deep LSTM for Workload Prediction

This section explains about the proposed Fractional Rider Deep LSTM model for workload prediction. The FC [24] is utilized for solving integral equation and derivative equation issues. The differential equations and fractional order integral are resolved through Laplace transforms. The fractional calculus procedure is explained as follows: initial step identifies Laplace transform of equation. The second stage is utilized to solve transform of undefined function, and the last step uses inverse Laplace transform to identify the best solution. Here, fractional calculus is used to accelerate computational performance of developed model. This approach needs minimum amount of period to process, thus computational period is reduced. On the other hand, Rider Deep LSTM [25] has four layers, namely input layer, one LSTM layer, fully connected layer and regression output layer. In LSTM, each neuron of hidden layer is, termed as memory cell, which involves self connected recurrent edge. The Rider Deep LSTM enhances the real time fault prediction. Therefore, FC and Rider Deep LSTM are integrated for better performance.

Based on Rider Deep LSTM, the update rule of overtaker is expressed as,

$$\phi_d^{rider} = \phi_{d-1} + \left[S_{d-1}^G * \phi^V \right] \tag{20}$$

$$\phi_d^{rider} - \phi_{d-1} = S_{d-1}^G * \phi^V \tag{21}$$

From FC,

$$S^\lambda \left[\phi_d^{rider} \right] = S_{d-1}^G * \phi^V \tag{22}$$

Here, $d+1=d$,

$$\phi_d^{rider} - \lambda \phi_{d-1}^{rider} - \frac{1}{2}\lambda \phi_{d-2}^{rider} - \frac{1}{6}(1-\lambda)\phi_{d-3}^{rider} - \frac{1}{24}(1-\lambda)(2-\lambda)\phi_{t-4}^{rider} = S_{d-1}^G * \phi^V \tag{23}$$

$$\phi_d^{rider} = \lambda \phi_{d-1}^{rider} + \frac{1}{2}\lambda \phi_{d-2}^{rider} + \frac{1}{6}(1-\lambda)\phi_{d-3}^{rider} + \frac{1}{24}\lambda(1-\lambda)(2-\lambda)\phi_{d-4}^{rider} + \left(S_{d-1}^G * \phi^V \right) \tag{24}$$

where, λ random number, which ranges from $[0,1]$, S_{d-1}^G is a directional indicator of ϕ_{d-1} and ϕ^V indicates the leading rider.

4.3. Spark-Based Work Load Prediction and Resource Allocation Using Fractional Rider-Harmony Search Algorithm

This section presents the Fractional Rider-HSA for allocating the resources. The Fractional Rider-HSA is the combination of ROA [27], FC [24], and HSA [26]. The combination of, ROA, FC and HSA adjust the correlated parameters for obtaining global optimal solution.

Solution Encoding

Solution encoding is an expression of solutions to identify optimal solution and control the optimization problems. This solution encoding is portrayed in Figure 3.

The steps of Fractional Rider-HSA are explained below:

***Step*-1: *Initialization*:** The initialization of Fractional Rider HSA technique is given below,

$$C_p = \left\{ C_p(u,v) \right\}; 1 \le u \le I, 1 \le v \le J \tag{25}$$

where, I indicates total riders, $C_p(u,v)$ denotes the position of u^{th} rider in v^{th} dimension and J is a total dimension.

***Step*-2: *Fitness function identification*:** The fitness function estimation is very essential for obtaining the best solution. Therefore, fitness function for every solution is described in multi-objective model. The optimal solution is determined at final iteration as each solution for obtaining best solution. Moreover,

VM_1	VM_2	…	VM_s
R_1	R_2	…	R_s

Figure 3. Solution encoding.

the fitness function is computed based on Equation (3) in multi-objective model section.

Step*-3**: ***Determine the updated Location: The optimal solutions are identified using the Fraction Rider HSA, and the updated equation is illustrated as,

$$\begin{aligned} C_{p+1}(u,v) = {} & (\lambda-1)C_p(u,v) + \frac{1}{2}\lambda C_{p-1}(u,v) + \frac{1}{6}(1-\lambda)C_{p-2}(u,v) \\ & + \frac{1}{24}\lambda(1-\lambda)(2-\lambda)C_{p-3}(u,v) \\ & + \frac{\alpha\left[C_p(\gamma,v)*(1-\phi(v))\right] \pm rand(0,1)\cdot S*\phi(v)}{1-\alpha\phi(v)} \end{aligned} \tag{26}$$

where, S specifies arbitrary distance bandwidth and $rand(0,1)$ is a random number and it lies from [0, 1].

Step*-4**: ***Evaluating feasibility: The feasibility is estimated using the fitness value, if a new value is better than previous one, then update the previous solution with new solution.

Step*-5**: ***Termination: The above processes are continual until the best solution is achieved for resource allocation process.

5. Results and Discussion

The result of proposed Fractional Rider Deep LSTM approach is illustrated in this section. The performance of developed technique is evaluated by various parameters, namely energy, MOS, delay, fairness, QE and error.

5.1. Experimental Setup

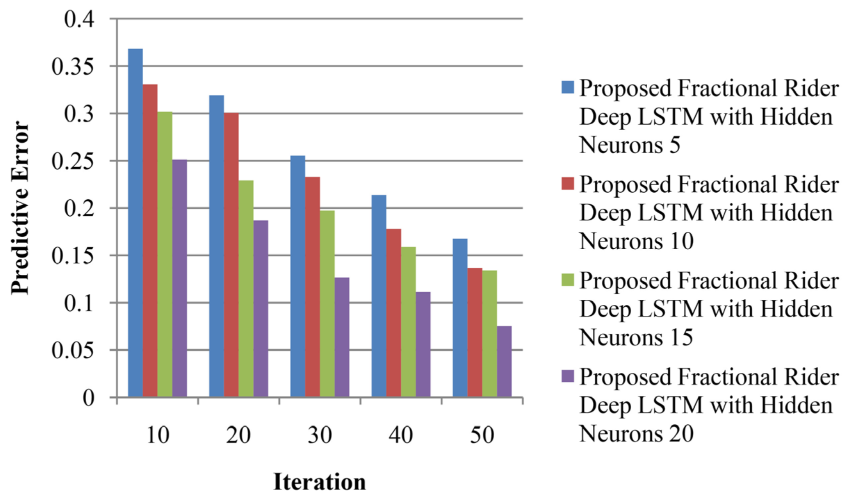
The developed Fractional Rider LSTM technique is executed in PYTHON with PC contains Windows 10 OS, 4GB RAM, and Intel i3 core processor.

5.2. Performance Metrics

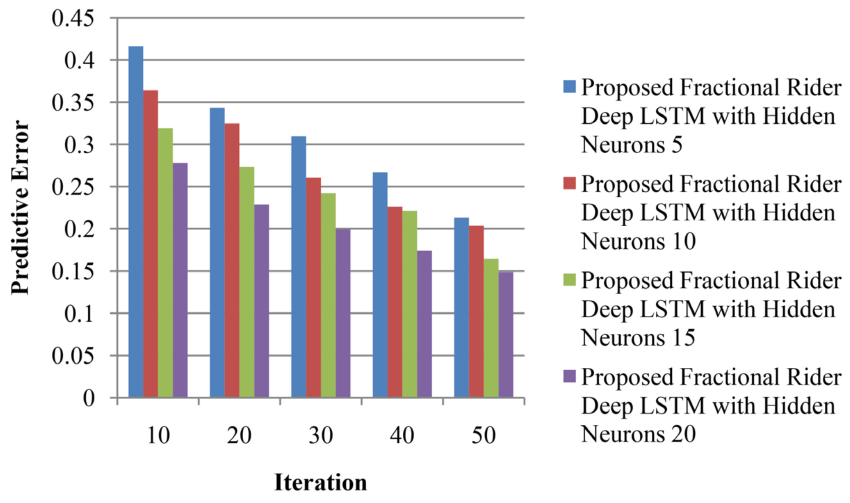
The performance metrics considered for analysis of existing cloud gaming techniques are fairness, MOS, QE, energy and delay. The detailed explanation of these metrics is illustrated in Section 5.

5.3. Performance Analysis

This section illustrates about the performance analysis of developed Fractional Rider Deep LSTM technique based on predictive error with different hidden layer. The performance analysis of developed Fractional Rider Deep LSTM with predictive error is depicted in **Figure 4**. **Figure 4(a)** portrays the performance analysis of developed Fractional Rider Deep LSTM using predictive error with different hidden layer for 200 game size. The developed Fractional Rider Deep LSTM based on predictive error with hidden layer 5 is 0.167, 10 is 0.136, 15 is 0.133 and 20 is 0.075 for 50th iteration. Similarly, the performance analysis of developed Fractional Rider Deep LSTM based on predictive error with various hidden layers for game size 300 is showed in **Figure 4(b)**. In 50th iteration, the



(a)



(b)

Figure 4. Performance analysis of developed Fractional Rider LSTM based on predictive error with different hidden layer.

developed Fractional Rider Deep LSTM using predictive error with hidden layer 5, 10, 15, 20 is 0.213, 0.203, 0.164 and 0.148.

5.4. Comparative Methods

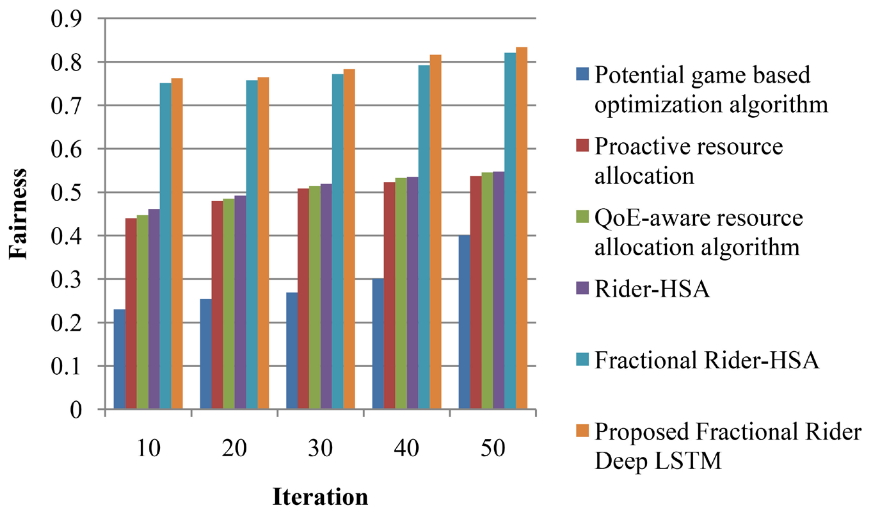
The developed Fractional Rider Deep LSTM technique is analyzed with comparative approaches, like Potential game based optimization algorithm, Proactive resource allocation algorithm [36], QoE-aware resource allocation algorithm [33], Rider-HSA method [34] and Fractional Rider-HSA technique for computing the performance.

5.5. Comparative Analysis

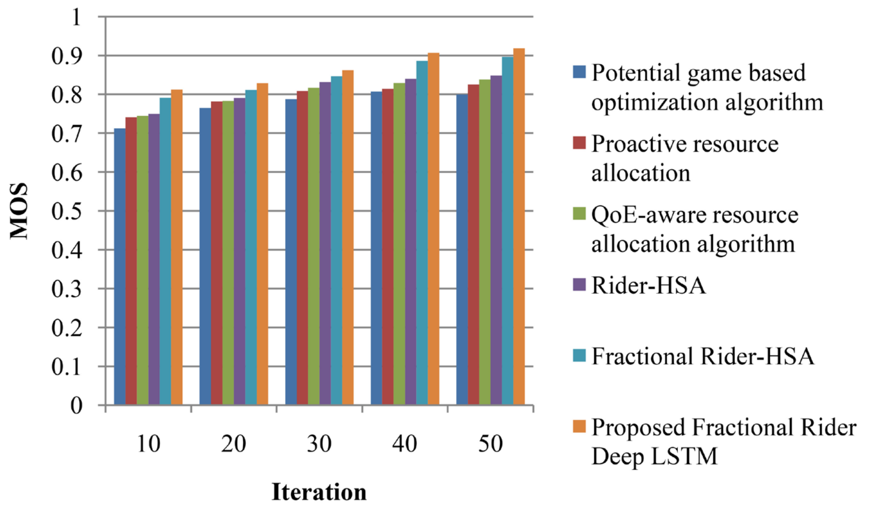
The comparative analysis of developed Fractional Rider Deep LSTM technique with existing systems are performed based on several parameters, like fairness, MOS, QE, energy and delay by varying amount of iterations.

1) Analysis with game size 200

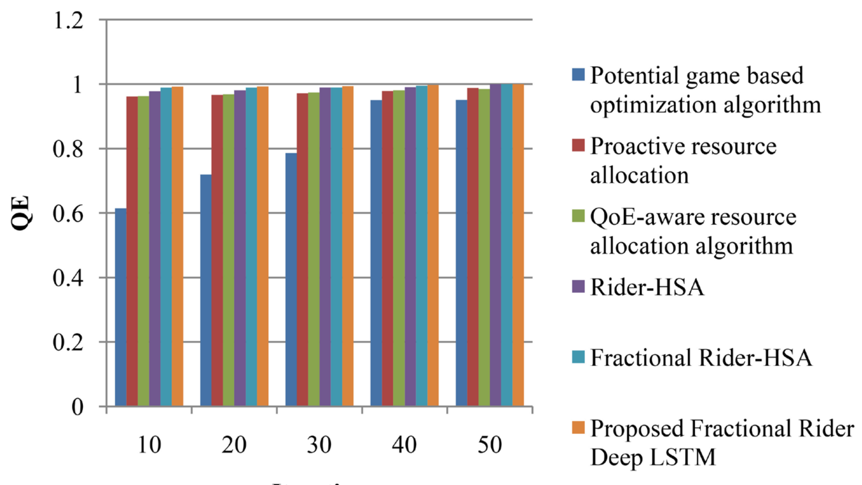
The comparative analysis of developed Fractional Rider Deep LSTM using fairness, MOS, QE, energy and delay with game size 200 is illustrated in Figure 5. Figure 5(a) shows the analysis of developed Fractional rider Deep LSTM model based on fairness by varying the number of iteration. The fairness value of existing workload prediction techniques, such as Potential game-based optimization algorithm is 0.300, Proactive resource allocation is 0.522, QoE-aware resource allocation technique is 0.532, Rider-HSA is 0.535, Fractional Rider-HSA is 0.792, whereas developed Fractional Rider Deep LSTM is 0.816 for 40th number of iterations. The comparative analysis of developed Fractional Rider Deep LSTM based on MOS by changing the number of iteration is displayed in Figure 5(b). When the number of iterations is 40, MOS value attained by Potential game-based optimization approach is 0.807, Proactive resource allocation model is 0.814, QoE-aware resource allocation algorithm is 0.828, Rider-HSA is 0.839,



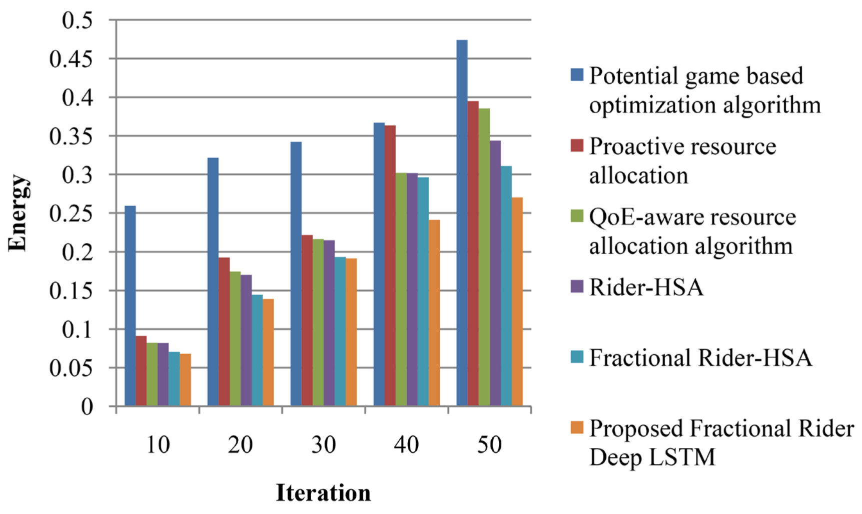
(a)



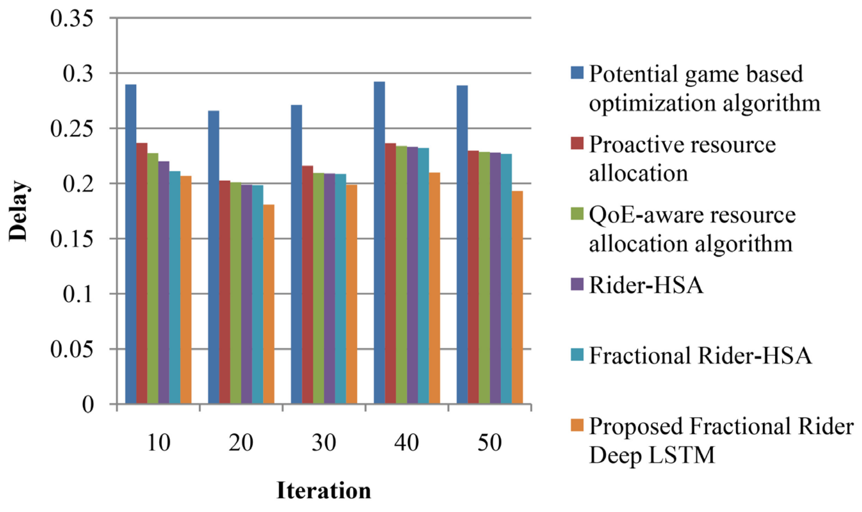
(b)



(c)



(d)



(e)

Figure 5. Analysis of developed Fractional Deep LSTM for game size 200 based on (a) fairness, (b) MOS, (c) QE, (d) Energy, (e) Delay.

Fractional Rider-HSA is 0.885 and developed Fractional Rider Deep LSTM technique is 0.907.

Figure 5(c) portrays the analysis of developed Fractional rider Deep LSTM technique using QE by varying iteration. For 40th iteration, QE value obtained by developed model is 0.997, while the existing techniques, such as Potential game based optimization technique, Proactive resource allocation, QoE-aware resource allocation scheme, Rider-HSA and Fractional Rider-HSA has 0.950, 0.978, 0.980, 0.990 and 0.995. The analysis of developed Fractional Rider Deep LSTM using energy by varying the iteration is indicated in Figure 5(d). When, number of iteration is 40, energy obtained by Potential game-based optimization approach is 0.367, Proactive resource allocation model is 0.363, QoE-aware resource allocation algorithm is 0.302, Rider-HSA is 0.301, Fractional Rider-HSA is 0.296 and developed Fractional Rider Deep LSTM approach is 0.241. Likewise, Figure 5(e) shows the comparative analysis of developed Fractional rider Deep LSTM based on delay parameter by varying the iteration number. The delay obtained by existing techniques, such as Potential game-based optimization algorithm is 0.292, Proactive resource allocation is 0.236, QoE-aware resource allocation technique is 0.233, Rider-HSA is 0.233, Fractional Rider-HSA is 0.232 and developed Fractional Rider Deep LSTM is 0.209 for 40th number of iteration.

2) Analysis with game size 300

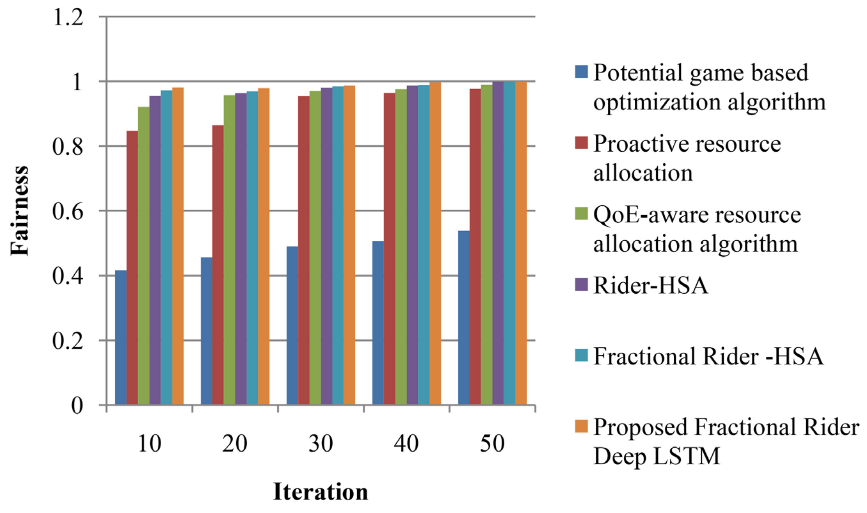
The comparative analysis of developed Fractional Rider Deep LSTM using fairness, MOS, QE, energy and delay with game size 300 is represented in Figure 6. Figure 6(a) displays the analysis of developed work prediction technique based on fairness by changing the number of iteration. For 40th iteration, the fairness obtained by developed model is 0.997, while the existing techniques, such as Potential game-based optimization technique, Proactive resource allocation, QoE-aware resource allocation scheme, Rider-HSA and Fractional Rider-HSA has 0.506, 0.964, 0.976, 0.987 and 0.988. The comparative analysis of developed Fractional Rider Deep LSTM using MOS by varying the number of iteration is presented in Figure 6(b). The MOS value of existing techniques, like Potential game-based optimization algorithm is 0.773, Proactive resource allocation is 0.798, QoE-aware resource allocation technique is 0.822, Rider-HSA is 0.852, Fractional Rider-HSA is 0.893, whereas developed Fractional Rider Deep LSTM method is 0.915 for 40th number of iteration.

Figure 6(c) shows the comparative analysis of developed Fractional rider Deep LSTM based on QE parameter by chancing the iteration number. When, number of iteration is 40, the QE value obtained by Potential game-based optimization approach is 0.937, Proactive resource allocation model is 0.935, QoE-aware resource allocation algorithm is 0.975, Rider-HSA is 0.986, Fractional Rider-HSA is 0.988 and developed Fractional Rider Deep LSTM technique is 0.998. Similarly, the analysis of developed Fractional rider Deep LSTM using energy by varying the iteration number is indicated in Figure 6(d). The energy of existing techniques, such as Potential game-based optimization algorithm is

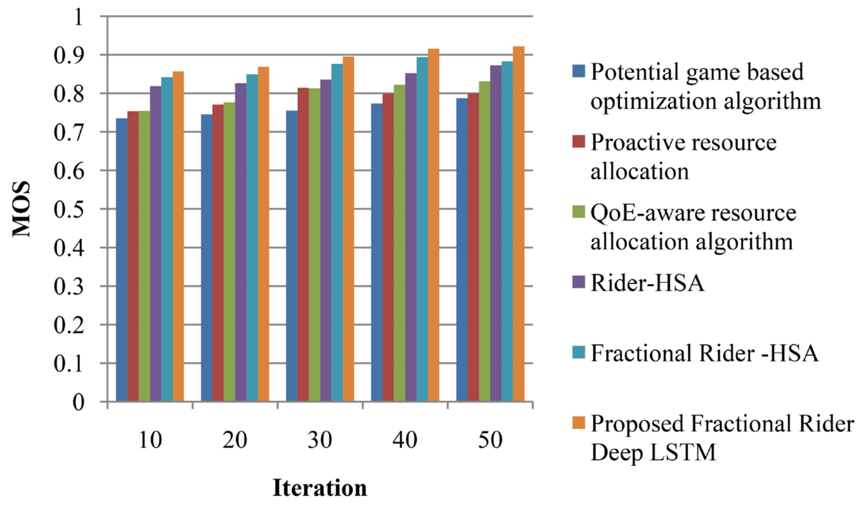
0.396, Proactive resource allocation is 0.336, QoE-aware resource allocation technique is 0.329, Rider-HSA is 0.295, Fractional Rider-HSA is 0.268, when developed Fractional rider Deep LSTM is 0.202 for 40th number of iterations. In addition, Figure 6(e) portrays the analysis of developed Fractional rider Deep LSTM technique using delay by varying iteration number. When number of iteration is 40, the delay obtained by Potential game-based optimization approach is 0.630, Proactive resource allocation model is 0.463, QoE-aware resource allocation algorithm is 0.461, Rider-HSA is 0.459, Fractional Rider-HSA is 0.459 and developed Fractional Rider Deep LSTM method is 0.457.

5.6. Comparative Discussion

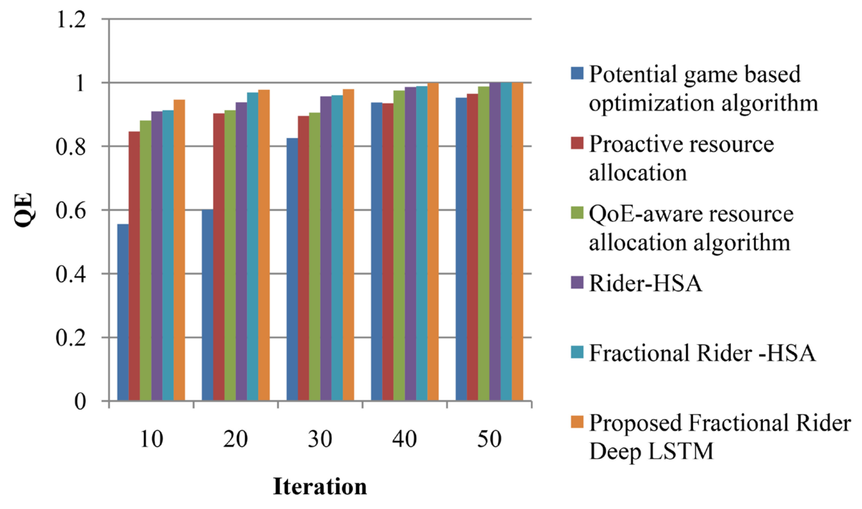
Table 1 represents the analysis of work load prediction approaches using fairness, MOS, QE, energy and delay parameters with game size 200 and 300. Here, the maximum fairness, MOS, QE and minimum energy and delay are considered as the best performance. From the table, the maximum fairness, MOS and QE is



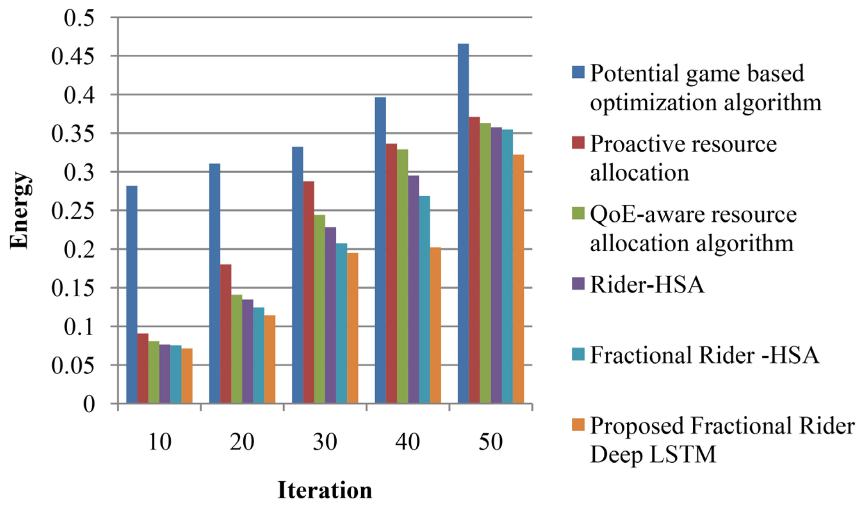
(a)



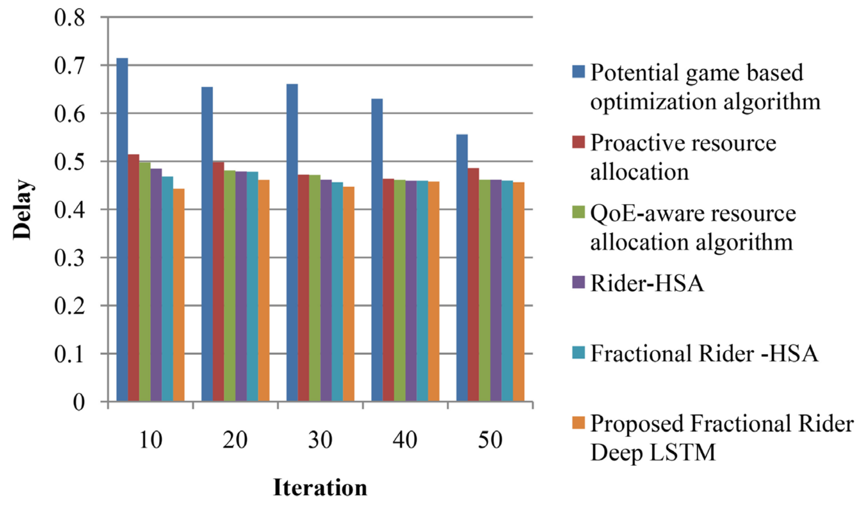
(b)



(c)



(d)



(e)

Figure 6. Analysis of developed Fractional Deep LSTM for game size 300 based on (a) fairness, (b) MOS, (c) QE, (d) Energy, (e) Delay.

Table 1. Comparative discussion.

Game size	Metrics	Potential game-based optimization algorithm	Proactive resource allocation algorithm	QoE-aware resource allocation algorithm	Rider-HSA	Fractional Rider-HSA	Proposed Fractional Rider Deep LSTM
200	Fairness	0.400	0.537	0.545	0.547	0.820	**0.834**
	MOS	0.799	0.824	0.837	0.848	0.896	**0.918**
	QE	0.951	0.987	0.985	0.995	0.997	**0.999**
	Energy	0.474	0.394	0.385	0.343	0.310	**0.270**
	Delay	0.288	0.229	0.228	0.227	0.226	**0.193**
300	Fairness	0.538	0.977	0.989	0.994	0.997	**0.999**
	MOS	0.787	0.799	0.830	0.873	0.883	**0.921**
	QE	0.952	0.965	0.987	0.996	0.998	**0.999**
	Energy	0.465	0.371	0.363	0.357	0.354	**0.322**
	Delay	0.556	0.486	0.461	0.461	0.460	**0.456**

obtained when the game size is 300 and minimum energy and delay are occurred when the game size is 200. In the proposed method, the multi objective function produced using several parameters, such as network definition factor, energy, gaming experience loss, fairness, predictive load, load and MOS. Also, the Rider Deep LSTM enhances the real time fault prediction. Moreover, workload of every resource is estimated and the workload is predicted using the developed Fractional Rider Deep LSTM network. Thus, from the below comparative discussion table it is well known, that the developed Fractional Rider Deep LSTM approach achieved maximum fairness, MOS, QE and less delay and energy.

6. Conclusion

This paper presents an effective workload prediction method based on developed Fractional Rider Deep LSTM network. Here, workload prediction-based distributed resource allocation system is developed in cloud gaming. Initially, workload of every resource is estimated using developed Fractional Rider Deep LSTM network. After that, resource allocation is performed based on fractional Rider-based HSA. Here, the Fractional Rider-based HSA is the combination of HSA, ROA and FC model. Once resource allocation is completed, workload is predicted using the developed Fractional Rider Deep LSTM network. The developed Fractional Rider Deep LSTM network is developed by integrating FC scheme and Rider Deep LSTM network method. Meanwhile, multi-objective parameters, such as MOS, fairness, predictive load, gaming experience loss, network parameter and energy are considered for efficient workload prediction. Along with this, the metrics, such as fairness, MOS, QE, energy and delay are considered for evaluating the performance of the developed method. Moreover, the developed workload prediction technique obtained maximum fairness of 0.999, MOS of

0.921, QE of 0.999 and minimal energy of 0.322 and delay of 0.456 for game size 300.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Vafamehr, A. and Khodayar, M.E. (2018) Energy Aware Cloud Computing. *The Electricity Journal*, **31**, 40-49. https://doi.org/10.1016/j.tej.2018.01.009

[2] Aboutorabi, S. and Rezvani, M.H. (2020) An Optimized Meta-Heuristic Bees Algorithm for Players' Frame Rate Allocation Problem in Cloud Gaming Environments. *The Computer Games Journal*, **9**, 281-304. https://doi.org/10.1007/s40869-020-00106-4

[3] Gopal, D.G. and Kaushik, S. (2017) Emerging Technologies and Applications for Cloud-Based Gaming: Review on Cloud Gaming Architectures. In: *Emerging Technologies and Applications for Cloud-Based Gaming*, IGI Global, Hershey, 67-87. https://doi.org/10.4018/978-1-5225-0546-4.ch003

[4] Mishra, M., Das, A., Kulkarni, P. and Sahoo, A. (2012) Dynamic Resource Management Using Virtual Machine Migrations. *IEEE Communications Magazine*, **50**, 34-40. https://doi.org/10.1109/MCOM.2012.6295709

[5] Wei, W., Fan, X.L., Song, H.B., Fan, X.M. and Yang, J.C. (2018) Imperfect Information Dynamic Stackelberg Game Based Resource Allocation Using Hidden Markov for Cloud Computing. *IEEE Transactions on Services Computing*, **11**, 78-89. https://doi.org/10.1109/TSC.2016.2528246

[6] Chen, K.T., Chang, Y.C., Tseng, P.H., Huang, C.Y. and Lei, C.L. (2011) Measuring the Latency of Cloud Gaming Systems. *Proceedings of the* 19*th ACM International Conference on Multimedia*, Scottsdale, 28 November-1 December 2011, 1269-1272. https://doi.org/10.1145/2072298.2071991

[7] Dinaki, H.E., Shirmohammadi, S. and Hashemi, M.R. (2020) Boosted Metaheuristic Algorithms for QoE-Aware Server Selection in Multiplayer Cloud Gaming. *IEEE Access*, **8**, 60468-60483. https://doi.org/10.1109/ACCESS.2020.2983080

[8] Mastelic, T., Oleksiak, A., Claussen, H., Brandic, I., Pierson, J.M. and Vasilakos, A.V. (2014) Cloud Computing: Survey on Energy Efficiency. *ACM Computing Surveys*, **47**, 1-36. https://doi.org/10.1145/2656204

[9] Han, Y.W., Guo, D.Y., Cai, W., Wang, X.F. and Leung, C.M. (2020) Virtual Machine Placement Optimization in Mobile Cloud Gaming through QoE-Oriented Resource Competition. *IEEE Transactions on Cloud Computing*. https://doi.org/10.1109/TCC.2020.3002023

[10] Amiri, M. and Mohammad-Khanli, L. (2017) Survey on Prediction Models of Applications for Resources Provisioning in Cloud. *Journal of Network and Computer Applications*, **82**, 93-113. https://doi.org/10.1016/j.jnca.2017.01.016

[11] Kumar, K.D. and Umamaheswari, E. (2018) Prediction Methods for Effective Resource Provisioning in Cloud Computing: A Survey. *Multiagent and Grid Systems*, **14**, 283-305. https://doi.org/10.3233/MGS-180292

[12] Singh, N. and Rao, S. (2012) Online Ensemble Learning Approach for Server Workload Prediction in Large Datacenters. *The Eleventh International Conference*

on Machine Learning and Applications (*ICMLA* 2012), Boca Raton, 12-15 December 2012, 68-71. https://doi.org/10.1109/ICMLA.2012.213

[13] Van Der Voort, M., Dougherty, M. and Watson, S. (1996) Combining Kohonen Maps with ARIMA Time Series Models to Forecast Traffic Flow. *Transportation Research Part C*: *Emerging Technologies*, **4**, 307-318. https://doi.org/10.1016/S0968-090X(97)82903-8

[14] Zhang, W., Li, B., Zhao, D., Gong, F. and Lu, Q. (2016) Workload Prediction for Cloud Cluster Using a Recurrent Neural Network. *International Conference on Identification*, *Information and Knowledge in the Internet of Things* (*IIKI*), Beijing, 20-21 October 2016, 104-109. https://doi.org/10.1109/IIKI.2016.39

[15] Gupta, S. and Dinesh, D.A. (2017) Resource Usage Prediction of Cloud Workloads Using Deep Bidirectional Long Short Term Memory Networks. 2017 *IEEE International Conference on Advanced Networks and Telecommunications Systems* (*ANTS*), Bhubaneswar, 17-20 December 2017, 1-6. https://doi.org/10.1109/ANTS.2017.8384098

[16] Amiri, M., Mohammad-Khanli, L. and Mirandola, R. (2018) An Online Learning Model Based on Episode Mining for Workload Prediction in Cloud. *Future Generation Computer Systems*, **87**, 83-101. https://doi.org/10.1016/j.future.2018.04.044

[17] Liu, X.-F., Zhan, Z.-H., Deng, J.D., Li, Y., Gu, T.L. and Zhang, J. (2018) An Energy Efficient Ant Colony System for Virtual Machine Placement in Cloud Computing. *IEEE Transactions on Evolutionary Computation*, **22**, 113-128. https://doi.org/10.1109/TEVC.2016.2623803

[18] Devagnanam, J. and Elango, N.M. (2020) Optimal Resource Allocation of Cluster using Hybrid Grey Wolf and Cuckoo Search Algorithm in Cloud Computing. *Journal of Networking and Communication Systems*, **3**, 31-40. https://doi.org/10.46253/jnacs.v3i1.a4

[19] Netaji, V.K. and Bhole, G.P. (2020) Optimal Container Resource Allocation Using Hybrid SA-MFO Algorithm in Cloud Architecture. *Multimedia Research*, **3**, 11-20. https://doi.org/10.46253/j.mr.v3i1.a2

[20] Mirjalili, S., Mirjalili, S.M. and Lewis, A. (2014) Grey Wolf Optimizer. *Advances in Engineering Software*, **69**, 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[21] Wang, G., Deb, S. and Coelho, L.S. (2015) Elephant Herding Optimization. *Proceedings of* 3*rd International Symposium on Computational and Business Intelligence* (*ISCBI*), Bali, 7-9 December 2015, 1-5. https://doi.org/10.1109/ISCBI.2015.8

[22] Amiri, M., Osman, H.A., Shirmohammadi, S. and Abdallah, M. (2016) Toward Delay-Efficient Game-Aware Data Centers for Cloud Gaming. *ACM Transactions on Multimedia Computing*, *Communications*, *and Applications* (*TOMM*), **12**, 1-9. https://doi.org/10.1145/2983639

[23] Amiri, M., Sobhani, A., Al Osman, H. and Shirmohammadi, S. (2017) SDN-Enabled Game-Aware Routing for Cloud Gaming Datacenter Network. *IEEE Access*, **5**, 18633-18645. https://doi.org/10.1109/ACCESS.2017.2752643

[24] Bhaladhare, P.R. and Jinwala, D.C. (2014) A Clustering Approach for the l-Diversity Model in Privacy Preserving Data Mining Using Fractional Calculus-Bacterial Foraging Optimization Algorithm. *Advances in Computer Engineering*, **2014**, Article ID: 396529. https://doi.org/10.1155/2014/396529

[25] Binu, D. and Kariyappa, B.S. (2020) Rider Deep LSTM Network for Hybrid Distance Score-Based Fault Prediction in Analog Circuits. *IEEE Transactions on Industrial Electronics*.

[26] Chakraborty, P., Roy, G.G., Das, S., Jain, D. and Abraham, A. (2009) An Improved

Harmony Search Algorithm with Differential Mutation Operator. *Fundamenta Informaticae*, **95**, 401-426. https://doi.org/10.3233/FI-2009-157

[27] Binu, D. and Kariyappa, B.S. (2018) RideNN: A New Rider Optimization Algorithm-Based Neural Network for Fault Diagnosis in Analog Circuits. *IEEE Transactions on Instrumentation and Measurement*, **68**, 2-26. https://doi.org/10.1109/TIM.2018.2836058

[28] Fernández-Cerero, D., Jakóbikb, A., Fernández-Montesa, A. and Kołodziejb, J. (2019) GAME-SCORE: Game-Based Energy-Aware Cloud Scheduler and Simulator for Computational Clouds. *Simulation Modelling Practice and Theory*, **93**, 3-20. https://doi.org/10.1016/j.simpat.2018.09.001

[29] Aslanpour, M.S., Ghobaei-Arani, M., Heydari, M. and Mahmoudi, N. (2019) LARPA: A Learning Automata-Based Resource Provisioning Approach for Massively Multiplayer Online Games in Cloud Environments. *International Journal of Communication Systems*, **32**, e4090. https://doi.org/10.1002/dac.4090

[30] Li, Y.S., Zhao, C.J., Tang, X.Y., Cai, W.T., Liu, X.G., Wang, G. and Gong, X.L. (2020) Towards Minimizing Resource Usage with QoS Guarantee in Cloud Gaming. *IEEE Transactions on Parallel and Distributed Systems*, **32**, 426-440. https://doi.org/10.1109/TPDS.2020.3024068

[31] Ghobaei-Arani, M., Khorsand, R. and Ramezanpour, M. (2019) An Autonomous Resource Provisioning Framework for Massively Multiplayer Online Games in Cloud Environment. *Journal of Network and Computer Applications*, **142**, 76-97. https://doi.org/10.1016/j.jnca.2019.06.002

[32] Bhojan, A., Ng, S.P., Ng, J. and Ooi, W.T. (2020) CloudyGame: Enabling Cloud Gaming on the Edge with Dynamic Asset Streaming and Shared Game Instances. *Multimedia Tools and Applications*, **43**, 32503-32523. https://doi.org/10.1007/s11042-020-09612-z

[33] Slivar, I., Skorin-Kapov, L. and Suznjevic, M. (2019) QoE-Aware Resource Allocation for Multiple Cloud Gaming Users Sharing a Bottleneck Link. *Proceedings of* 22*nd Conference on Innovation in Clouds*, *Internet and Networks and Workshops* (*ICIN*), Paris, 19-21 February 2019, 118-123. https://doi.org/10.1109/ICIN.2019.8685890

[34] Guo, D.Y., Han, Y.W., Cai, W., Wang, X.F. and Victor, C.M. (2019) QoE-Oriented Resource Optimization for Mobile Cloud Gaming: A Potential Game Approach. *ICC* 2019 *IEEE International Conference on Communications* (*ICC*), Shanghai, 20-24 May 2019, 1-6.

[35] Ge, Y., Zhang, Y.K., Qiu, Q.R. and Lu, Y.-H. (2012) A Game Theoretic Resource Allocation for Overall Energy Minimization in Mobile Cloud Computing System. *Proceedings of* 2012 *ACM*/*IEEE International Symposium on Low Power Electronics and Design*, Redondo Beach, 30 July-1 August 2012, 279-284.

[36] Haouari, F., Baccour, E., Erbad, A., Mohamed, A. and Guizani, M. (2019) QoE-Aware Resource Allocation for Crowdsourced Live Streaming: A Machine Learning Approach. *International Conference on Communications* (*ICC*), Shanghai, 20-24 May 2019, 1-6. https://doi.org/10.1109/ICC.2019.8761591