JOURNAL OF CRITICAL REVIEWS

ISSN- 2394-5125 VOL 06. ISSUE 06. 2019

AN OPTIMIZED HYBRID MODEL FOR LOAD BALANCING IN CLOUD USING MACHINE LEANING

[1]Dr. BVS Varma, Professor, Dept. of CSE, DNR College of Engineering and Technology, Bhimavaram, A.P, India

[2]Dr. A Ramamurthy, Professor, Dept. of CSE, DNR College of Engineering and Technology, Bhimavaram, A.P, India

ABSTRACT: Despite significant infrastructure improvements in Cloud computing, still faces numerous challenges in terms of load balancing. There have been various metaheuristic approaches being used for proper load balancing in the cloud. However, most of the approaches consider only a single QoS (Quality of Service) metric and ignore many important factors. The performance efficiency of these approaches can be further enhanced by implementing with machine learning (ML) techniques. An optimized hybrid model for load balancing in cloud using machine leaning is presented in this paper. An optimized hybrid model that combines Support Vector Machine (SVM) along with Cat Swarm Optimization (CSO) and Ant Colony Optimization (ACO) is being implemented with File Type Formatting (FTF) to provide a better load balancing solution in cloud computing. The classification is performed using SVM considering various file formats such as audio, video, text maps, and images in the cloud. The resultant data class provides high classification accuracy which is further fed into a metaheuristic algorithm namely ACO using FTF for better load balancing in the cloud. Then, the data is input to the modified load balancing algorithm CSO that efficiently distributes the load on Virtual machines (VMs). Simulation results compared to existing approaches showed an improved performance of QoS metrics such as SLA (Service Level Agreement) violation, throughput time, response time, migration time, energy consumption and average execution time.

KEY WORDS: Cloud computing, Load balancing, Machine learning, QoS, VMs, SVM, CSO, ACO and FTF.

I. INTRODUCTION

Cloud computing is a method of providing services to the user by using internet through web-based technology and applications [1]. Cloud computing possess distributed technologies to satisfy a variety of applications and user needs.

Sharing resources, software, information via internet are the main interest of cloud computing with an aim to reduced capital and data transfer cost, better performance in terms of response time and data processing time, maintain the system [2]. So there are various technical challenges that needs to be addressed like Virtual machine relocation, server consolidation, fault tolerance, high availability and scalability but central issue is the load balancing, it is the mechanism of spreading the load among various nodes of a distributed system to improve both resource deployment and job response time while also avoiding a situation where some of the nodes are having huge amount of load while other nodes are doing nothing or idle with very little work [3]. In Cloud computing environment, load balancing needs to distribute the dynamic local workload evenly between all the nodes. In load balancing transfer the whole load of the system to the one node to another node of the system for better resource utilization and to diversity the response time of the job, simultaneously removing a state in which some of the node are over loaded while some other are under loaded [4]. It is used to gain a high user fulfilment and resource utilization ratio, hence improving the overall performance of the system. Proper load balancing can help in utilization of the available resources optimally, thereby minimizing the resource consumption. It also help in implementing enabling scalability, fault tolerant and over-provisioning, reducing response time etc [5]. Load balancing faces one of the challenges to distribute a large amount of data and allocate a suitable resource at the time of task allocation. One of the challenges of task scheduling in cloud is to assign the tasks to different VMs so that the load balancing is achieved with minimum resources [6]. The advantage is to better utilize the resources on cloud and fulfill the demands of users in a timely manner. These approaches combine the relative benefits of load balancing algorithm backed up by powerful machine learning models such as Support Vector Machines (SVM). In the cloud, data exists in huge volume and variety that requires extensive computations for its accessibility, and hence performance efficiency is a major concern. Recent research manifested that load balancing techniques based on metaheuristics provide better solutions for proper scheduling and allocation of resources in the cloud [7]. These approaches often adopt multi-objective QoS metrics, such as reduced SLA violations, reduced makespan, high throughput, low overload, low energy consumption, high optimization, minimum migrations, and higher response time.

II. LITERATURE SURVEY

With the availability of high-speed internet at a lower cost and shoot up in the storage and processing technologies, cloud computing resources are more powerful and available at a cheaper cost. Generally, in a cloud computing scenario, the tasks are assigned to VM's if they can do the task and they have enough resources. While one task is running, we assign another task to the same VM without checking the availability of sufficiency of bandwidth because of this task will be waiting without allowing the next task to execute so the overall execution time increases and causes the underutilization of resources. To overcome this in [8] a task scheduling for cloud computing has been proposed which follows a non-linear programming model. In this methodology, it sets a condition that tasks are independent of each other and allocate proper no of tasks to each VM depending on the available network bandwidth. Comparing the results showed that it took less time for the completion of tasks by using this algorithm.

BhawnaMallick [9] work on use of virtual load balancer, which check the status of current virtual machine and give this return id to virtual machine, which are least, loaded. If virtual machine is not found, then it returns the value of -1 then datacenter controller queues up the incoming request. RenGao et.al [10] works on the two strategies forwardbackward technique and max-min protocols so that to find the candidate node and distribution of load according to it. In these pheromones, initialization and pheromones bring to update according to the physical resources. Using master slave architecture as like the first task is presented to the master node after the load distribution is decided by the master node. Ekta Gupta et.al [11] algorithm is evoked by using ACO (ant colony optimization) that ant work togetherin foraging character of ant. The intensity of pheromone trail is varying on different factors such as quality of food sources; calculate distance of food from source etc. So, according to this pheromone updates the ant decides the next node. Nikita

Haryani et.al [12] works on counter variable that start checking the variable counter of each server node and data center processor. Load balancing method is count on on least link mechanism and it is belonging to the dynamic organizing procedure. it counts the number of connections for each server dynamically to find the approximate the load. An optimized task scheduling algorithm was proposed in [13], It compares the execution time of all tasks over fastest and slowest executing resources and compare the values and try to get an optimized value accordingly applied either ResourceAware-Scheduling algorithm (RASA) or min-in scheduling algorithm. The results were optimized makespan.

In [14] multi-objective tasks scheduling algorithm for cloud computing optimization of throughput was proposed. The main idea behind this work is that rather than having only one criterion for scheduling, it is required to consider various criteria like execution time, cost, the bandwidth of user, etc. The improved cost-based algorithm for task scheduling in cloud computing was proposed in [15]. The strategy used here is to group the tasks in cloud computing technology. Resources may vary in their cost and their performance. Rather than randomly assigning tasks to the user's group the tasks corresponding to the capacity of each cloud resource and later send grouped jobs to the respective resources. The activity-based costing algorithm was taken into consideration by taking different parameters like processing cost and processing time with grouping and without grouping. The results were better with grouping than without grouping of tasks

II. AN OPTIMIZED HYBRID MODEL FOR LOAD BALANCING IN CLOUD USING ML

The model framework combining the Ml techniques of SVM along with CSO and ACO to make an optimized hybrid model with the objective of improving the load balancing and performance in the cloud environment. The architecture of the optimized hybrid model using machine learning techniques is shown in Fig. 1. This architecture is divided into two main modules: ‘Data Classification based on SVM’ and ‘Load Balancing using CSO and ACO.

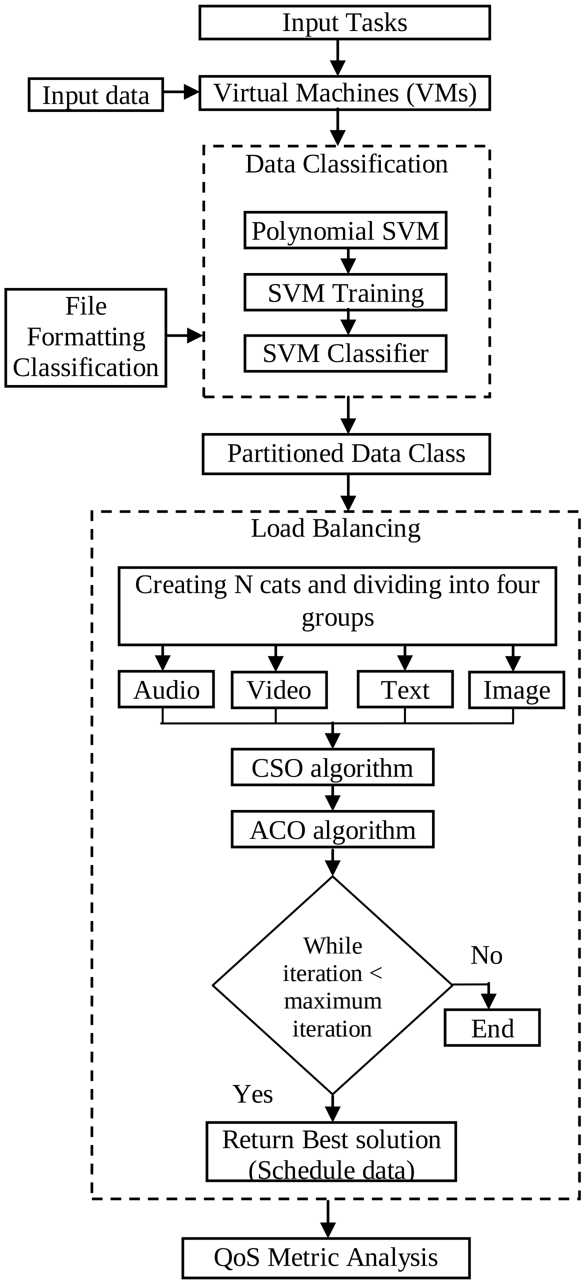

Fig. 1: Architecture of Optimized Hybrid Model Using Machine Learning Techniques

The input to the data classification module is the collection of diverse data in the form of video, text, audio, and images, which are stored in the cloud environment. The classification module takes the input data randomly and then performs the classification on these data using polynomial SVM. The output of this algorithm is in the form of the partitioned data class. The second module performs load balancing using CSO and ACO. ACO is selected by considering its capabilities such as: ACO is dominant over genetic algorithms and simulated annealing approaches, as the convergence time of ACO is faster than genetic algorithms. The performance analysis of the proposed model is then performed to achieve an efficient load balancing by considering the parameters such as execution time, number of migrations, optimization time, throughput time, and overhead time.

3.1 Data Classification

Input Data: Data is collected from different cloud sources and then pre-processed to transform it as per our model requirements. The format of the collected data from the cloud is comprised of video, audio, text, and images. These data sets are diverse and are of different sizes.

Classification Using SVM: The data collected are then classified with the help of SVM. At first, the SVM classifier determines the type of data (audio, video, text, image) based on features and then classify the data by assigning it to a particular class. For this SVM introduces Kernel function to change the original data space into a higher dimension space having a function that includes the transformation function with dot product. VMs are divided into four types of sets, such as AudioVM, VideoVM, TextVM, and ImageVM based on input data. Each set of VM has different processing and storage resources in a cloud environment. More precisely, each machine (VM) is assigned a task based on task requirements. After that, the SVM classifier identifies the set of VM types based on the requirements, size, and features of the tasks. Here the respective VM is assigned concerning each task. Hence, SVM intends to classify data and match it to the most suitable class type and VM type. There are two types of classic kernel functions that are used in SVMs, one of them is the radial basis function kernel and the other is a polynomial kernel. Where u_i, is used for support vector, $\propto_i$ is represented as Lagrange multiplier and u_j is known as the label of membership class (+1, -1) where n=1,2,3…. N. Equation (3) shows the polynomial function,

$$POLY(u,v) = \left((u^k v + 1)\right)^s \text{ ------ (1)}$$

where ‘s’ is the polynomial degree. The polynomial kernel function is used with SVMs and other kernel models representing the similarity between features over the polynomials of the original variables. A polynomial kernel is defined as:

$$K(x, x_i) = 1 + \sum (x \, X \, x_i)^d \text{ ------- (2)}$$

Here, d=1, this confirms to the linear kernel. The output of the classification phase is in the form of classified tasks, thus reducing the computational cost such as to avoid pre-processing of features learning, features extraction, data conversion, data transformation, and data classification at the scheduling phase.

3.2 Load Balancing

In the second part, load balancing is performed over the classified data. Here ACO is used that performs task scheduling and as a result, it returns the complete scheduled data. The network of VMs is developed in the form of an undirected weighted graph as shown in Fig. 2. The VMs’ network can be represented as an undirected graph $G = (V, E)$ where V represents the virtual

machine (VM) or node and E represents the undirected edge having a probability a weight that shows the overload and underload intensity between two nodes. After the data classification, load balancing is performed and mapping of tasks on virtual machines is computed using a metaheuristic algorithm called CSO and ACO where each machine is assigned a task based on available resources in cloud environment. In the load balancing phase, these data are called tasks. Let us assume set of virtual machines for video, audio, text, image, respectively as,
$VideoVM = \{VM_1, VM_2, \ldots., VM_n\}$,
$AudioVM = \{VM_1, VM_2, \ldots., VM_n\}$,
$TextVM = \{VM_1, VM_2, \ldots., VM_n\}$,
$ImageVM = \{VM_1, VM_2, \ldots., VM_n\}$
Each set of machines is responsible to execute one task. Each task is executed for a period of maximum iterations and is evaluated using computational cost in the form of time. The mapping of tasks on virtual machines is computed using the SVM, where each machine is assigned a task based on requirements, size, and features of the tasks. Once the number of tasks and VMs are selected, the scheduling process will be initiated. Initially, N instances are created and split into G groups.

CSO algorithm: CSO takes into consideration the behavior of the cats into two modes that are seeking mode and tracing mode. The cat behavior is considered for searching a solution space. Every cat has its position having d-dimensions with different velocities used for each dimension. Every cat is evaluated using fitness function, if the fitness is not equal then compute the probability using equation (3), and by default, the probability value is set to 1. We used the Boolean flag variable to identify whether the cat is in seeking mode or tracing mode. The tracing mode is considered in terms of its fitness function where the position of the cat is changed according to the fitness function. The fitness function of CSO can be obtained with the help of equation (5).

$P_i = \frac{FS_i - FS_{max}}{FS_{max} - FS_{min}}$, where $0 < i < j$, ----- (3)

where P_i shows the probability, value associated with the position of ith Cat. FS_i is the fitness of ith cat, FS_{max} represents the maximum fitness value and FS_{min} represents the minimum fitness value achieved so far.
The tracing mode of the CSO method is described in terms of the movement of cats that is based on the outstanding hunting skills of cats. In tracing mode, the movement of cats is according to their velocities in each dimension and then updating their positions accordingly. The updated positions of cats and velocities are calculated using equations (4) and (5). These equations are:

$$V_{i,d}(t) = V_{i,d} + r_i c_i (X_{best,d} - X_{i,d}), d = 1,2,..,M \text{ --- (4)}$$

$$X_{i,d} = X_{i,d} + V_{i,d} \qquad \text{------ (5)}$$

Here, various terms are used for the position of the Cats like $X_{best,d}$ is the best position of a cat in d-dimensional space, $X_{i,d}$ is the position of Cat_i , $V_{i,d}$ is the velocity of c_i. r_i is a random value between [0, 1], c_i is the acceleration coefficient that extends the Cats velocity to move into solution space, which is set to 2.0, and t is the iteration number. Mixture Ratio (MR) is used to combine the two modes in the algorithm that is seeking mode and tracing mode and to determine the ratio of Cats in the modes. The control variable MR is set to 1%-3% that determines the position a Cat which is either seeking or tracing mode. It means that at any instance, 10% to 30% of the Cats are in tracing mode, and the rest of the Cats are in seeking mode. Here local search refers to tracing mode and global search refers to seeking mode. Cats spend most of the time in resting mode (seeking mode), so, the MR value should be a tiny value to show their behavior in the real world. There is a need to put the control check on the velocity of Cat for every dimension value so that velocities must be in the range and have not crossed maximum. This control check is added through inertia weight (w) for which the optimum value must be between [0.4-0.9]. The optimum solution will be available somewhere between the values.

ACO Algorithm: ACO has the ability of adapting changes continuously when the requirements change dynamically. Ants can find a high-quality solution in a search space and they share their knowledge in the form of pheromone update strategy and solve the problem efficiently. Further, ACO can be utilized in solving the load balancing problem that results in reduced computational time. The initial pheromone level is set first as 0.1. Initial pheromone value lies between two nodes that is VMi and VMj. After first iteration, this pheromone level is globally updated. Each ant 'k' moves from current node i (VM) to next node j (VM) by calculating the probability of ρ_{ij}^{k} of crossing the edge using the following equation

$$\rho_{ij}^{k} = \frac{(\tau_{ij})^{\alpha}(n_{ij})^{\beta}}{\sum_{1}^{n}(\tau_{ij})^{\alpha}(n_{ij})^{\beta}} \quad \text{------ (6)}$$

The probability ρ_{ij}^{k} from node i to node j depends upon two parameters that are pheromone level τ_{ij} and desirability of moves from node i to j, which is denoted by n_{ij} α and β are used to control the influence of τ_{ij} and n_{ij}.

The amount of pheromone shows the type of nodes (VM), the ant is searching. A greater amount of pheromone along with trailing path shows that the target node is overloaded so the ant will try to find another path with less amount of pheromones, that is after encountering an overloaded node it will find the underloaded node and assign a task to that node. In ant colony optimization, a pheromone is updated locally and globally. Local pheromone is updated by using equation (7)

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \rho\tau_{ij}^{0} \quad \text{------- (7)}$$

Where τ_{ij} is the pheromone level from node i to node j, when each ant traverses an edge ij, ρ is constant pheromone evaporation coefficient and τij is the initial pheromone level on edge ij. Second level of pheromone is global pheromone which takes place at the end of each iteration when all k ants have constructed the paths. The global pheromone level is computed using equation (8).

$$\tau_{ij} = (1-\rho)\tau_{ij} + \frac{\Delta\tau_{ij}}{L^{k}} \quad \text{------ (8)}$$

where 'm' is number of ants and $\Delta\tau_{ij}^{k}$ is the amount of pheromone deposited by ant k at edge ij in one iteration. L^{k} is the length of the trail t_i that k ant constructed. Large value of $\Delta\tau_{ij}$ increases the amount of pheromone level on each edge of the constructed paths as the time passes and is computed as:

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \quad \text{--------- (9)}$$

Where, $\Delta\tau_{ij} = \frac{1}{Completion\ \ Time}$ --------- (9) $CompletionTime_{(t_i,m_j)} = StartTime_i - EstTime_{(t_i,m_j)}$ ------ (10)

The completion time is computed using equation (10), as start time of the task is depending upon the completion time of task that is previously assigned to the respective machine. This time is helpful in load balancing. Here, $StartTime_i$ is assigned randomly to the task i, when the machine is available, and $EstTime_{(t_i,m_j)}$ is the time estimate to complete the task i at machine j.

IV. RESULTS

In this section, the evaluation of result analysis is divided into two major parts. In the first part, the classification performance FTF SVM in the cloud is evaluated. In the second part, as output of the SVM is fed into load balancing stage, the performance of QoS metrics on load balancing in the cloud environment are evaluated.

4.1 Accuracy Performance of FTFSVM The accuracy of the developed algorithm FTFSVM is evaluated, which shows its average performance taken together from audio, video, image and text. Performance metrics comprising of accuracy, precision and recall are used to provide the classifier performance. On average, highest classification performance is observed in the developed classification model. This shows that FTFSVM is classifying files quite accurately which will have a huge impact when used in scheduling.

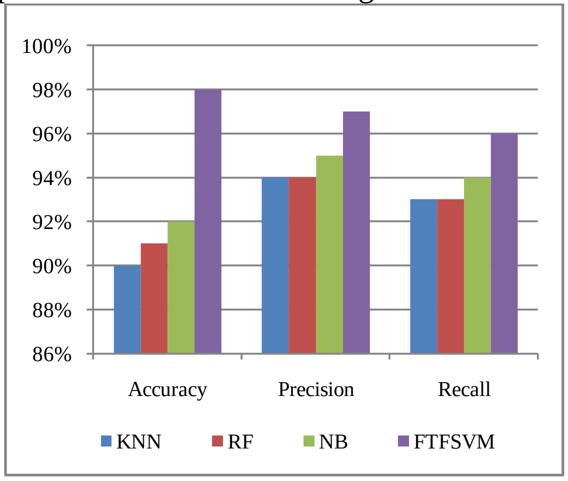

Fig. 2: Classification Performance Analysis of FTFSVM

In Fig. 2 some of the well-known classifiers are considered such as Random Forest (RF), Naïve Bayes (NB) and K-Nearest Neighbor (K-NN) in which their classification performance is compared with the FTFSVM classifier of the present hybrid model. It can be seen that the overall performance of FTFSVM is better as compared to other classifiers.

4.2 Evaluation of QoS metrics

Total of six QoS metrics are compared, such as SLA violations, throughput time, response time, migration time, energy consumption and average execution time with the base line models such as ACO (Ant Colony Optimization) and CSO (Cat Swarm Optimization). There is a total of 60,000 tasks on which evaluations are performed. All algorithms are implemented in CloudSim 4.0 with the same configuration and environmental setting to make the results reliable.

In this case, SLA violation occurs if VM takes more time than allotted CPU time. SLA violations against varying number of tasks and VMs for all baselines are shown in Figure 3. It is observed that till 100 VMs, no major change in SLA violations is seen in the graph which becomes more obvious after 500 and above VMs. The least number of SLA violations confirmed that optimized hybrid load balancing model has performed fewer migrations and avoiding complexity.

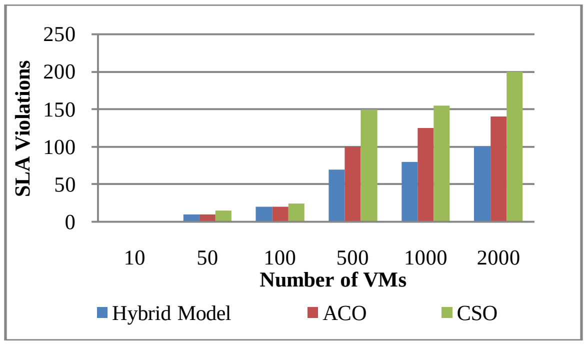

Fig. 3: Performance analysis on SLA Violations

Performance analysis of hybrid model on Energy consumption is depicted in Fig. 4 in which comparative analysis is performed for baseline models.

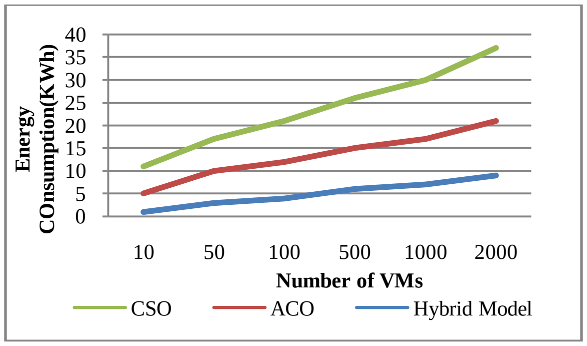

Fig. 4: Performance analysis on Energy Consumption

From the very start, a significant difference is seen which continues till 2000 VMs. After a gradual increase in the number of tasks from 1000 to 2000, few of the baselines such as ACO and CSO start to consume slightly more energy which is further followed by a few other baselines. Likewise, the increase in VMs from 5 to 10, 50, 100, 500, 1000, and 2000 also results in an increase in energy consumption which keeps on getting increased with every additional VMs. In 2000 VMs, most of the baselines suffer from huge energy consumption leaving only DFTF with comparatively least consumption of energy.

Table 1 shows the performance analysis of QoS metrics. Here the hybrid Model is compared with the other two baseline models such as ACO and CSO in terms of some Qos metrics called throughput time, response time, migration time and average execution time. The QoS metrics are analyzed here with the Performance Percentages with respect to hybrid model.

Table 1: Performance analysis of QoS metrics

QoS metric	Hybrid Model	ACO	CSO
Average Throughput Time	23%	32%	43%
Response time	0.9%	36%	54%
Migration time	23%	34%	41%
Average execution time	21%	33%	45%

It can be seen that the Hybrid model has a total of 23% throughput time as compared to ACO 32% and CSO 43%. Similarly the percentage of Hybrid model performance with other models is obtained as response time of 0.9%, migration time of 23%, average execution time of 21%.

V. CONCLUSION

The conducted study has devised a hybrid approach in two phases. In the first phase, SVM is modified for making accurate classifications over several file formats such as audio, video, image, and text. The initially classified data class has shown best classification as compared to known classifiers such as NB, RF and K-NN over-developed validation metrics comprising of accuracy, precision, and recall. The output of the SVM data classification is a data class that is fed into optimized Hybrid machine learning model for load balancing. The optimized Hybrid machine learning model is a metaheuristic algorithm that exploits the original ACO and CSO objective function in such a way that it fits well for further reducing or improving scheduling time to some significant extent. The result analysis illustrated that the optimized Hybrid machine learning model was most successful in all QoS metrics such as SLA violations, throughput time, response time, migration time, energy consumption and average execution time and made huge improvements while outperforming them. Due to the reduced number of SLA violations, a hybrid approach (if consider energy-aware schemes) has improved energy consumption which is a big issue for data centers. Hybridization provides better optimization which helps in achieving optimum resource utilization.

VI. REFERENCES

[1] Cheng, J. Li and S. Nazarian, "Drl-cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers", 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 129-134, 2018.
[2] Chunlin Li, Yun Chang Liu and Xin Yan, "Optimization-based resource allocation for software as a service application in cloud computing", Journal of Scheduling, vol. 20, no. 1, pp. 103-113, 2017.
[3] A. Khosravi, L. L. Andrew and R. Buyya, "Dynamic vm placement method for minimizing energy and carbon cost in geographically distributed cloud data centers", IEEE Transactions on Sustainable Computing, vol. 2, no. 2, pp. 183-196, 2017.
[4] Venkateshwarlu Velde, B. Rama, “An advanced algorithm for load balancing in cloud computing using fuzzy technique”, 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), 2017
[5] Ronak R. Patel and Swachil J. Patel, "Improved Ga Using Population Reduction For Load Balancing In Cloud Computing", 2016 Intl. Conference on Advances in Computing Communications and Informatics (ICACCI), Sept. 21¬24, 2016
[6] Akash Dave, Bhargesh Patel and Gopi Bhatt, "Load Balancing In Cloud Computing Using Optimization Techniques: A Study", IEEE, 2016.
[7] Esma Insaf Djebbar and Ghalem Belalem, "Tasks scheduling and resource allocation for high data management in scientific cloud computing environment", Lecture Notes in Computer Science, pp. 16-27, 2016.
[8] Abdul Razaque, Nikhileshwara Reddy Vennapusa, ,Nisargkumar Soni, Guna Sree Janapati and Khilesh Reddy V., “Task Scheduling in Cloud Computing”, 2016 IEEE Long Island Systems,Applications and Technology Conference (LISAT).

[9] BhawnaMallick and ReenaPanwar ,IEEE, ”Load Balancing in Cloud Computing Using Dynamic Load Management Algorithm” 2015International Conference on Green Computing and Internet of Things (ICGCloT), 2015
[10] Gao, Ren, and Juebo Wu. "Dynamic Load Balancing Strategy for Cloud Computing with Ant Colony Optimization." Future Internet 7.4 (2015): 465- 483.
[11] Ekta Gupta and VidyaDeshpande. "A Technique Based on Ant Colony Optimization for Load Balancing in Cloud Data Center." Information Technology (ICIT) 2014 International Conference on, IEEE, 2014
[12] NikitaHaryani,, DhanammaJagli, “Dynamic Method for Load Balancing in Cloud Computing“, IOSR Journal of Computer Engineering(IOSR-JCE), Volume 16, Issue 4, Ver. IV, Jul, Aug. 2014
[13] Shubham Mittal, and Avita Katal,” An Optimized Task Scheduling Algorithm in Cloud Computing”, IEEE 6th International Conference on Advanced Computing, 2016.
[14] Atul Vikas Lakra, and Dharmendra Kumar Yadav, “Multi-Objective Tasks Scheduling Algorithm for Cloud Computing Throughput Optimization”, Procedia Computer Science 48,107 – 113, International Conference on Intelligent Computing, Communication & Convergence, ICCC-2015 .
[15] Mrs.S.Selvarani, and Dr.G.Sudha Sadhasivam, “Improved CostBased Algorithm For Task Scheduling In cloud Computing” , 2010 IEEE International Conference on Computational Intelligence and Computing Research.