



Article

Machine-Learning-Based Approach for Virtual Machine Allocation and Migration

Suruchi Talwani ¹, Jimmy Singla ¹, Gauri Mathur ¹, Navneet Malik ¹, N. Z. Jhanjhi ^{2,*}, Mehedi Masud ³ and Sultan Aljhdali ³

¹ School of CSE, Lovely Professional University, Phagwara 144001, India

² School of Computer Science (SCS), Taylor's University, Subang Jaya 47500, Malaysia

³ Department of Computer Science, College of Computers and Information Technology, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia

* Correspondence: noorzaman.jhanjhi@taylors.edu.my

Abstract: Due to its ability to supply reliable, robust and scalable computational power, cloud computing is becoming increasingly popular in industry, government, and academia. High-speed networks connect both virtual and real machines in cloud computing data centres. The system's dynamic provisioning environment depends on the requirements of end-user computer resources. Hence, the operational costs of a particular data center are relatively high. To meet service level agreements (SLAs), it is essential to assign an appropriate maximum number of resources. Virtualization is a fundamental technology used in cloud computing. It assists cloud providers to manage data centre resources effectively, and, hence, improves resource usage by creating several virtual machine (VM) instances. Furthermore, VMs can be dynamically integrated into a few physical nodes based on current resource requirements using live migration, while meeting SLAs. As a result, unoptimised and inefficient VM consolidation can reduce performance when an application is exposed to varying workloads. This paper introduces a new machine-learning-based approach for dynamically integrating VMs based on adaptive predictions of usage thresholds to achieve acceptable service level agreement (SLAs) standards. Dynamic data was generated during runtime to validate the efficiency of the proposed technique compared with other machine learning algorithms.

Keywords: machine learning; virtual machine; migration; allocation; cloud computing



Citation: Talwani, S.; Singla, J.; Mathur, G.; Malik, N.; Jhanjhi, N.Z.; Masud, M.; Aljhdali, S. Machine-Learning-Based Approach for Virtual Machine Allocation and Migration. *Electronics* **2022**, *11*, 3249. <https://doi.org/10.3390/electronics11193249>

Academic Editors: Seokjoo Shin and Juan-Carlos Cano

Received: 6 September 2022

Accepted: 30 September 2022

Published: 9 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

By enhancing isolation between application resource usage, allocation, and management, virtualization offers the prospect of improved efficiency in cloud data centres [1]. One of the most intriguing aspects of virtualization is live migration. As a result, virtualization becomes more appealing. Operating virtual machines (VMs) can be relocated effortlessly between physical hosts via live migration. Service providers offer hosting of available applications to ensure quality of service, which is defined according to a service level agreement (SLA), and, in most circumstances, expressed in terms of application availability.

Cloud computing has become a requirement for today's IT businesses. The exchange of data and information over the Internet is widespread. The applications generate a large amount of data that must be kept, and which is constantly transmitted over the Internet. To address the issue of storage space, numerous kinds of big data are gathered and saved on a cloud server; as a result, anyone can access the required data from any place via the Internet. Amazon, IBM, Google and Microsoft have established themselves as major cloud service providers. According to one study, the average resource utilization of a cloud data centre is about thirty percent. However, idle energy consumption can reach seventy percent of the total data centre power consumption. As a result, much energy is lost unnecessarily [1]. Furthermore, the server cannot be left with a light workload [2]. As a result, numerous

approaches have been proposed for identifying uncluttered and overloaded machines to save energy. These approaches include migration, virtualization, job mapping and consolidation. The virtualization technology concept is used in many studies. The power management problem has been overcome using the notion of virtualization, which enables overburdened servers to spread their workload to multiple virtual machines without impacting service performance [3]. As a result, efficient VM usage has played an essential role in providing cost-effective services to clients [4]. Cloud computing is becoming increasingly valuable for storing enormous amounts of data in today's world. Virtual machines (VMs) provide virtual application resources to a large number of users at once. With an application window on a desktop, VM allows any organization operating system to function as if it were a completely independent computer. How VM works is determined by the historical data utilized by the server. Flexibility and low cost are the primary advantages of VM. Virtualization technology assists in balancing the load on the server by deploying applications to machines that are less or moderately loaded. This procedure entails offsetting several virtual computers to relieve the workload on the actual device. The key elements which need to be handled intelligently, and in a balanced way, are service level agreements (SLA), energy consumption, hosting and the number of migrations. Any imbalance between the given metrics can degrade the performance and service quality of the data centre's services. This means that underutilized and over-utilized equipment should be easily distinguishable [5]. In this regard, investigating real and virtual machines is a time-consuming operation. As a result, machine learning architecture is a viable option. Artificial intelligence and swarm intelligence have been used in combination to handle the critical parameters which affect service quality [6,7]. The authors of the latter studies provided a hybrid optimization strategy-based intelligent VM allocation and migration framework, which was inspired by similar research [8–10]. ABC (Ant Bee Colony) and CS (Cuckoo Search), two well-known bio-inspired algorithms, were also integrated to find the best virtual machine for load balancing, while meeting energy needs and addressing SLA limitations. An overview of virtual machine allocation and migration techniques is provided in Figure 1.

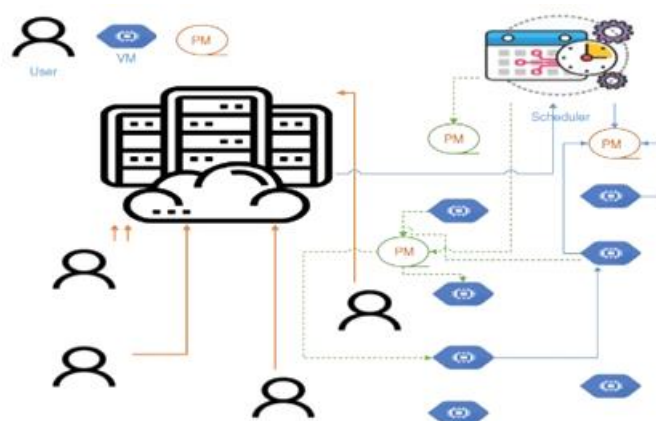


Figure 1. Virtual Machine Allocation and Migration Technique.

The main contributions of this research which differentiate it from existing research are:

1. Here, an optimization methodology is designed which assists in the development of a green computing environment which results in an eco-friendly use of resources.
2. Performance evaluation is undertaken in terms of different performance parameters, such as used energy consumption, the total number of migrations and SLA violations. Here, these parameters have better values.
3. A novel machine-learning-based approach is introduced to dynamically integrate VMs based on adaptive predictions of usage thresholds to achieve acceptable service level agreements (SLAs).

4. Dynamic data is generated during runtime to validate the efficiency of the proposed technique compared with other machine learning algorithms.
5. The proposed approach is also useful in cases of dynamic workload, i.e., changing workloads over time. The dynamic data help to create a dynamic architecture as the workload continuously changes in a cloud environment. Thus, the most efficient virtual machine can be selected depending on the dynamics of the load and energy.

A novel hybrid technique, comprising a combination of a swarm intelligence algorithm and a machine learning classifier, is proposed for the allocation and migration of virtual machines. In the literature, the focus is either on swarm intelligence algorithms or machine learning classifiers for virtual machine allocation and migration, but the proposed hybrid technique provides better results compared to existing techniques described in the literature.

This article is organized as follows:

Section 2 discusses related work, including existing studies addressing virtual machine allocation and migration. Section 3 describes the study methodology. Section 4 details the results and provides related discussion. The paper's conclusions and suggestions for future research are presented in Section 5.

2. Related Work

It has been found that switching between servers and computing in cloud data centres consumes a significant amount of energy. Although VM migration consumes energy, it improves the data centre's overall execution efficiency. The allocation of machines is the most critical aspect of virtualization, followed by migration. Researchers have used various methodologies to determine the optimum VM to enable minimal migration to save energy throughout the virtualization process. Several applications and factors affect the VM migration process. Some of the problems that have been addressed concerning VM migration include economical VM migration, the heterogeneous nature of cloud resources, system workload, and memory. Another significant challenge in VM migration is security risk [11]. Yakhchi used the Cuckoo Search (CS) optimization technique to detect overused hosts, resulting in more efficient resource management. The minimum migration time policy was then implemented to transfer machines from being overused to underused or moderately used [12]. A simulation analysis revealed that, with a low number of migrations, the average value of energy usage was 19.95 kWh. Dhanoa and Khurmi presented a hybrid approach to accomplish power-efficient VM migration. This project included SLA violations, energy usage, virtual machine migrations and a genetic algorithm that assisted in improving the response time. The primary intent of the study was to optimize live migrations for reduced power consumption in various scenarios.

In contrast to the basic algorithm, simulation investigation revealed that the hybrid VM allocation saved 72 percent of power when delivering quality service [13,14]. Jiang [15] suggested a data ABC (Ant Bee Colony) energy model for making VM consolidation decisions based on global optimization. This model depends on the usage rates of the GPU and the CPU. The study considered two policies for live VMs: the first was called VM selection, and the second VM allocation. Compared to existing models, the ABC model saved 25% to 35% energy [15]. Perumal presented a fuzzy hybrid bio-inspired meta-heuristic strategy to solve the problem of VM placement. The study focused on power usage, resource wastage, and virtual machine placement in the cloud. The experimental reports revealed that the hybrid algorithm outperformed the ACO, Firefly, MMAS, and FFD algorithms [16]. Barlaskar et al. presented Enhanced Cuckoo Search (ECS), a VM placement algorithm inspired by the Cuckoo Search (CS) algorithm, which helped to address the energy consumption problem in cloud data centres. Planet Lab was used to track the workload of the ECS algorithm. The results of comparing ECS to the Optimized Firefly Search (OFS) algorithm, the Ant Colony (AC) method and the Genetic Algorithm (GA) showed that the amount of power utilized by ECS was low, but SLA, as well as VM migration performance, was maintained [16,17].

Ruan [17] presented a performance-to-power ratio-based VM allocation and migration technique. This ratio was calculated using machine usage levels that were sampled. This data was utilized to ensure that the computers were operating power-efficiently without sacrificing performance. A comparison analysis revealed that the energy consumption decreased effectively, with a percentage of 69.31 using the VM selection and allocation framework [17]. This was affected by reducing the number of migrations and shutdowns, while maintaining minimum performance loss. The Cuckoo Search with Firefly (CS-FA) method was developed by Kumar et al. utilizing cloud computing for load balancing. It was used to determine a virtual machine's capacity and load. The CSFA algorithm's primary goal was to complete two tasks: first, to identify the optimal virtual machines (VMs) to allot the work, and second, to migrate overloaded VM's tasks to under-loaded VM's tasks. The CSFA algorithm was able to migrate two tasks when the numbers of tasks were equivalent to 40, whereas the present technique can migrate six tasks [18,19]. Karthikeyan [20] used hybrid optimization to overcome the energy restrictions. These researchers combined the Bat and Artificial Bee Colony (ABC) algorithms, and then used the Nave Bayes classifier to assign the VM to meet the energy requirements [20]. To achieve optimal resource use while decreasing energy consumption, Jangra combined an artificial neural network (ANN) and the Cuckoo Search algorithm. The authors sorted the virtual machines depending on the CPU usage, which was calculated using the modified best-fit decreasing (MBFD) method. When comparing the suggested CS-based method to the existing SI-technique-based approach, it was discovered that the proposed CS-based approach consumed 13.15 percent less energy [21]. Talwani and Singla [22] proposed an Enhanced Artificial Bee Colony (E-ABC) algorithm to distribute the load equally to multiple VMs. The E-ABC algorithm decreased the VM workload to save energy by migrating the task to an underloaded host from an overloaded VM. According to simulation results, the E-ABC algorithm fared better with respect to scalability and the occurrence of fewer VM migrations. Compared to the present proposal, the E-ABC approach saved 15 to 17 percent of the energy while reducing the number of migrations by 10% [23–26].

The proposed approach is different from existing approaches in terms of the optimization methodology used for the development of a green computing environment, the machine-learning-based approach to dynamically integrate VMs based on adaptive predictions of usage thresholds, and the dynamic data to be generated during runtime to validate the efficiency of the technique [27,28].

The advantages of existing approaches described in the literature are reduced energy consumption and optimized resource allocation during virtual machine allocation and migration; however, these parameters may be further improved by using swarm intelligence techniques with machine-learning classifiers. There are some disadvantages as well. In some of the existing studies, there is no verification of the overloading of physical machines. Some studies reported a borderline problem. Optimal virtual machine placement and migration still represent an NP hard problem [29,30].

3. Proposed Methodology

The concept of the deployment of virtual machines resolves problems associated with the use of physical machines. However, its application also has its problems. The traditional notion is that a virtual machine is created and powered on when the PM is overloaded. Buya (co-owner: Cloud Sim) saw this as a severe problem and developed the migration minimization (MM) method as a VM transfer policy and the MBFD (modified best-fit decreasing) method as an allocation policy. Certain drawbacks were identified in allocation policies, such as misallocation and unnecessary energy consumption, which contribute to the problem of global warming. This research proposes optimization methods that contribute to the development of green computing environments by increasing resource use and reducing energy consumption.

The primary areas of this proposal are allocation policy optimization and the mutual validation of migration systems to optimize QoS. Evaluation of the performance of the proposed approach involved consideration of the following parameters:

- the consumption of energy;
- the total number of migrations;
- SLA violations.

The steps involved in undertaking the research are listed below.

3.1. Allocation Process

Definition 1. In the cloud environment, an active architecture is the allocation mechanism. An allocation $A(H, At)$ where At is the total allocated tasks and H denotes the total number of hosts.

Problem Definition

For $A(H, At)$, if sufficient resources are present, H_{next} for “ μ ” load utilisation should be less than the load utilisation of the network; to handle the incoming tasks, then host selection in the A is feasible. If the satisfaction of the considered parameter, i.e., load utilisation, is not fulfilled, then the structure must search for a node which depends on the lemma.

Lemma 1. Includes two elements, such as false and true, which are further represented by 0 and 1, respectively. These are shown in Table 1.

Table 1. Representation of elements.

1	If $f(Search_{Cost}) < f(Wait_{Cost})$
0	Otherwise

The usage of virtual machine resources is requested to evaluate the allocation process. The rules below are followed in the allocation procedure:

- The sorting of the VMs should be undertaken based on the requirements of CPU utilization.
- Check the availability of resources on the physical machine (PM).
- If the physical machine meets the resource requirements, assign VM to PM.
- PM resources are reduced according to demand coverage.

A significant problem faced by the modified best-fit decreasing algorithm is borderline issues, which result in virtual machine migration and energy consumption. Hence, avoiding false migration and reducing the migration number is necessary.

The steps for VM allocation to PM using the MBFD algorithm are as follows:

Step 1: Input Parameters: HostList, VMList Output: VMs allocated to PMs

Step 2: VMList. sort_in DECREASING_CPU_Utilization()

Step 3: For each VM in VMList do

Step 4: Min_Power ← MAX

Step 5: Allocated_Host ← NULL

Step 6: For each host in HostList do

Step 7: if host has enough resources for VM then

Step 8: Power ← estimate_Power(host, VM)

Step 9: if Power < Min_Power then

Step 10: Allocated_Host ← host

Step 11: Min_Power ← Power

Step 12: if Allocated_Host ≠ NULL then

Step 13: Allocate VM to Allocated_Host

Step 14: Return allocation list

3.2. Transition between Physical Machine and Virtual Machine

Physical machines are utilized by the cloud framework to deal with the incoming workload. A physical medium that includes central processing units, process utilities and random-access memory is known as a physical machine. By considering the capabilities of physical machines, virtual machines are allocated to them for faster execution of parallel processing. A number of computing algorithms, such as MBFD, have been implemented for the allocation process. In the migration process, the virtual machines are migrated from one PM to another PM when observed to be unsuitable. Massive energy consumption occurs in the migration process; therefore, to reduce the energy consumption, it is also essential to decrease the migrations. The primary reason for migration is misallocation. Minimization of Migration (MM) is an effective algorithm for preventing migration. The service level agreement (SLA) is violated if the service provider fails to offer the promised service within the specified time interval, also known as SLA-V. It is also increased by over-exhausting the load, leading to considerable consumption of energy. Green-computing-based frameworks are also introduced, which help to reduce energy consumption.

3.2.1. Finding Over-Utilized, Normal-Utilized and Under-Utilized Hosts Using CPU

Threshold policy steps are as follows:

- Step 1: For each host in the host list,
- Step 2: Obtain CPU utilization
- Step 3: Set minimum and maximum threshold values for CPU utilization
- Step 4: If obtained CPU utilization < minimum threshold value, then
- Step 5: Host is under-utilized and all VMs of that host are migrated
- Step 6: Switch off that host
- Step 7: If obtained minimum threshold value < CPU utilization < maximum threshold value, then
- Step 8: Host is normal-utilized
- Step 9: If obtained CPU utilization > maximum threshold value, then
- Step 10: Host is over-utilized
- Step 11: Migrate some VMs from that host to new PMs

3.2.2. Selection of VMs for Migration from Over-Utilized Host Using Enhanced Artificial

Bee Colony Algorithm is as follows:

- Step 1: For every overloaded PM,
- Step 2: Do
- Step 3: Employ policy of virtual machine selection
- Step 4: Select the adequate virtual machine for migration using enhanced artificial Bee Colony approach
- Step 5: Addition of VM to migration list
- Step 6: Reallocate the migrated VM to new PM using MBFD algorithm.

3.2.3. Analysis of Post-Migration Process Using Support Vector Machine Algorithm

The training process is as follows:

- Step 1: Set input parameters: Bf, Ids. Bf, Pre-trained-structure (Pts), Label-set (Ls) \in {Above Post Performance (APP), Below Post Performance (BPP)}. The Pts contain the data from previous learning for 1000 simulations for the migration process of SVM that is available in the set. Bf is the created feature during the execution of the Energy-Aware ABC algorithm. Ids contain the identity number of VMs in the array form.
- Step 2: Initialize support vector machine algorithm with Bf, Ls.
- Step 3: Kernel set types = {"Linear", "Polynomial", "RBF"}
- Step 4: Trained-SVM (T-SVM) = Train (Bf, Ls, Kernel set)
- Step 5: Test the SVM algorithm with different validation ratios {0.15, 0.30, 0.50}.
- Step 6: Embed T-SVM to Pts.

Each allocation procedure follows the context awareness policy. With this policy, the system collects information about the environment and adapts accordingly. This policy assesses whether the unit's demand meets the fundamental standards or not. For instance, if the virtual machine requires 500 MB of RAM and the host has only 520 MB as available RAM, then the virtual machine is allocated to the host. However, in this case, the borderline issue is faced by the host, as it is left with only 20 MB of RAM. For smooth functioning, the real-time simulations are not dependent on the policy of content awareness. Run time entities must consume more resources than were requested at the allocation time. Hence, in this case, either wait until the required resource is unavailable and busy performing other tasks or migrate the virtual machines from their current host. This will lead to expenditure and network delay, which are unnecessary and incompatible with any procedure.

In real life scenarios, the proposed approach helps to improve utilization of resources, the load-balancing of processing nodes, the isolation of applications, fault tolerance in virtual machines, and to increase the portability of nodes and the efficiency of the physical server. A novel contribution of the proposal is the incorporation of swarm intelligence technique to address SLA violations and the number of migrations. Enhanced ABC is also proposed here to achieve optimal allocation of virtual machines.

4. Result and Discussion

The proposed methodology was compared with k-nearest neighbors (KNN) and decision tree classification algorithms concerning energy consumption, violations of SLA and the number of VM migrations. One of the most basic approaches for pattern classification is the k-nearest neighbors (KNN) method. This technique is based on proximity, to enable grouping of individual data points. When paired with past information, it has been used as a validated classification technology in many disciplines, producing significant results. Every unlabeled example in the training set is classified by the majority occurring label among its KNN. As a result, the distance utilized to locate nearest neighbors impacted its classification performance. Most of the KNN classifiers were not based on any statistical data regularities, and could be evaluated from a vast set of training examples with labels where prior knowledge is missing. Vector inputs were used to represent the Euclidean distances between examples utilized in many KNN classifiers to evaluate the similarities.

When a class was known in advance during the observation of the training sample, it was efficient to use a decision tree for the classification. Classes could be either hypothesis-based or user-provided in the learning sample. Let us assume a parent node and right and left child nodes which are, respectively, nodes of the parent node. Consider a training sample whose variable matrix includes the observations and a large number of variables and a class vector consisting of observations with integers of the class. The splitting rule is used to build a classification tree. It splits the learning sample into smaller sections.

Support vector machines (SVMs) could discover the hyper-plane of the maximum margin and enable nonlinear classification. SVM is a supervised machine-learning-based algorithm that is used for classification. The following is a mathematical representation of the data points (or training set D), $D = \{(x_1, y_1), (x_2, y_2), (x_n, y_n)\}$.

Where the n -dimensional real vector is represented by x_i ; its values will be either 1 or -1 , with reference to the class with which the particular point is associated.

During the training process, the classification function $F(x)$ is calculated by the SVM, and the form of the function is $F(x) = w \times (x - b)$, where w and b are the weights and bias, respectively. To identify if the classifier classifies the training set into accurate classes, it is necessary to ensure that the output of the negative data point is always negative and the output of positive points is always represented in positive numbers.

A detailed comparison, in terms of the number of migrations, SLA violations and energy consumption, is shown in Tables 2–4, respectively.

Table 2. Comparison in terms of number of migrations.

Total Host Count	Total VM Count	Host to VM Ratio	No of Migrations (Proposed Work)	No of Migrations (KNN)	No of Migrations (DT)	No of Migrations (ABC)	No of Migrations (E-ABC)	No of Migrations (CS)	No of Migrations (CS-FA)
10	100	0.1	1	3	2	3	2	3	2
12	120	0.1	0	1	1	1	1	1	1
14	144	0.097222222	5	7	9	8	7	7	6
17	173	0.098265896	2	5	4	4	4	4	3
20	208	0.096153846	3	4	7	5	5	4	4
24	250	0.096	1	2	2	3	2	2	2
29	300	0.096666667	7	13	9	9	9	8	8
35	360	0.097222222	2	3	3	4	3	3	3
42	432	0.097222222	11	18	13	15	14	13	12
50	518	0.096525097	4	6	5	5	5	5	4
60	622	0.096463023	12	20	12	14	13	13	13
72	746	0.096514745	14	24	15	16	15	15	15
86	895	0.096089385	2	8	3	5	4	4	3
103	1074	0.095903166	15	17	15	16	15	15	15
124	1289	0.096198604	22	34	28	32	31	30	27
149	1547	0.096315449	9	13	11	12	11	11	10
179	1856	0.096443966	10	15	14	12	12	11	11
215	2227	0.096542434	29	31	31	30	30	30	30
258	2672	0.096556886	12	28	13	18	18	16	14
310	3206	0.096693699	8	11	10	11	10	10	10

Table 3. Comparison in terms of SLA violations.

Total Host Count	Total VMCount	Host to VM Ratio	SLA-V (Proposed Work)	SLA-V (KNN)	SLA-V (DT)	SLA-V (ABC)	SLA-V (E-ABC)	SLA-V (CS)	SLA-V (CS-FA)
10	100	0.1	0.117831305	0.192684183	0.495728135	0.4954489983	0.288375324	0.459656523	0.425256547
12	120	0.1	0.123501408	0.224489968	0.493648697	0.485296325	0.285631478	0.458525865	0.429658745
14	144	0.097222222	0.109418336	0.184520281	0.415923406	0.475152635	0.278585856	0.445256586	0.424456589
17	173	0.098265896	0.103696941	0.241277083	0.47391736	0.478585856	0.275858632	0.435652545	0.412523669
20	208	0.096153846	0.116399684	0.233451006	0.458978091	0.475265653	0.272586995	0.432565632	0.411258563
24	250	0.096	0.161862539	0.219948321	0.419979907	0.465252636	0.265896532	0.421256358	0.405236669
29	300	0.096666667	0.162199011	0.180132547	0.433464511	0.468585623	0.268541233	0.425866662	0.412523695
35	360	0.097222222	0.101470858	0.1909137	0.424089097	0.465236996	0.265554633	0.422243652	0.4123565322
42	432	0.097222222	0.175170848	0.258375394	0.492845224	0.452653325	0.262354123	0.412533652	0.4058742336
50	518	0.096525097	0.110376499	0.214693422	0.47950591	0.442568852	0.262542336	0.4125896322	0.400215236
60	622	0.096463023	0.154017005	0.209699641	0.415626302	0.432589633	0.2612525233	0.4121255663	0.402558963
72	746	0.096514745	0.164409902	0.249760245	0.402138486	0.4258533667	0.2602145632	0.4112586321	0.402845566
86	895	0.096089385	0.146467731	0.183922017	0.463598021	0.402152226	0.245252526	0.39852478	0.398542685
103	1074	0.095903166	0.17234055	0.18411073	0.429542177	0.3852526852	0.235252637	0.385241526	0.382536251
124	1289	0.096198604	0.112387853	0.254897348	0.482555724	0.365552881	0.225656233	0.385121527	0.356531251
149	1547	0.096315449	0.144353564	0.1776709	0.456545998	0.365232362	0.225896632	0.373591328	0.322536252
179	1856	0.096443966	0.15095052	0.177235918	0.441656345	0.323232562	0.223565896	0.366361529	0.312501253
215	2227	0.096542434	0.114923564	0.181371937	0.477201981	0.285256963	0.212536252	0.353265921	0.302525254
258	2672	0.096556886	0.1301144	0.239308452	0.451164842	0.265853332	0.204521252	0.345221212	0.288500255
310	3206	0.096693699	0.098748823	0.18488957	0.445684889	0.245856363	0.235252632	0.335241523	0.282536256

Improvement Analysis in Terms of Confusion Matrix

Improvement analysis was also performed in terms of the confusion matrix. A total of six confusion matrices were generated and compared with each other. All six matrices are shown in Table 5. In the case of KNN, ABC and CS, two cases in each were wrongly classified, and in the case of the decision tree, E-ABC and CS-FA, one case in each was wrongly classified, but using the proposed approach, all the cases were correctly classified.

Table 4. Comparison in terms of energy consumption.

Total Host Count	Total VM Count	Host to VM Ratio	Energy (Proposed Work)	Energy (KNN)	Energy (DT)	Energy (ABC)	Energy (E-ABC)	Energy (CS)	Energy (CS-FA)
10	100	0.1	14.31392	22.7238	44.78887	21.1258	20.1025	18.5025	17.2036
12	120	0.1	14.67201	16.62476	16.07147	21.1574	20.1047	18.2036	17.3025
14	144	0.097222	12.9972	18.92474	44.61832	20.1548	18.1258	18.5369	16.3026
17	173	0.098266	13.01527	18.20591	23.36902	22.1479	20.1720	20.2589	19.2014
20	208	0.096154	12.21091	13.00463	22.53962	20.1369	17.1596	18.3695	16.2589
24	250	0.096	11.47695	11.59025	21.9669	19.1589	16.1247	17.1478	15.6147
29	300	0.096667	15.22248	17.93753	17.39447	23.1025	20.1036	20.1258	19.3698
35	360	0.097222	11.20958	11.36541	26.32962	20.1987	16.1025	15.1025	14.2589
42	432	0.097222	13.11293	24.01095	38.13931	22.1002	18.5025	17.3025	16.1478
50	518	0.096525	12.59919	27.15201	23.93679	21.1687	17.2036	17.2325	15.1598
60	622	0.096463	12.02804	23.94889	20.94188	20.1598	16.2047	16.2589	15.1258
72	746	0.096515	13.76747	16.38431	29.35553	21.1478	15.2054	15.3025	16.3698
86	895	0.096089	12.14222	22.46474	25.75727	20.3698	15.1025	15.0123	16.0213
103	1074	0.095903	10.46098	25.069	24.94082	18.2589	14.2589	14.2563	15.2103
124	1289	0.096199	10.55519	14.15034	13.50405	18.3698	14.3698	14.2147	15.2365
149	1547	0.096315	11.28898	22.82299	17.16675	19.0213	15.2103	15.2145	15.9874
179	1856	0.096444	10.51275	27.59739	39.97436	18.2458	14.2369	14.3214	14.2154
215	2227	0.096542	11.40067	14.16637	26.94556	19.2369	15.2367	15.2146	15.2369
258	2672	0.096557	14.33465	26.12634	38.19942	18.2589	15.2147	15.2365	15.2103
310	3206	0.096694	12.23972	13.27008	44.19743	15.2154	13.2587	13.2134	13.2147

Table 5. Improvement analysis in terms of the confusion matrix.

KNN		
Class	0	1
0	33	00
1	02	09
DT		
Class	0	1
0	33	00
1	01	10
ABC		
Class	0	1
0	33	00
1	02	09
E-ABC		
Class	0	1
0	33	00
1	01	10
CS		
Class	0	1
0	33	00
1	02	09
CS-FA		
Class	0	1
0	33	00
1	01	10
Proposed Work		
Class	0	1
0	33	00
1	00	11

5. Conclusions and Future Scope

In this investigation, the performance of classification techniques was evaluated in terms of the false negative rate (FNR) and the true positive rate (TPR). The classifier output was considered as TPR if the VM migration was correctly classified into the class named ‘migrated’ due to high usage. Similarly, if the classifier output was considered an FNR, VM migration was classified as ‘not migrate’ from the current server.

KNN, ABC and CS could accurately identify VM migration in the dynamic data with 100 percent TPR and 81.8 percent FNR. The decision trees, E-ABC, CS-FA were able to accurately identify and predict VM migration in the dynamic data with 100 percent TPR and 90.9 percent FNR. The proposed methodology was able to accurately identify and predict VM migration in the dynamic data with 100 percent TPR and 100 percent FNR. It is clear from the confusion matrix that the performance of the proposed methodology based on SVM was better compared to KNN and the decision trees. In future, another set of similar optimization algorithms may be investigated to further improve the existing approach. Multi-class ML approaches may also be integrated for performance analysis against other state-of-the-art techniques. In the future, research may also be undertaken on resource constraints in the cloud data centre which is to be accessed by the cloud broker.

Author Contributions: Conceptualization, S.T. and J.S.; methodology, S.T.; Software, G.M.; validation, N.M.; formal analysis, N.J.; investigation, S.T.; resources, M.M.; writing, S.T.; visualization, M.M.; supervision, S.A. All authors have read and agreed to the published version of the manuscript.

Funding: Taif University Researchers Supporting Project Number (TURSP-2020/73), Taif University, Taif, Saudi Arabia.

Data Availability Statement: The data is available on request.

Conflicts of Interest: All authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Buyya, R.; Yeo, C.S.; Venugopal, S.; Broberg, J.; Brandic, I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.* **2009**, *25*, 599–616. [\[CrossRef\]](#)
2. Zhang, X.; Wu, T.; Chen, M.; Wei, T.; Zhou, J.; Hu, S.; Buyya, R. Energy-aware virtual machine allocation for cloud with resource reservation. *J. Syst. Softw.* **2019**, *147*, 147–161. [\[CrossRef\]](#)
3. Moon, Y.; Yu, H.; Gil, J.M.; Lim, J. A slave ants-based ant colony optimisation algorithm for task scheduling in cloud computing environments. *Hum. Cent. Comput. Inf. Sci.* **2017**, *7*, 28. [\[CrossRef\]](#)
4. Keller, R.; Hafner, L.; Sachs, T.; Fridgen, G. Scheduling flexible demand in cloud computing spot markets. *Bus. Inf. Syst. Eng.* **2020**, *62*, 25–39. [\[CrossRef\]](#)
5. Arani, M.G.; Rahmanian, A.A.; Shamsi, M.; Kenari, A.R. A learning-based approach for virtual machine placement in cloud data centers. *Int. J. Commun. Syst.* **2018**, *31*, e3537. [\[CrossRef\]](#)
6. Manoharan, H.; Teekaraman, Y.; Kshirsagar, P.R.; Sundaramurthy, S.; Manoharan, A. Examining the effect of aquaculture using sensor-based technology with machine learning algorithm. *Aquac. Res.* **2020**, *51*, 4748–4758. [\[CrossRef\]](#)
7. More, S.; Singla, J. A Study on Automated Grading System for Early Prediction of Rheumatoid Arthritis. In Proceedings of the 6th International Conference on Communication and Electronics Systems (ICCES), Coimbatre, India, 8–10 July 2021; pp. 1293–1300. [\[CrossRef\]](#)
8. Sundaramurthy, S.C.; Kshirsagar, P. Prediction and Classification of Rheumatoid Arthritis using Ensemble Machine Learning Approaches. In Proceedings of the International Conference on Decision Aid Sciences and Application (DASA), Sakheer, Bahrain, 8–9 November 2020; pp. 17–21. [\[CrossRef\]](#)
9. Kshirsagar, P.R.; Manoharan, H.; Al-Turjman, F.; Kumar, K. Design and Testing of Automated Smoke Monitoring Sensors in Vehicles. *IEEE Sens. J.* **2022**, *22*, 17497–17504. [\[CrossRef\]](#)
10. Kshirsagar, P.R.; Akojwar, S.G. Prediction of neurological disorders using optimised neural network. In Proceedings of the 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), Paralakhemundi, India, 3–5 October 2016; pp. 1695–1699. [\[CrossRef\]](#)
11. Ahmad, R.W.; Gani, A.; Hamid, S.H.A.; Shiraz, M.; Xia, F.; Madani, S.A. Virtual machine migration in cloud data centers: A review, taxonomy, and open research issues. *J. Supercomput.* **2020**, *71*, 2473–2515. [\[CrossRef\]](#)
12. Yakhchi, M.; Ghafari, S.M.; Yakhchi, S.; Fazeli, M.; Patooghi, A. Proposing a load balancing method based on Cuckoo Optimization Algorithm for energy management in cloud computing infrastructures. In Proceedings of the 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO), Istanbul, Turkey, 27–29 May 2015; pp. 1–5.

13. Dhanoa, I.S.; Khurmi, S.S. Power efficient hybrid VM allocation algorithm. *Int. J. Comput. Appl.* **2015**, *127*, 39–43.
14. More, S.; Singla, J. A generalised deep learning framework for automatic rheumatoid arthritis severity grading. *J. Intell. Fuzzy Syst.* **2021**, *41*, 7603–7614. [[CrossRef](#)]
15. Jiang, J.; Feng, Y.; Zhao, J.; Li, K. Data ABC: A fast ABC based energy-efficient live VM consolidation policy with data-intensive energy evaluation model. *Future Gener. Comput. Syst.* **2017**, *74*, 132–141. [[CrossRef](#)]
16. Perumal, B.; Aramudhan, M.; Saravanaguru, R.K. Fuzzy Bio-Inspired Hybrid Techniques for Server Consolidation and Virtual Machine Placement in Cloud Environment. *Cybern. Inf. Technol.* **2017**, *17*, 52–68. [[CrossRef](#)]
17. Ruan, X.; Chen, H.; Tian, Y.; Yin, S. Virtual machine allocation and migration based on performance-to-power ratio in energy-efficient clouds. *Future Gener. Comput. Syst.* **2019**, *100*, 380–394. [[CrossRef](#)]
18. More, S.; Singla, J.; Song, O.Y.; Tariq, U.; Malebary, S. Denoising medical Images using deep learning in IoT environment. *Comput. Mater. Contin.* **2021**, *69*, 3127–3143. [[CrossRef](#)]
19. Kumar, K.; Raguathan, T.; Vasumathi, D.; Prasad, P. An Efficient Load Balancing Technique based on Cuckoo Search and Firefly Algorithm in Cloud. *Int. J. Intell. Eng. Syst.* **2020**, *13*, 422–432. [[CrossRef](#)]
20. Karthikeyan, K.; Sunder, R.; Shankar, K.; Lakshmanaprabu, S.K.; Vijayakumar, V.; Elhoseny, M.; Manogaran, G. Energy consumption analysis of Virtual Machine migration in cloud using hybrid swarm optimisation (ABC–BA). *J. Supercomput.* **2020**, *76*, 3374–3390. [[CrossRef](#)]
21. Jangra, S. Cuckoo-Neural Approach for Secure Execution and Energy Management in Mobile Cloud Computing. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)* **2021**, *12*, 654–663.
22. Talwani, S.; Singla, J. Enhanced Bee Colony Approach for reducing the energy consumption during VM migration in cloud computing environment. In Proceedings of the 1st International Conference on Computational Research and Data Analytics (ICCRDA 2020), Rajpura, India, 24 October 2020; Volume 1022, p. 012069.
23. Pande, S.K.; Panda, S.K.; Das, S.; Sahoo, K.S.; Luhach, A.K.; Jhanjhi, N.Z.; Alroobaea, R.; Sivanesan, S. A resource management algorithm for virtual machine migration in vehicular cloud computing. *Comput. Mater. Contin.* **2021**, *67*, 2647–2663.
24. Shafiq, D.A.; Jhanjhi, N.Z.; Abdullah, A. Load balancing techniques in cloud computing environment: A Review. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 3910–3933. [[CrossRef](#)]
25. Shafiq, D.A.; Jhanjhi, N.; Abdullah, A. Machine Learning Approaches for Load Balancing in Cloud Computing Services. In Proceedings of the 2021 National Computing Colleges Conference (NCCC), Taif, Saudi Arabia, 27–28 March 2021; pp. 1–8. [[CrossRef](#)]
26. Shafiq, D.A.; Jhanjhi, N.Z.; Abdullah, A.; Alzain, M.A. A Load Balancing Algorithm for the Data Centres to Optimize Cloud Computing Applications. *IEEE Access* **2021**, *9*, 41731–41744. [[CrossRef](#)]
27. Babbar, H.; Rani, S.; Masud, M.; Verma, S.; Anand, D.; Jhanjhi, N. Load balancing algorithm for migrating switches in software-defined vehicular networks. *Comput. Mater. Contin.* **2021**, *67*, 1301–1316. [[CrossRef](#)]
28. Li, L.; Dong, J.; Zuo, D.; Wu, J. SLA-Aware and Energy-Efficient VM Consolidation in Cloud Data Centers Using Robust Linear Regression Prediction Model. *IEEE Access* **2019**, *7*, 9490–9500. [[CrossRef](#)]
29. Azab, S.S.; Hefny, H.A. Swarm intelligence in semi-supervised classification. *arXiv* **2017**, arXiv:1706.00998.
30. Naik, B.B.; Singh, D.; Samaddar, A.B. FHCS: Hybridisedoptimisation for virtual machine migration and task scheduling in cloud data center. *IET Commun.* **2020**, *14*, 1942–1948. [[CrossRef](#)]