symmetry

MDPI

Article

PSO-Based Ensemble Meta-Learning Approach for Cloud Virtual Machine Resource Usage Prediction

Habte Lejebo Leka [1,*], Zhang Fengli [1,*], Ayantu Tesfaye Kenea [2], Negalign Wake Hundera [3], Tewodros Gizaw Tohye [1] and Abebe Tamrat Tegene [1]

[1] School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610056, China
[2] Department of Computer Science and Engineering, School of Electrical Engineering and Computing, Adama Science and Technology University, Adama P.O. Box 1888, Ethiopia
[3] School of Computer Science and Technology, Zhejiang Normal University, Jinhua 321004, China
* Correspondence: lejebohabte@gmail.com (H.L.L.); fzhang@uestc.edu.cn (Z.F.)

Abstract: To meet the increasing demand for its services, a cloud system should make optimum use of its available resources. Additionally, the high and low oscillations in cloud workload are another significant symmetrical issue that necessitates consideration. A suggested particle swarm optimization (PSO)-based ensemble meta-learning workload forecasting approach uses base models and the PSO-optimized weights of their network inputs. The proposed model employs a blended ensemble learning strategy to merge three recurrent neural networks (RNNs), followed by a dense neural network layer. The CPU utilization of GWA-T-12 and PlanetLab traces is used to assess the method's efficacy. In terms of RMSE, the approach is compared to the LSTM, GRU, and BiLSTM sub-models.

Keywords: BiLSTM; cloud system; ensemble learning; PSO; LSTM; GRU

Citation: Leka, H.L.; Fengli, Z.; Kenea, A.T.; Hundera, N.W.; Tohye, T.G.; Tegene, A.T. PSO-Based Ensemble Meta-Learning Approach for Cloud Virtual Machine Resource Usage Prediction. *Symmetry* **2023**, *15*, 613. https://doi.org/10.3390/sym15030613

Academic Editor: Ruay-Shiung Chang

Received: 15 December 2022
Revised: 4 January 2023
Accepted: 5 January 2023
Published: 28 February 2023

1. Introduction

Cloud service providers in cloud computing environments own vast quantities of varied resources that cloud customers can lease on a pay-as-you-go basis. Typically, cloud service providers aim to enhance resource usage to increase profits, whereas cloud customers aim to increase cost performance to rent adequate resources for their apps. In spite of this, cloud users' applications may experience varying workloads and also have varying quality of service (QoS) needs on the resources offered by cloud providers, necessitating either greater or reduced resource demands over time [1,2]. On the other hand, there is always going to be a latency before the resource is actually usable, which could pose a problem for programs that need to scale their resource usage on the fly. Therefore, it is indeed crucial for cloud providers and users alike to face the challenge of supplying adequate resources with efficient usage QoS assurance [3]. Overprovisioning wastes resources, which increases energy usage and the operational expense of cloud providers, while underprovisioning can cause a reduction in quality of service and the loss of cloud customers.

This problem can be mitigated with accurate foresight of the future workload behavior of resources through careful monitoring. This problem can be addressed by effectual monitoring and keeping a log of how much time and energy are spent on different resources including processor, memory, storage space, and network bandwidth. Afterwards, these indications of past usage can be analyzed and put to use in predictive analysis. This foresight is crucial for effectively supplying cloud resources. The cloud resource management, for instance, may make well-informed decisions about matters such as virtual machine (VM) scaling and VM migration if it has a good idea of how much of a demand is expected in the future and how much of that demand can be forecast using past usage data. Therefore, no QoS drops, maximum resource utilization, and low energy waste during operation of

cloud data centers can be achieved. That is why in ever-changing cloud environments, accurate resource forecasting is a crucial first step toward efficient provisioning and control of available resources. Forecasting and provisioning of cloud-based resources have been the subject of a wide range of academic investigations. In [1], a cloud resource prediction model based on a deep belief network is suggested to forecast central processing unit (CPU) utilization in future workloads. Predicting host CPU utilization in cloud computing using recurrent neural networks has been proposed in [2]. In [3], it is suggested to anticipate workloads for the cloud environment using a statistical hybrid model. A gated recurrent unit (GRU) has been used to predict cloud workload accurately [4].

When it comes to cloud resource provisioning, a number of infrastructure-level resources are taken into account. These include things such as memory, storage, CPU, and network bandwidth. Of these, CPU utilization is crucial for effective management of energy utilization and the proper execution of apps in clouds [1,5–7]. Increased CPU load can indeed lead to aberrant performance [8], and studies have shown that energy usage correlates with CPU usage [9]. Thus, cloud resource allocation necessitates the use of automated, adaptive approaches for predicting CPU demand [8,10]. Despite the abundance of literature concerning cloud resource provisioning forecasting methods [1,2,11], the challenge of forecasting cloud-based multiple virtual machine (VM) CPU utilization has received much less attention. Present-day cloud-based software often consists of a collection of interconnected/cooperating service components, each of which performs a unique task [12–15]. These service components exist in isolation on several distinct virtual machines, all of which collaborate to deliver the service to the end user. Therefore, we present a method for predicting workload that takes into account the CPU utilization of multiple VMs as opposed to only one. Additionally, cloud data centers often house a huge number of processing servers, which necessitate a great deal of power for their functioning, lighting, power source, and coolant, and thus a high operation costs and carbon footprint. As a result, cutting down on energy use and costs is becoming a priority in both business and research. One more challenge for cloud data centers is the need for energy-aware resource allocation in the development of energy-efficient green cloud computing, hence the implementation of smart and efficient resource utilization forecasting strategies is important for increasing the resource utilization ratio and decreasing the energy usage. Additionally, the high and low oscillations in cloud workload are another significant symmetrical issue that necessitates consideration. As the workload pattern changes frequently, it has trouble with fitting. Therefore, there is no single model that can reliably predict the demand for cloud resources in the future [1,16].

In recent years, in order to increase forecasting accuracy, an ensemble technique that integrates numerous models into a hybrid model has been regularly presented [17]. Ensemble models can be formed by combining predictions from multiple models in an effective manner. An ensemble approach for anticipating multiple VM CPU usage is presented in this paper. Therefore, to determine upcoming workload values, the proposed model takes into account predictions of several heterogeneous particle swarm optimization (PSO)-based base models. Additionally, each base model's prediction was fed into the meta-model, which predicts the final outcome. Accordingly, this study introduces an ensemble model for cloud resource provisioning that comprises a gated recurrent unit (GRU), a long short-term memory (LSTM), and a bidirectional long short-term memory (BiLSTM). The following are the primary contributions of our study in this paper:

- First, we propose an ensemble learning-based workload prediction method based on recurrent neural network (RNN) variants and particle swarm optimization (PSO).
- Second, we provide a comparison of the proposed model with PSO-based LSTM, Bi-LSTM, and GRU prediction models for time series prediction algorithms.
- We conducted a series of experiments to evaluate the performance of the proposed PSO-based ensemble prediction model on the dataset of the GWA-T-12 and PlanetLab traces.

The remainder of the paper is organized as follows. The forecasting problem is described in Section 2. Then, in Section 3, the recent major advancements in workload

prediction methods using machine learning are briefly reviewed. The PSO-based ensemble prediction model is deeply covered in Section 4. The assessment criteria that were employed to gauge and contrast the effectiveness of the proposed work are described in Section 5, and then, the finding of the experiments is presented in Section 6. The final section, Section 7, discusses the paper's conclusion.

2. Problem Definition

In Figure 1, we see a typical cloud computing setup that includes some sort of workload estimation module. Both the forecasting model's estimation data and the resource manager's history data are fed into the system as inputs. Using this information, the resource manager module can optimize the service providers' resource use and energy consumption by implementing appropriate scaling policies. The system's forecasting module is crucial because it can foresee changes in the demand for resources.

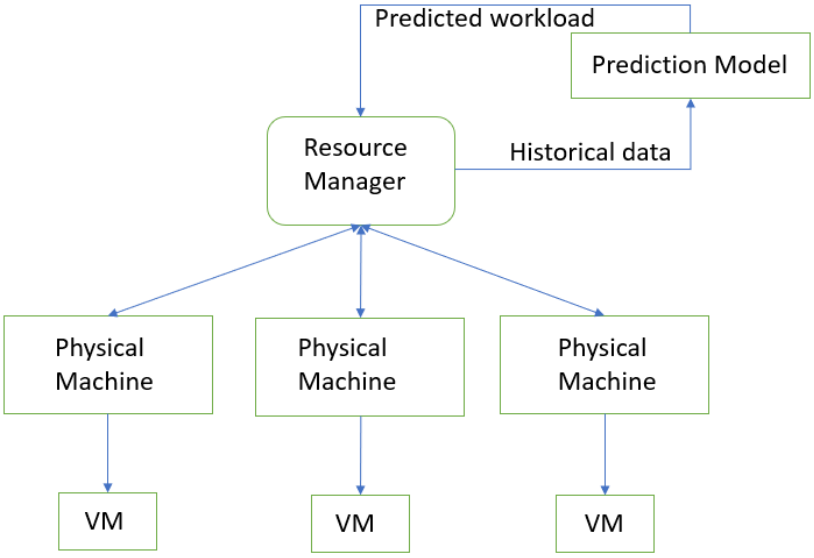

Figure 1. Architecture of Cloud Computing system with prediction Model.

Assume a cloud data center has both physical computers (P) and virtual machines (V), each of which has access to a set amount of hardware resources (CPU, memory, and network speed). Running several different $|A|$ programs in the virtual machine requires a wide range of system resources. Only if a virtual machine has sufficient memory, CPU speed, and storage space can applications be installed on it. It is possible for the cloud system to supply virtual machine resources on request with automatic scaling on the basis of trustworthy workload forecasts, thereby protecting quality of service and decreasing energy usage. A prediction model would be created to discern patterns in workload variance and intrinsic attributes on the basis of workload traces, allowing for future workload predictions for a virtual machine. List of variables and notations are described in Table 1.

Table 1. List of Variables.

Notation	Description
P	Physical computers
V	Virtual machines
$\|A\|$	Application hosted in data center

Most current cloud-based software is made up of a set of interconnected/cooperating service parts that each fulfill a particular function [12–14]. These service parts are dispersed across many different types of virtual machines that work together to provide these services. As a result, we provide a workload prediction approach based on multiple virtual machine

workload trace datasets rather than just one VM workload. A time series is a group of observations of clearly defined data points gathered over time from repeated measurement. It is assumed that the problem of workload forecasting is a time series regression problem and time series forecasting assumes a time series consists of a pattern and some random error [18]. Workload trace data with N variables of length T are represented as $X = (x_1, x_2, x_3, \ldots, x_T)\ T \in \mathbb{R}^{T \times N}$ where for each $t \in \{1, 2, 3, \ldots, T\}, x_t \in \mathbb{R}^N$ represents observations of all variables at time t. The measurement x_t^n denotes the observation of the n^{th} variable of x_t. The variable N considered in the workload trace is CPU usage that is tracked and saved over time from multiple VMs of a cloud data center. It is measurements of CPU usages of VMs for hosting applications. In other words, $x_{1,t}, x_{2,t}, x_{3,t}, \ldots, x_{k,t}$ may be used to denote the related variables (multiple VM CPU usage) and the following form may be used to represent the prediction of $x_{1,t+h}$ at the end of period t.

$$\hat{x}_{1,t+h} = f_1(x_{1,t}, x_{2,t}, x_{3,t}, \ldots, x_{k,t}, x_{1,t-1}, x_{2,t-1}, x_{3,t-1}, \ldots, x_{k,t-1}, x_{1,t-2}, \ldots) \quad (1)$$

As shown in Equation (1), the extraction of sequence information from the workload traces of the virtual machines was required to predict the time information at $x_{1,t+h}$, where h is the desired horizon after the current time stamp that we manually set. The proposed forecasting model's objective is to reduce the difference (error) between the predicted CPU usage value $\hat{x}_{1,t+h}$ and the actual CPU usage $x_{1,t+h}$ for each time.

3. Recent Key Contribution

Recently, cloud service providers have tackled the difficulties of providing quality of service (QoS) and competing for available resources. In recent years, many studies on time series prediction for cloud resource on-demand prediction have been published. Statistical and artificial neural network methods have been adopted to solve this issue, and their efficacy has been the subject of numerous studies [3]. Multiple studies have made an effort to predict cloud-based CPU utilization. Workload in the cloud evolves over time and is connected across periods of time. The CPU load can then be anticipated by analyzing past CPU utilization patterns. Only forecasting models developed utilizing machine learning algorithms, including deep learning approaches, are provided in this section because their baselines and the proposed model share the same domain. However, a comprehensive study of resource management strategies is provided in [19–22].

3.1. Deep Learning-Based Approaches

With the help of a neural network and differential evolution with built-in adaptation, the prediction accuracy was improved [23]. The evolutionary method improves precision by probing the space of possible answers from different angles and employing a wide range of possible answers. This decreases the likelihood of becoming trapped in local optima in comparison to gradient-based machine learning. In [24], a better prediction model based on a neural network was introduced to facilitate precise scaling operations. The suggested strategy classified the VMs based on how they were used before predicting how they would be used in the future. The system used a multilayer perceptron classification technique to accomplish this. Containerized cloud applications can take advantage of the Performance-aware Automatic scaler for Cloud Elasticity (PACE) architecture, which was released in [25] to automatically scale resources in accordance to fluctuating workload. Through the use of convolutional neural networks (CNNs) with K-means, a flexible scaling strategy that anticipates upcoming workload requirements has been developed. The authors of [26] looked at a VM prediction model that forecasts the usage of VMs' resources such as CPU and memory to identify if a program is CPU and/or memory heavy. The research takes a Bayesian approach to predicting VM resource consumption during the workweek. It was recommended in [27] that a machine learning-based model be used to predict load and energy consumption, as well as to manage data center resources. Many machine learning techniques, such as the gated recurrent network (GRU), ElasticNet (EN), ARD regression (ARDR), linear regression, and ridge regression, were explored. The results of

the experiments show that GRU is superior to other networks. Using a regressive predictive multilayer perceptron (MLP) model, [28] attempted to foresee VM power consumption in a data center of a cloud. The model achieves 91% accuracy in making forecasts. The paper [29] proposes a preemptive deep learning-based strategy for automatic scaling docker containers in accordance with fluctuating workload variations during runtime. A forecasting/estimation model which uses a recurrent long short-term memory (LSTM) neural network was proposed to foresee upcoming workload of HTTP. With respect to elastic acceleration and autoscaling associated with provisioning, the proposed model outperforms an artificial neural network. CPU and memory utilization, as well as energy consumption, were forecasted for the next time slot using a multiobjective evolutionary algorithm in [30]. In [11], a method for predicting VM workload is given that makes use of a generative deep belief network (DBN) made up of multiple layers of generative restricted Boltzmann machines (RBMs) with a regression layer model. The authors of [31] presented a solution for anticipating cloud data center workload. The paper's prediction relied on an LSTM network. Predicting cloud data center workload was proposed in [13]. The paper's prediction was made based on an LSTM network. Predicting a VM's upcoming workload using correlation data from its workload trace history was carried out in [32]. The study [1] suggests a BiLSTM deep learning model to accurately predict CPU usage prediction to improve resource provisioning and reduce energy consumption of cloud data centers. In [33], a multivariate BiLSTM model-based autoscaling frame has been proposed to optimize resource provision of cloud data centers and, as per the experiment, the model outperforms other univariate models. GRU-CNN workload prediction has been proposed in [34] to predict incoming workload requests in a cloud data center. The findings suggest that deep learning approaches are superior to traditional methods for making long-term workload predictions.

3.2. Hybrid Approaches

Fitting a certain workload pattern is usually easy for a prediction model that uses a single forecasting model, but it struggles with real-world data when the pattern varies rapidly over time [35]. These conditions persist despite the over- and underprovisioning of resources. Two online learning ensemble learning algorithms [36] were created to anticipate workloads. Changes in a cloud workload demand pattern are reflected rapidly by the models. In order to deal with workloads that are constantly shifting, a novel cloud workload prediction framework [37] was created. It uses multiple predictors to build an ensemble model that can accurately anticipate real-world workloads. CloudInsight, a framework for predicting workloads, was created using a number of forecasters [38]. It integrates eight distinct forecasting algorithms/models from the fields of time series analysis, regression analysis, and machine learning in an effort to improve forecast accuracy. A cloud workload forecast which relies on a weighted wavelet support vector machine is proposed for predicting the server load sequence in a cloud data center [39]. Parameter optimization utilizing a particle swarm optimization (PSO)-based approach was developed in this study.

Several methods have been put forth and used to forecast the workloads of VMs in cloud computing centers as shown in Table 2. It was found that the aforementioned works failed to model and predict the workload demand for multiple VMs. The fact that VMs are interdependent and host related applications that use the resources of multiple VMs is denied by the fact that they were designed and trained for a single VM workload. In addition, the majority of the literature modeled the VM prediction problem using a single method. So that higher forecasting accuracy could be achieved, an integration of different methods and approaches was used to model and anticipate the workloads. To deal with the aforementioned workload challenges, an ensemble strategy is proposed in this work for cloud data center VM workload anticipation.

Table 2. Some typical literature on workload prediction for cloud resource prediction.

References	Technique	Resource for Prediction	Dataset	Metrics	Strengths
[33]	BiLSTM	Virtual machine	GWA-T-12 and GWA-T-13 dataset	Accuracy, memory usage, and network-received throughput	Experiments on different actual workload datasets
[10]	Learning automata (LA)	Virtual machine	CoMon project	Accuracy, CPU load prediction	Virtual machine using LA to tune the weights of multiple prediction models
[40]	CNN-LSTM	Virtual machine	GWA-T-12 dataset	Accuracy, (CPU, memory, and network usage)	Used filtering mechanism to reduce noise in the workload data before inputting into the CNN layer
[41]	BiLSTM	Physical machine	Google load trace data	Accuracy, CPU	Multistep-ahead prediction
[1]	DP-CUPA learning (AR, GM, DBN, and PSO)	Physical machine	Google cluster usage trace	Accuracy, CPU usage	PSO to improve the accuracy of prediction results
[34]	GRU-CNN	Physical machine	All the request data of the Shanghai Telecom base station	Accuracy, request data	Hybrid method CNN and LSTM
PSO-based ensemble model	PSO-based ensemble meta-learning approach (GRU, BiLSTM, LSTM)	Virtual machine	Google cluster and PlanetLab traces	Accuracy, CPU usage	PSO to improve the accuracy of prediction results and consider multiple VMs' CPU usage unlike other studies

4. Cloud Virtual Machine Resource Usage Prediction Based on PSO Ensemble Model

The proposed PSO ensemble model consists of three primary steps: (1) modeling the base prediction model with common recurrent neural network models, i.e., GRU, LSTM, and BiLSTM, (2) optimizing the weights of the base models using a meta-heuristic optimization algorithm, i.e., particle swarm optimization (PSO), and (3) integrating the sub-models/base models. The ultimate forecasting is generated by the meta/learning model, which is fed predictions from the individual networks.

4.1. Particle Swarm Optimization

Kennedy, an American social psychologist, created the PSO algorithm, which is a meta-heuristic. On the basis of Hepper's model for mimicking bird or fish swarms [42,43], Kennedy and Eberhart improved an algorithm of particles flying in the solution space and reaching the optimum solution. With no gradient information and with fewer parameters, PSO is straightforward and simple to carry out. It can be said that PSO is appropriate for both engineering applications and scientific research. We encountered some non-linear issues in our study, and the PSO method—which is now widely used—was developed to find the ideal solution.

Each solution in particle swarm optimization is treated as a particle, and each iterative process determines the present position by applying its own fitness function. Additionally, a particle's velocity determines every step's distance and direction of motion.

Velocity of a particle with its position P_i, $i = \{1, 2, \ldots, N\}$, where N is the overall quantity of particles, updates each iterative process in accordance with the rules:

$$V_{id}^{k+1} = wV_{id}^{k} + c_1 r_1 (p_{id}^{k} - X_{id}^{k}) + c_2 r_2 (p_{gd}^{k} - X_{id}^{k}) \quad (2)$$

$$X_{id}^{k+1} = X_{id}^{k} + V_{id}^{k+1} \tag{3}$$

In this formula, k denotes the number of current iterations; V_{id}^{k} and X_{id}^{k} stand for velocity and position, respectively; c_1 and c_2 stand for a learning coefficient of the local best position p_{id}^{k} and global best position p_{gd}^{k} of the particle p_i; r_1, r_2 stand for two arbitrary numbers in a range of 0 to 1 to raise the chance of search; and w stands for inertia weight to regulate the searchability in a solution space. Compared to other methods, PSO is a good option because of its quick convergence, low computational cost, and global search optimality. These PSO characteristics are crucial for reducing the complexity of the proposed ensemble model. It has been successfully applied in many applications; in [44], PSO was used to model offshore wind power forecast. The paper [43] used PSO in a deep learning predictive model to predict GES mapping. PSO has been used in an LSTM which is used for modeling ship motion prediction in [45]. PSO has been used as a parameter optimization algorithm to find the optimal combination of the parameters of a deep learning model in [39].

Base model weights are calculated using PSO in this research paper. As a measure of fitness, we calculate the sum of squared errors. Every particle's present fitness value is compared to its best fitness value of the past, and if the current value is lower, the particle is updated with the new value and position. A comparison is made between the current fitness value and the global best position. Then, if the current value is lower, the global best position is updated to include the current value and also the position of the current particle. The following three points are supported by experimental evidence. Since the proposed ensemble model makes use of PSO, it is superior to alternatives that rely on slower learning algorithms [46]. Therefore, the proposed model minimizes the mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean squared error (RMSE).

4.2. Ensemble Learning Techniques

An ensemble approach includes various forecasting models, each of which is stated as a base forecaster or expert forecaster, to forecast the expected occurrence of an event. The estimations of each expert are combined using a new model to calculate the ensemble's final result. Figure 2 shows the theoretical layout of an ensemble-based anticipating approach. The objective of the ensemble learning approach is to boost prediction accuracy or overall performance by combining decisions from various base predictors into a new model. MaxVoting, averaging, extreme gradient boosting (XGB), weighted averaging, gradient boosting machine (GBM), stacking, blending, adaptive boosting (AdaBoost), bagging, boosting [47], and other ensemble learning methods are just a few examples. Various ensemble models have distinct traits and can be applied to address distinct challenges across a range of domains.

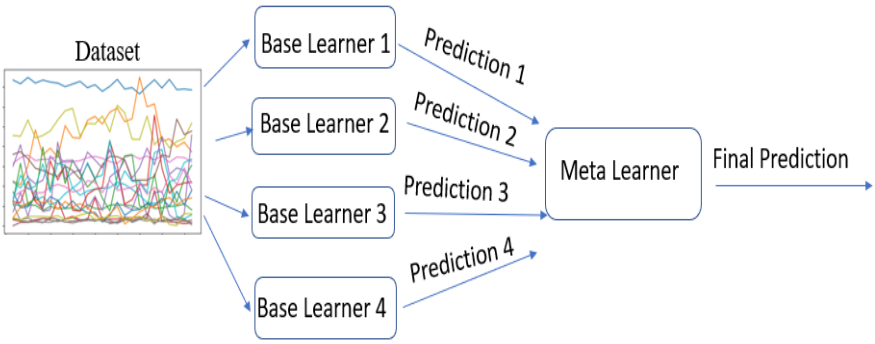

Figure 2. Stacked Generalization ensemble [48].

A straightforward illustration of the ensemble learning approach is that a diverse set of individuals is more likely to reach wiser decisions than an individual is. The same rule holds true for machine learning (ML) models and deep learning (DL) models; a variety of

ML or DL models seem to be more likely to achieve improved results than a standalone model [49] because every model provides a distinctive advantage and may be used to complement one another to make up for their weaknesses.

4.3. Long Short-Term Memory Neural Networks

Human beings may not constantly have to rethink their ideas. Every word is understood in the context of the ones that came before it. Therefore, memory is crucial for recognition, and classic neural networks lack this memory capability. LSTM, a unique kind of RNN that has the capability of recognizing past history, was proposed by [50]. As LSTM has some memory cells, it can handle a variety of time series problems that other machine learning models found challenging. Since it can store significant historical data in the memory cell state and ignore irrelevant data, LSTM is especially successful and well-suited when used in the time series domain. Its structure generally consists of three gates, i.e., input gate, forget gates, and output gates. The LSTM unit handles this intricate job by modifying input data which are kept in the cell state. The forget gate layer is the first gate, which determines the data to be deleted versus kept from the memory cell state. The forget gate layer formula is as follows:

$$f_t = \sigma\Big(W_f \cdot [h_{t-1}, x_t] + b_f\Big) \tag{4}$$

where f_t stands for the forget gate at time t, W_f for the forget neuron's weights, h_{t-1} for the result from the prior cell state at time $t-1$, x_t for the input value at time t, b_f for the forget gate's bias, and σ for the neuron's sigmoid function.

The new information that the neuron needs to store is calculated by the input gate, which involves two steps for making a decision. The values that will be updated are first decided upon by the sigmoid function i_t. The tanh layer $\widetilde{C}_t$ produces a new result value that is then used update the current cell state. The definitions of the formulas are:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5}$$

$$\widetilde{C}_t = \tan\mathrm{h}(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{6}$$

where i_t stands for the gate input at time t, W_i stands for the input weight, and C_t stands for candidate of the cell state at time t; b_i stands for the input gate bias, and b_c stands for the bias of the cell state.

The information that will be output is decided upon by the output gate layer, which is the last gate. The equation is stated as:

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

where W_o stands for the output's weight of neurons, b_o for output gate biases, and o_t for the output gate at time t [50]. By using the product of outcomes of the forget gate f_t and the cell state at the previous time step c_{t-1} plus the product of input gate output i_t and the candidate value $\widetilde{C}_t$, the cell state is updated. The cell state update can be made to the state of the cell using the following equation:

$$c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{C}_t \tag{8}$$

The outcomes from the output unit gate and the cell state at the current time step can then be used to update the h_t which is the hidden cell state (or final output) at the current time state. The equation is defined as follows:

$$h_t = o_t \odot \tan\mathrm{h}(c_t) \tag{9}$$

The LSTM model used in this study is fed information about the workload of four different VMs over the course of an hour. The input is set up with a time window of 48

pasts. Four VM workload traces are included in the input data. In Section 4.5, the specifics of the LSTM structures will be covered.

4.4. Bidirectional Long Short-Term Memory Neural Networks

A bidirectional LSTM (BiLSTM) model for sequence processing comprises two LSTMs, one of which receives input data in a forward manner while the other handles the reverse direction. Figure 3 shows how it is made up of bidirectional recurrent long short-term memory units that function as a forward and backward pair.

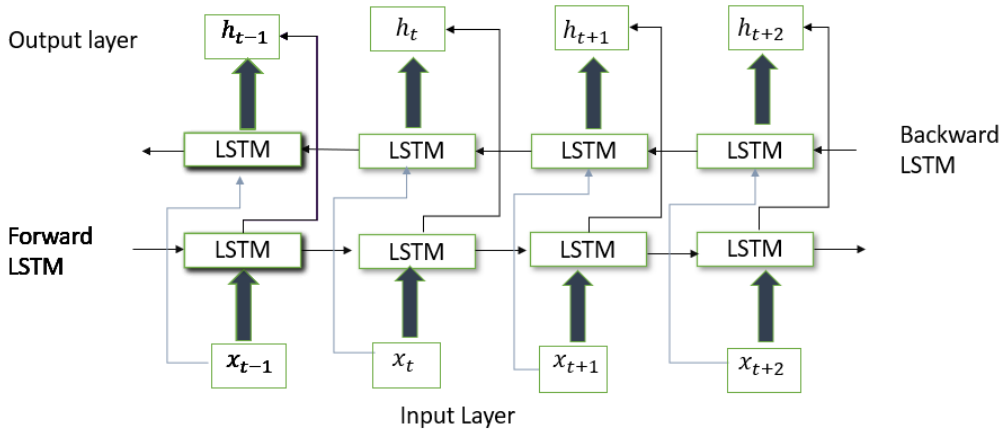

Figure 3. A bidirectional LSTM Architecture.

The foundation of the bidirectional LSTM dependent network is the LSTM layer at its core. The fundamental principle of the bidirectional LSTM unit is to process time series data from the forward and reverse directions using two distinct hidden layers in order to model the effects of past and upcoming information upon the present hidden state, respectively. The forward LSTM unit's internal calculation procedure is described in Equations (5)–(9). The following formula illustrates how the reverse LSTM unit updates the hidden state information primarily using the future information h_{t+1}:

$$\overleftarrow{h}_t = f_1(\overleftarrow{x}_t; \overleftarrow{h}_{t+1}; \overleftarrow{\Theta}_{LSTM})$$

$$\begin{cases} \overleftarrow{i}_t = \sigma(\overleftarrow{W}_i \cdot \left[\overleftarrow{h}_{t+1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_i) \\ \overleftarrow{f}_t = \sigma(\overleftarrow{W}_f \cdot \left[\overleftarrow{h}_{t+1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_f) \\ \overleftarrow{o}_t = \sigma(\overleftarrow{W}_o \cdot \left[\overleftarrow{h}_{t+1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_o) \\ \overleftarrow{\widetilde{C}}_t = \tanh(\overleftarrow{W}_c \cdot \left[\overleftarrow{h}_{t+1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_c) \\ \overleftarrow{c}_t = \overleftarrow{f}_t \odot \overleftarrow{c}_{t+1} + \overleftarrow{i}_t \odot \overleftarrow{\widetilde{C}}_t \\ \overleftarrow{h}_t = o_t \odot \tanh(c_t) \end{cases} \tag{10}$$

4.5. Gated Recurrent Unit Neural Networks

Gated recurrent units (GRUs) were first presented in [51] to tackle the vanishing gradient issue of the classical RNNs through the use of update gates and reset gates. For the same reason as LSTM, it is proposed to fix issues with long-term memory, the gradient in backpropagation, and the gating units that control the flow of data within the unit. A gated recurrent unit does not have its own memory cell, which is a major contrast to the LSTM layer. Before proceeding to the next step, the GRU's update gate determines the

amount of information from the past that must be transmitted to the future. According to the update gate equation:

$$z_t = \sigma(W^z x_t + U^z h_{t-1}) \tag{11}$$

where x_t stands for the input at time t, W^z stands for the weights for the update gate, h_{t-1} stands for the stored values at $t-1$ units, U^z stands for weights of the past hidden state h_{t-1}, and z_t represents the update gate at a time step t.

A hidden state h_{t-1} that was transmitted to the GRU from the past node and includes data about it has a current input of x_t. The GRU receives the result/output y_t of the currently hidden node and the hidden state h_t passed to the next node by combining x_t and h_{t-1}.

The reset gate is the second main part of the GRU. How much of the previous data needs to be forgotten is determined by the reset gate. Here is the definition of the reset gate equation:

$$r_t = \sigma(W^r x_t + U^r h_{t-1}) \tag{12}$$

In order to determine how much of the old memory needs to be preserved, the computation of the nominee hidden layer, which can be thought of as the new data at the time, is used. For example, if it is 0, then only the data of the current value are included.

$$\widetilde{h}_t = \tan \mathrm{h}(W x_t + r_t \odot U h_{t-1}) \tag{13}$$

The internal process the GRU uses to update its memory is one of its strengths.

$$h_t = (1-z) \times h_{t-1} + z \times h' \tag{14}$$

where h_{t-1} and h_t are the state memory variable at the past state $t-1$ and current state t, respectively, and z represents the update gate state.

The GRU, as opposed to LSTM, which calls for multiple gating, uses a single z to accomplish forgetting and selection memory. As a result, the GRU may be more effective than LSTM during the training phase.

4.6. Architecture Overview

Ensemble learning combines different machine learning and deep learning models in order to carry out forecasting or classify data. Combining the forecasts from many neural networks [52] helps to decrease the variation of forecasts and generalization error. Figure 4 depicts the blended ensemble model block diagram, which is an ensemble model utilized in this study, as well as an overview of the architecture. The suggested prediction model is described here; it makes use of an ensemble method to boost forecast precision.

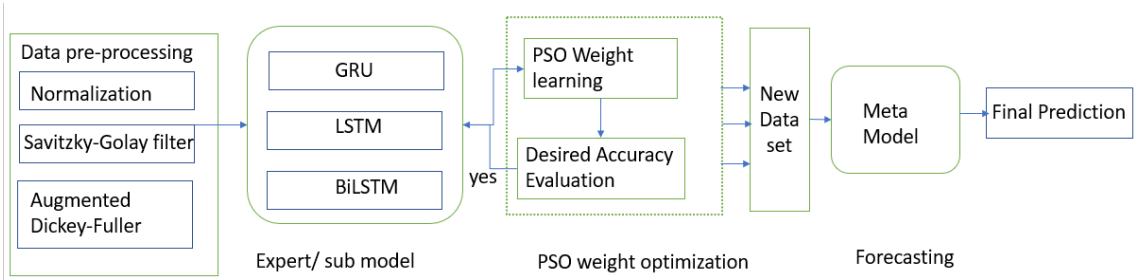

Figure 4. Block diagram the proposed PSO-based Ensemble Model.

The proposed PSO-based ensemble approach consists of three steps:

Step 1: The historical time series data of multiple VMs on CPU usage are preprocessed to generate matrix data with equal lengths.

Step 2: The matrix is used as the input of the BiLSTM, GRU, and LSTM models and optimized using PSO, by which intermediate prediction results can be obtained and used to generate a new data matrix for training the meta-model.

Step 3: Based on the new data matrix, the meta-model network is designed to generate the final prediction results.

There are four essential components: data preprocessing, which scales and organizes the dataset; expert, which contains base predictor models; optimizer, which optimizes the weights associated with each base prediction; and forecasting, which evaluates the accuracy of the model. Figure 4 shows that data are passed along from one stage or component to the next. Only if the accuracy indicated by the labels on the arrows (yes/no) is high will the next phase be executed.

Figure 5 depicts the whole workflow of the ensemble model. The algorithm's inputs were the CPU utilization data of four distinct virtual machines. After applying the Savitzky–Golay filter [53] to eliminate noise and smooth the data trend, min–max normalization was applied to scale the dataset to workload values in the range [0, 1]. Each expert and base model receives the supplied data. After applying PSO to optimize the weights of each expert model, we estimated the future workload data. To determine the final result of the predictive ensemble model, each expert's prediction is also fed into the meta-learner, which produces the final prediction.

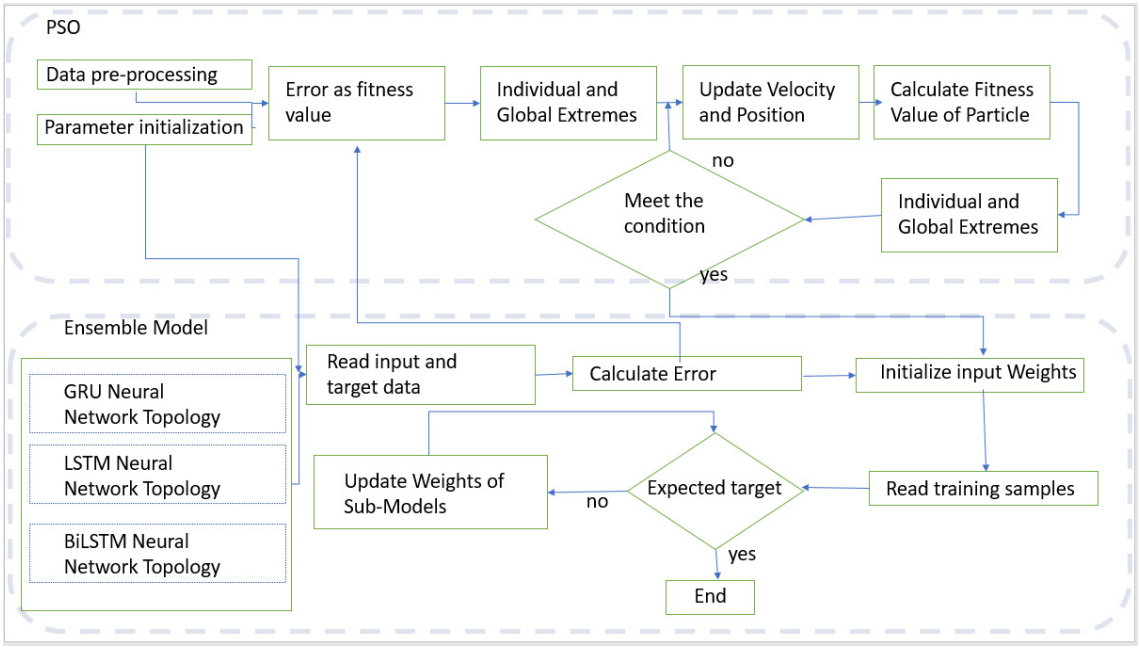

Figure 5. Proposed predictive PSO-based Ensemble model.

Ensemble Expert Learning with PSO Optimization

Throughout the course of our investigations, we have investigated a vast array of setup possibilities. Our empirical data guided the selection of all of our parameters, including epoch size, neuron count, and layer count. The initial level of the blended ensemble model consists of three RNNs: the GRU, the BiLSTM, and the LSTM model. Training data, validation data, and test data were previously separated from the dataset. For the training of the blended ensemble model, each dataset is required. The GRU model, BiLSTM model, and LSTM model are sub-models of level 1 that are trained using the training data. For the level 1 models, each sub-model's weight is optimized using PSO. Following the initial phase of training, the learned level 1 models are applied to the validation data which effectively serve as the actual training data for the second level of models. In addition, the testing data are used to determine the ultimate forecast and accuracy of the proposed model.

First, for sub-model 1, the GRU model is trained using the training data. This GRU model consists of only four layers, each containing 50 neurons. The model is trained with 100 iterations with 0.2 dropouts per hidden layer. PSO is utilized for training model weights. After training the GRU model, the validation dataset is used to produce the initial prediction. The initial predictions made by the GRU utilizing validation are those based on GRU validation predictions.

The BiLSTM model, which is the second sub-model, is then trained. Similarly, the BiLSTM model that we construct has two layers, one of which is dense, with 50 neurons per layer. Additionally, we use 0.2 dropouts per hidden layer and the model is trained for a total of 100 iterations. Using the PSO for training the model weights, the BiLSTM model is trained in the same manner as the GRU model. After training, the validation dataset is provided to the BiLSTM in order to provide BiLSTM validation predictions.

Finally, the LSTM sub-model is trained using the train dataset as the third base/sub-model. The model consists of only three layers, each containing 32 neurons. In that the PSO is used to train the weights of the model, the LSTM model undergoes similar training steps as the GRU and BiLSTM models. Once we have a trained LSTM model, we feed the validation sample data into the trained model to obtain LSTM validation predictions.

We create a new dataset with the form of $p \times m$ (p indicates the total quantity of forecasts and m represents the total quantity of models) by combining the GRU validation prediction, BiLSTM validation prediction, and LSTM validation prediction. Then, this newly generated dataset is utilized for fitting the second stage meta-learner model. A meta-learner is another term for the second level. It consists of a dense neural network with one hidden layer of 32 neurons and rectified linear unit (ReLu) function is used as an activation function. As is carried out for the validation dataset, again, the testing samples of the original dataset are used to test the sub-models. Then, again, the result generates another $p \times m$ testing dataset which is used to test the meta-model. Therefore, on the basis of the newly generated test predictions from the sub-models, the meta-learner will subsequently generate the final predictions of our proposed PSO-based ensemble model.

5. Evaluation Metrics

Various error measurements were utilized to evaluate the precision of predictions. We employed the three statistical metrics [54] mean absolute error (MAE), which is less biased for large mistakes and outliers but may not effectively capture huge errors, standard deviation (SD), and variance (V). In the trials, root mean square error (RMSE) and mean absolute percent error (MAPE) were also used to gauge the prediction efficacy of the predictive performance of our suggested ensemble model. RMSE is defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2} \tag{15}$$

$$\text{AE} = \frac{\sum_{i=1}^{n}|x_i - \hat{x}_i|}{n} \tag{16}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{x_i - \hat{x}_i}{x_i}\right| \tag{17}$$

where n is the number of forecasts, x_i is a vector of recorded values associated with the variable being forecasted, $\hat{x}_i$ is the estimated/forecasted values.

6. Experiment and Dataset

We trained and evaluated our proposed ensemble model using CPU utilization information for four distinct virtual machines extracted from the GWA-T-12 dataset and PlanetLab traces. Our experiment's data were obtained from the bitbrains GWA-T-12 and PlanetLab traces. Bitbrains is the long-term, large-scale workload trail of a distributed cloud data center in the real world. The data center's virtual machines host business-critical

applications. We employed a storage area network (SAN) containing information for 1250 virtual machines. The consumption of system resources such as memory, the central processor unit, network input/output, and disk input/output was monitored every five minutes. PlanetLab, an environment for exploring cloud-based software, was the second dataset used. Currently, PlanetLab has over a thousand nodes spread across seven hundred locations worldwide, all of which report their virtual machines' CPU utilization every five minutes in real time. We have randomly selected four VM samples from the PlanetLab and the GWA-T-12 dataset's fast storage trace that exhibit typical workload patterns. All of the CPU usage of the specified VMs was used as input data for the prediction model. There was a 60:20:20 split between the training set, the validation set, and the test set. The CPU utilization dataset was standardized to the range [0–1] using a min–max scaler before we used it to train our model. The CPU usage history window of the four virtual machines that exhibit a similar pattern served as the input vector for the ensemble learning model. The objective was to forecast the CPU utilization of the selected four VMs at one-hour intervals. Using RMSE, MAE, and MAPE, the suggested ensemble model's prediction accuracy was tested. We used the Python programing language and TensorFlow deep learning framework to evaluate our model.

6.1. Experimental Results

VM Selection Results

Table 3 illustrates the Pearson's correlation scores of the virtual machines that showed common trends for the GWA-T-12 dataset. Clearly, of the data presented in the table, the VM CPU usage had a significant positive association. This means that the VMs are interdependent that the CPU consumption of one VM affects the other VMs' CPU usage. Thus, the CPU consumption of VMs served as inputs for the multiple input ensemble model used to evaluate the accuracy of prediction.

Table 3. VM selection of the GWA-T-12 dataset.

	VM1_CPU_USAGE	**VM2_CPU_ USAGE**	**VM3_CPU_ USAGE**	**VM4_CPU_ USAGE**
VM1_CPU_usage	1	0.529261	0.399846	0.702094
VM2_CPU_usage	0.529261	1	0.40572	0.764131
VM3_CPU_usage	0.399846	0.40572	1	0.458631
VM4_CPU_usage	0.702094	0.764131	0.458631	1

6.2. Prediction Performance Results

The experimental result of the sub-models and the proposed PSO-based ensemble model is illustrated in Tables 4–7. The proposed model has a lower forecasting error than the sub-models LSTM, GRU, and BiLSTM. It reduces prediction error by 79.72–91.95% and 82.64–86.91% on the RMSE metric as is shown in Tables 4 and 5 for the GWA-T-12 dataset and PlanetLab trace, respectively. Throughout the experiments, we compared the predicted CPU utilization of the four VMs with their actual CPU utilization and calculated the MAE, MAPE, and RMSE values. The predicted values can be used to anticipate the CPU usage of the VMs for the next hour. As a result, VM resources can be effectively managed due to the efficient allocation of resources, such as CPU, to the VMs hosting the workloads on demand. The proposed blended PSO ensemble model is compared with the sub-models LSTM, BiLSTM, and GRU, which are widely used by cloud service provider centers for predicting future resource workloads in order to automatically provide resources on demand and revoke user resources for the system in advance in order to meet service level agreements. On the basis of the experimental findings, the proposed PSO ensemble model outperforms the competition. The MAE is 2.207, the MAPE is 0.545, and the RMSE is 3.146, as shown in Table 6. A similar comparison is made between the three sub-models and the performance of the suggested model on the mean CPU usage of the PlanetLab traces of data using the MAE, MAPE, and RMSE. The experimental data demonstrate that, in comparison to

the individual models, the suggested PSO ensemble model reduces error by 86.91%. The outcomes show that the suggested method generates forecasts with a higher degree of accuracy. The findings of the experiment are detailed in Tables 6 and 7. As demonstrated in Tables 4–7, the proposed blended PSO ensemble model is superior compared to all other models in every metric. The blended PSO-based ensemble model significantly improves MAE, MAPE, and RMSE values.

Table 4. Percentage improvement results of PSO ensemble model on the GWA-T-12 dataset.

Model	MAE	MAPE	RMSE
GRU	89.19%	93.82%	91.95%
LSTM	79.42%	94.94%	79.72%
BiLSTM	85.05%	94.59%	86.19%

Table 5. Percentage improvement results of PSO ensemble model on the PlanetLab dataset.

Model	MAE	MAPE	RMSE
GRU	86.13%	88.06%	86.91%
LSTM	82.98%	84.55%	82.64%
BiLSTM	86.2%	88.01%	86.88%

Table 6. Proposed ensemble model compared against other models on the GWA-T-12 dataset.

Model	MAE	MAPE	RMSE
GRU	20.423	8.826	39.111
LSTM	10.725	10.771	15.514
BiLSTM	14.766	10.079	22.79
PSO Ensemble	2.207	0.545	3.146

Table 7. Proposed ensemble model compared against other models on the PlanetLab dataset.

Model	MAE	MAPE	RMSE
GRU	3.15	9.709	4.143
LSTM	2.568	7.502	3.123
BiLSTM	3.167	9.67	4.132
PSO Ensemble	**0.437**	**1.159**	**0.542**

Figures 6–9 show that the predicted values of the RMSE learning curve of both the train and validation data are close to each other and our model converges faster and smoother than the other models for the GWA-T-12 dataset. The RMSE values of train and test data have small variations, which means the model has been generalized adequately, as shown in Figure 9.

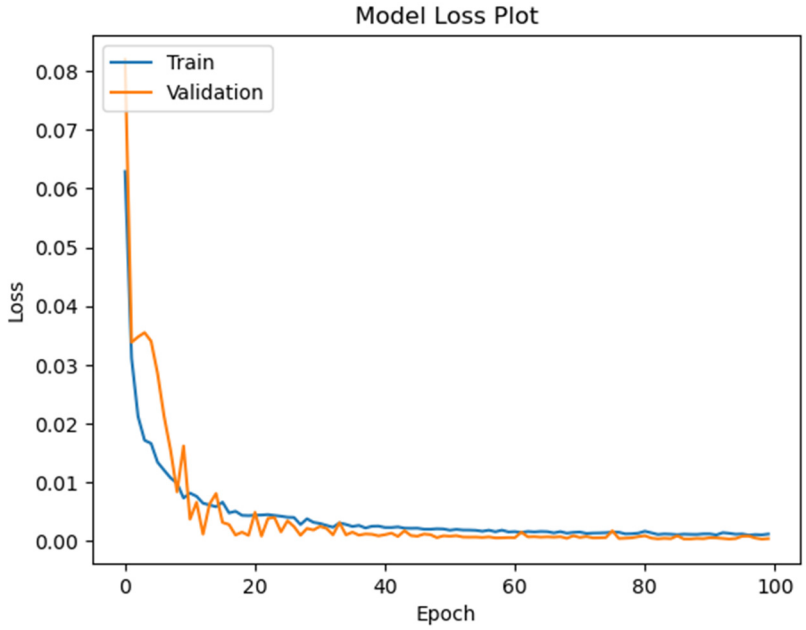

Figure 6. Loss graph of BiLSTM model on GWA-T-12 dataset.

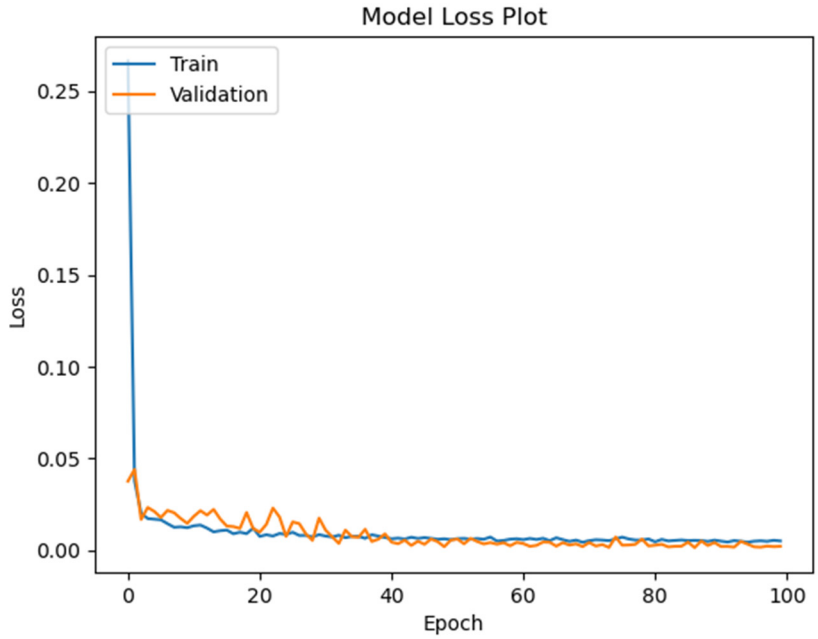

Figure 7. Loss graph of GRU model on GWA-T-12 dataset.

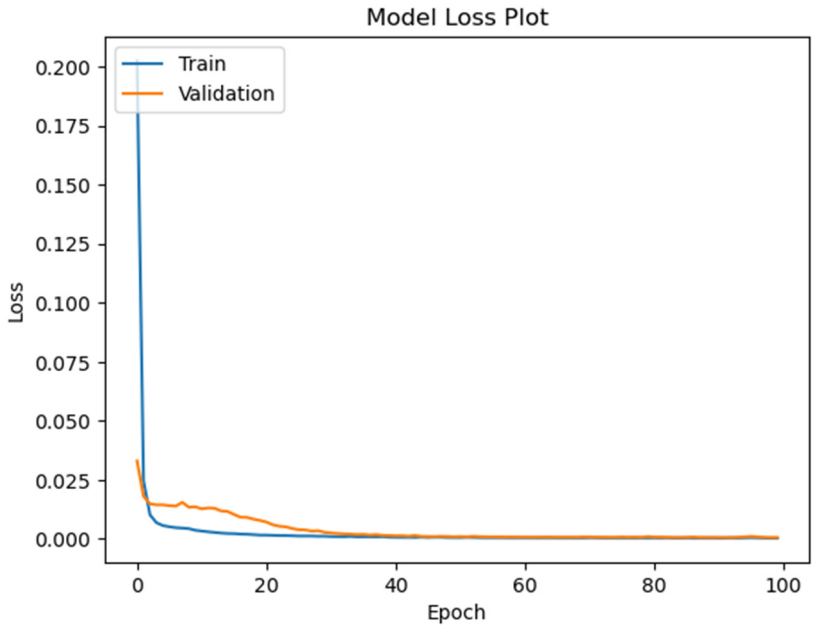

Figure 8. Loss graph of LSTM model on GWA-T-12 dataset.

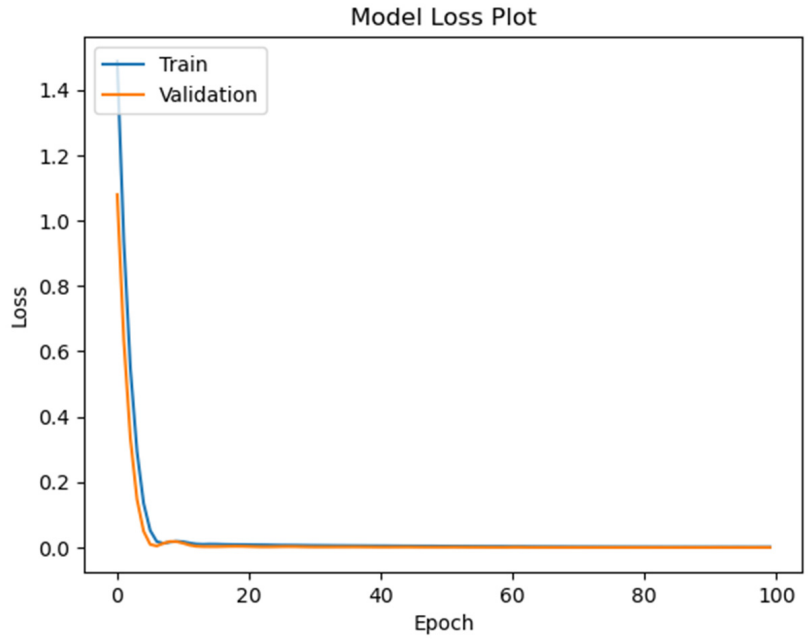

Figure 9. Loss graph of the proposed PSO ensemble model on GWA-T-12 dataset.

Figures 10–13 shows the base models and the proposed model's predicted CPU usage versus the actual CPU usage of GWA-T-12 dataset.

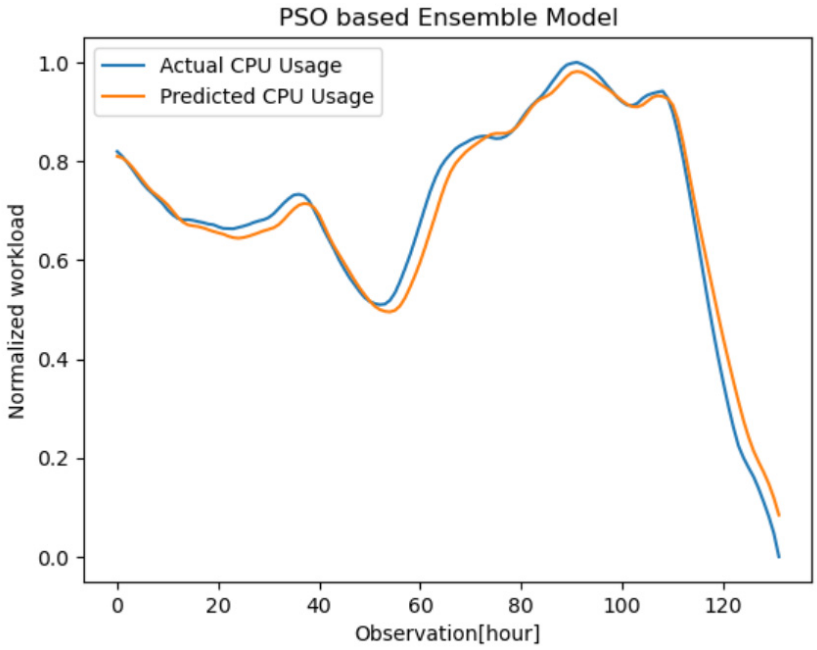

Figure 10. Result of the proposed ensemble model using the GWA-T-12 dataset.

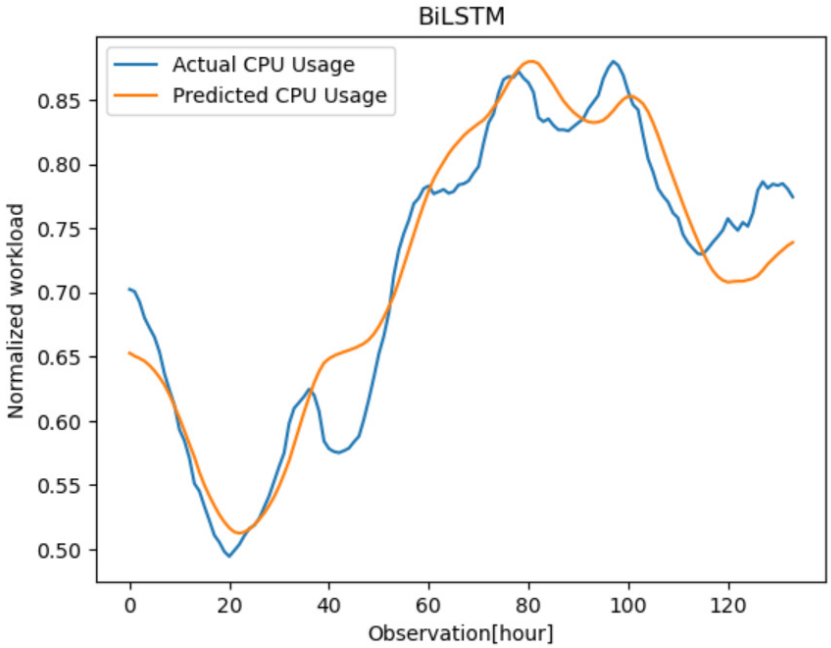

Figure 11. Result of the BiLSTM model using the GWA-T-12 dataset.

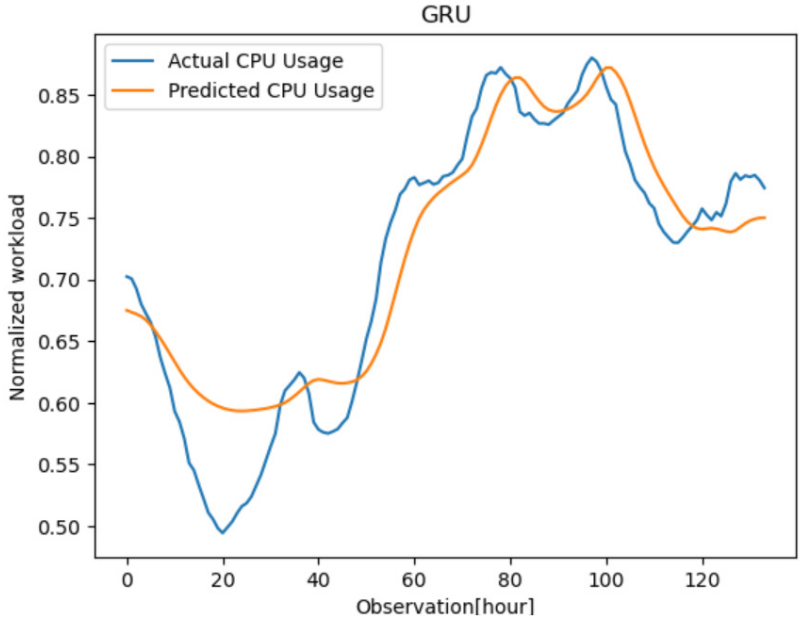

Figure 12. Result of the GRU model using the GWA-T-12 dataset.

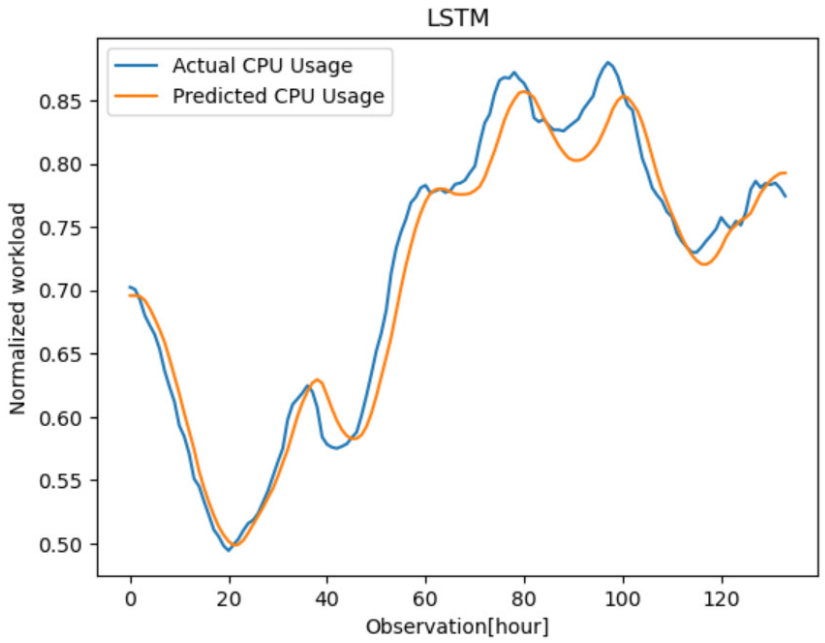

Figure 13. Result of the LSTM model using the GWA-T-12 dataset.

7. Conclusions

The widespread adoption of cloud computing has had a significant effect on the efficiency of organizations in a variety of sectors. Autoscaling, the dynamic adjustment of cloud infrastructure by adding or removing resources on the go, is crucial. Workload requirements, however, are difficult to forecast because they alter continuously over time. To predict the demand on resources, various studies have suggested using time series forecasting. These research efforts have mostly concentrated on one aspect of time series prediction: the univariate case. The proposed model does not rely on a central neural network to create workload forecasts. However, there are numerous variables that might affect the VMs' CPU consumption, therefore, it tends to fluctuate a lot. Predicting future trends in VMs' CPU utilization is difficult for the reason that it is usually impossible for a model to understand all of the characteristics of the data. The suggested model is a multimodel ensemble that

can simultaneously learn diverse data features while simultaneously suppressing noise, and this is the main contribution of the paper. This study offered a method for predicting the workload of several virtual machines sharing common trends, which may be used to optimize cloud computing. In order to predict workload demand, or CPU utilization, this study proposes a deep learning architecture that combines multiple recurrent neural networks into one. The mixed ensemble model accounts for the temporal variations that have an effect on CPU utilization. To test the viability of the approach and the proposed blending ensemble model, an experiment was performed on several RNN models in a similar context and with the same settings. The efficacy of the blended ensemble model was measured in a variety of ways, including absolute percentage error, mean absolute error and root mean squared error. Considering our experiment, our blended ensemble model performed considerably better against state-of-the-art methods in all benchmarks, suggesting that the proposed model holds tremendous promise for anticipating cloud resource demands for cloud service providers. The method's efficiency was measured by analyzing the CPU usage on GWA-T-12 and PlanetLab traces. The method was assessed in relation to the LSTM, GRU, and BiLSTM components of the deep learning models. The findings shows that the technique has a lower root mean squared error than the other models by a margin of 79.72% to 91.95%.

Due to the large number of parameters involved in deep learning model training, enhancing training performance has been a topic of intense interest and critical need in the field. This work investigates the potential of the particle swarm optimization algorithm to enhance the training and accuracy efficiency of the ensemble model. Providers of cloud computing services must be able to accurately forecast cloud workload in order to efficiently meet their customers' demands while minimizing operating expenses. Specifically, the experiments provide preliminary validation of the performance of the proposed model for anticipating cloud workloads. The findings suggest that cloud service providers can utilize the given ensemble model to distribute VM resources in advance based on workload predictions.

In the future, we will try to build a more robust forecasting model which will considers various performance metrics from different datasets such as Google cluster and Alibaba. This will provide large-scale flexibility in the forecasting model. Again, we will try to design an energy-efficient autoscaling framework, based on the CPU usage knowledge obtained from the proposed PSO-based ensemble model.

Author Contributions: Conceptualization, H.L.L.; Data curation, H.L.L. and A.T.T.; Formal analysis, H.L.L.; Funding acquisition, H.L.L.; Investigation, H.L.L.; Methodology, H.L.L.; Project administration, Z.F.; Resources, H.L.L., A.T.K., N.W.H. and T.G.T.; Software, H.L.L.; Supervision, Z.F.; Validation, H.L.L., A.T.K. and A.T.T.; Visualization, H.L.L. and N.W.H.; Writing—original draft, H.L.L. and Z.F.; Writing—review and editing, H.L.L., Z.F., N.W.H. and T.G.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (62271128, 61802033), Sichuan Regional Innovation Cooperation Project (2020YFQ0018), key Rand D Projects of Sichuan Science and Technology Program (2022ZDX0004, 22ZDX0046, 2022YFG0212, 2021YFG0027, 2020YFG0475, 2019YJ0543).

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: The data that support the findings of this study are openly available at http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains (accessed on 12 September 2022) and https://github.com/beloglazov/planetlab-workload-traces (accessed on 12 September 2022).

Acknowledgments: We would like to extend our gratitude to Zhang Fengli for her detailed oversight, as well as to the Editors and Reviewers for the contributions they made to this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wen, Y.; Wang, Y.; Liu, J.; Cao, B.; Fu, Q. CPU usage prediction for cloud resource provisioning based on deep belief network and particle swarm optimization. *Concurr. Comput.* **2020**, *32*, e5730. [CrossRef]
2. Duggan, M.; Mason, K.; Duggan, J.; Howley, E.; Barrett, E. Predicting host CPU utilization in cloud computing using recurrent neural networks. In Proceedings of the 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST), Cambridge, UK, 11–14 December 2017; pp. 67–72. [CrossRef]
3. Devi, K.L.; Valli, S. Time series-based workload prediction using the statistical hybrid model for the cloud environment. *Computing* **2022**, *105*, 353–374. [CrossRef]
4. Chen, Z.; Hu, J.; Min, G.; Zomaya, A.Y.; El-Ghazawi, T. Towards Accurate Prediction for High-Dimensional and Highly-Variable Cloud Workloads with Deep Learning. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *31*, 923–934. [CrossRef]
5. Wen, Y.; Wang, Z.; Zhang, Y.; Liu, J.; Cao, B.; Chen, J. Energy and cost aware scheduling with batch processing for instance-intensive IoT workflows in clouds. *Future Gener. Comput. Syst.* **2019**, *101*, 39–50. [CrossRef]
6. Subirats, J.; Guitart, J. Assessing and forecasting energy efficiency on Cloud computing platforms. *Future Gener. Comput. Syst.* **2015**, *45*, 70–94. [CrossRef]
7. Akram, U.; Fülöp, M.T.; Tiron-tudor, A.; Topor, D.I. Impact of Digitalization on Customers' Well-Being in the Pandemic Period: Challenges and Opportunities for the Retail Industry. *Int. J. Environ. Res. Public Health* **2021**, *18*, 7533. [CrossRef]
8. Yang, D.; Cao, J.; Fu, J.; Wang, J.; Guo, J. A pattern fusion model for multi-step-ahead CPU load prediction. *J. Syst. Softw.* **2013**, *86*, 1257–1266. [CrossRef]
9. Hsu, C.H.; Slagter, K.D.; Chen, S.C.; Chung, Y.C. Optimizing energy consumption with task consolidation in clouds. *Inf. Sci.* **2014**, *258*, 452–462. [CrossRef]
10. Rahmanian, A.A.; Ghobaei-Arani, M.; Tofighy, S. A learning automata-based ensemble resource usage prediction algorithm for cloud computing environment. *Future Gener. Comput. Syst.* **2018**, *79*, 54–71. [CrossRef]
11. Qiu, F.; Zhang, B.; Guo, J. *A Deep Learning Approach for VM Workload Prediction in the Cloud*; IEEE: New York, NY, USA, 2016.
12. Qiu, F.; Zhang, B.; Guo, J. A deep learning approach for VM workload prediction in the cloud. In Proceedings of the 2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Shanghai, China, 30 May–1 June 2016; pp. 319–324. [CrossRef]
13. Zhang, Q.; Yang, L.T.; Yan, Z.; Chen, Z.; Li, P. An Efficient Deep Learning Model to Predict Cloud Workload for Industry Informatics. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3170–3178. [CrossRef]
14. Ruan, L.; Bai, Y.; Li, S.; He, S.; Xiao, L. Workload time series prediction in storage systems: A deep learning based approach. *Clust. Comput.* **2021**, *26*, 25–35. [CrossRef]
15. Fülöp, M.T.; Breaz, T.O.; He, X.; Ionescu, C.A.; Cordo, G.S. The role of universities' sustainability, teachers' wellbeing, and attitudes toward e-learning during COVID-19. *Front. Public Health* **2022**, *10*, 981593. [CrossRef]
16. Kumar, J.; Singh, A.K.; Buyya, R. Ensemble learning based predictive framework for virtual machine resource request prediction. *Neurocomputing* **2020**, *397*, 20–30. [CrossRef]
17. Yazdanian, P.; Sharifian, S. *E2LG: A Multiscale Ensemble of LSTM/GAN Deep Learning Architecture for Multistep-Ahead Cloud Workload Prediction*; Springer: Berlin/Heidelberg, Germany, 2021; Volume 77. [CrossRef]
18. Kalekar, P. Time series forecasting using Holt-Winters exponential smoothing. *Kanwal Rekhi Sch. Inf. Technol.* **2004**, *4329008*, 1–13. Available online: http://www.it.iitb.ac.in/~praj/acads/seminar/04329008_ExponentialSmoothing.pdf (accessed on 10 December 2022).
19. Weingärtner, R.; Bräscher, G.B.; Westphall, C.B. Cloud resource management: A survey on forecasting and profiling models. *J. Netw. Comput. Appl.* **2015**, *47*, 99–106. [CrossRef]
20. Manvi, S.S.; Krishna Shyam, G. Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey. *J. Netw. Comput. Appl.* **2014**, *41*, 424–440. [CrossRef]
21. Li, Z.; Yu, X.; Yu, L.; Guo, S.; Chang, V. Energy-efficient and quality-aware VM consolidation method. *Future Gener. Comput. Syst.* **2020**, *102*, 789–809. [CrossRef]
22. Sun, X.; Ansari, N.; Wang, R. Optimizing Resource Utilization of a Data Center. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2822–2846. [CrossRef]
23. Kumar, J.; Singh, A.K. Workload prediction in cloud using artificial neural network and adaptive differential evolution. *Future Gener. Comput. Syst.* **2018**, *81*, 41–52. [CrossRef]
24. Kumar, K.D.; Umamaheswari, E. EWPTNN: An Efficient Workload Prediction Model in Cloud Computing Using Two-Stage Neural Networks. *Procedia Comput. Sci.* **2019**, *165*, 151–157. [CrossRef]
25. Chouliaras, S.; Sotiriadis, S. Auto-scaling containerized cloud applications: A workload-driven approach. *Simul. Model. Pract. Theory* **2022**, *121*, 102654. [CrossRef]
26. Kumaraswamy, S.; Nair, M.K. Intelligent VMs prediction in cloud computing environment. In Proceedings of the 2017 International Conference on Smart Technologies for Smart Nation (SmartTechCon), Bengaluru, India, 17–19 August 2017; pp. 288–294. [CrossRef]
27. Khan, T.; Tian, W.; Ilager, S.; Buyya, R. Workload forecasting and energy state estimation in cloud data centres: ML-centric approach. *Future Gener. Comput. Syst.* **2022**, *128*, 320–332. [CrossRef]

28. Deepika, T.; Prakash, P. Power consumption prediction in cloud data center using machine learning. *Int. J. Electr. Comput. Eng.* **2020**, *10*, 1524–1532. [CrossRef]
29. Imdoukh, M.; Ahmad, I.; Alfailakawi, M.G. Machine learning-based auto-scaling for containerized applications. *Neural Comput. Appl.* **2020**, *32*, 9745–9760. [CrossRef]
30. Tseng, F.H.; Wang, X.; Chou, L.D.; Chao, H.C.; Leung, V.C.M. Dynamic Resource Prediction and Allocation for Cloud Data Center Using the Multiobjective Genetic Algorithm. *IEEE Syst. J.* **2018**, *12*, 1688–1699. [CrossRef]
31. Kumar, J.; Goomer, R.; Singh, A.K. Long Short Term Memory Recurrent Neural Network (LSTM-RNN) Based Workload Forecasting Model for Cloud Datacenters. *Procedia Comput. Sci.* **2018**, *125*, 676–682. [CrossRef]
32. Varma, P.R.K.; Kumari, V.V.; Kumar, S.S. *Progress in Computing, Analytics and Networking*; Springer: Singapore, 2018; Volume 710, ISBN 978-981-10-7870-5.
33. Dang-Quang, N.M.; Yoo, M. An Efficient Multivariate Autoscaling Framework Using Bi-LSTM for Cloud Computing. *Appl. Sci.* **2022**, *12*, 3523. [CrossRef]
34. Gan, Z.; Chen, P.; Yu, C.; Chen, J.; Feng, K. Workload Prediction based on GRU-CNN in Cloud Environment. In Proceedings of the 2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI), Shijiazhuang, China, 22–24 July 2022; pp. 472–476. [CrossRef]
35. Chen, Z.; Zhu, Y.; Di, Y.; Feng, S. Self-adaptive prediction of cloud resource demands using ensemble model and subtractive-fuzzy clustering based fuzzy neural network. *Comput. Intell. Neurosci.* **2015**, *2015*, 17. [CrossRef]
36. Singh, N.; Rao, S. Ensemble learning for large-scale workload prediction. *IEEE Trans. Emerg. Top. Comput.* **2014**, *2*, 149–165. [CrossRef]
37. Kim, I.K.; Wang, W.; Qi, Y.; Humphrey, M. Forecasting Cloud Application Workloads with CloudInsight for Predictive Resource Management. *IEEE Trans. Cloud Comput.* **2020**, *10*, 1848–1863. [CrossRef]
38. Kim, I.K.; Wang, W.; Qi, Y.; Humphrey, M. CloudInsight: Utilizing a Council of Experts to Predict Future Cloud Application Workloads. In Proceedings of the 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2–7 July 2018; pp. 41–48. [CrossRef]
39. Zhong, W.; Zhuang, Y.; Sun, J.; Gu, J. A load prediction model for cloud computing using PSO-based weighted wavelet support vector machine. *Appl. Intell.* **2018**, *48*, 4072–4083. [CrossRef]
40. Ouhame, S.; Hadi, Y.; Ullah, A. An efficient forecasting approach for resource utilization in cloud data center using CNN-LSTM model. *Neural Comput. Appl.* **2021**, *33*, 10043–10055. [CrossRef]
41. Shen, H.; Hong, X. Host Load Prediction with Bi-directional Long Short-Term Memory in Cloud Computing. *arXiv* **2020**, arXiv:2007.15582.
42. Okwu, M.O.; Tartibu, L.K. Particle Swarm Optimisation. *Stud. Comput. Intell.* **2021**, *927*, 5–13. [CrossRef]
43. Band, S.S.; Janizadeh, S.; Pal, S.C.; Saha, A.; Chakrabortty, R.; Shokri, M.; Mosavi, A. Novel ensemble approach of deep learning neural network (Dlnn) model and particle swarm optimization (pso) algorithm for prediction of gully erosion susceptibility. *Sensors* **2020**, *20*, 5609. [CrossRef]
44. Yuan, C.; Tang, Y.; Mei, R.; Mo, F.; Wang, H. A PSO-LSTM Model of Offshore Wind Power Forecast considering the Variation of Wind Speed in Second-Level Time Scale. *Math. Probl. Eng.* **2021**, *2021*, 2009062. [CrossRef]
45. Yao, Y.; Han, L.; Wang, J. LSTM-PSO: Long Short-Term Memory Ship Motion Prediction Based on Particle Swarm Optimization. In Proceedings of the 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC), Xiamen, China, 10–12 August 2018. [CrossRef]
46. Gudise, V.G.; Venayagamoorthy, G.K. Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks. In *Proceedings of the Comparison of Particle Swarm Optimization and Backpropagation as Training Algorithms for Neural Networks*; IEEE: New York, NY, USA, 2003; Volume 2, Available online: https://ieeexplore.ieee.org/document/1202255 (accessed on 3 January 2023).
47. Hartmanis, J.; Leeuwen, J. *Van Multiple Classifier Systems—First International Workshop, MCS 2000, Proceedings*; LNCS; Springer: Berlin/Heidelberg, Germany, 2000; Volume 1857.
48. Ensemble Learning: 5 Main Approaches. 2019. Available online: https://www.kdnuggets.com/2019/01/ensemble-learning-5-main-approaches.html (accessed on 8 December 2022).
49. Aksoy, A.; Ertürk, Y.E.; Erdoğan, S.; Eyduran, E.; Tariq, M.M. Estimation of honey production in beekeeping enterprises from eastern part of Turkey through some data mining algorithms. *Pak. J. Zool.* **2018**, *50*, 2199–2207. [CrossRef]
50. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
51. Cho, K.; van Merriënboer, B.; Bahdanau, D.; Bengio, Y. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*; Association for Computational Linguistics, Stroudsburg, PA, USA, 2014; pp. 103–111. [CrossRef]
52. Brownlee, J. Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions. *Mach. Learn. Mastery Python* **2018**, *1*, 539.

53. Merkle, F.H.; Discher, C.A. Controlled-Potential Coulometric Analysis of N-Substituted Phenothiazine Derivatives. *Anal. Chem.* **1964**, *36*, 1639–1643. [CrossRef]
54. Shi, R.; Jiang, C. Three-Way Ensemble Prediction for Workload in the Data Center. *IEEE Access* **2022**, *10*, 10021–10030. [CrossRef]