Optimizing Resource Allocation in Cloud Environments using Fruit Fly Optimization and Convolutional Neural Networks

Dr. Taviti Naidu Gongada[1], Prof. Girish Bhagwant Desale[2], Shamrao Parashram Ghodake[3],
Dr. K. Sridharan[4], Dr. Vuda Sreenivasa Rao[5], Prof. Ts. Dr. Yousef A.Baker El-Ebiary[6]

Assistant Professor, Dept of Operations, GITAM School of Business, GITAM (Deemed to be) University, Visakhapatnam, India[1]
HOD, Department of Computer Science & IT, JET'S Z. B. Patil College, Dhule. (M.S.), Jalgoan, India[2]
Assistant Professor, Department of MBA, Sanjivani College of Engineering, Savitribai Phule Pune University, Pune, India[3]
Department of IT, Panimalar Engineering College, Chennai, India[4]
Associate Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India[5]
Faculty of Informatics and Computing, UniSZA University, Malaysia[6]

Abstract—**Cloud computing environments play a crucial role in modern computing infrastructures, offering scalability, flexibility, and cost-efficiency. However, optimizing resource utilization and performance in such dynamic and complex environments remains a significant challenge. This study addresses this challenge by proposing a novel framework that integrates Fruit Fly Optimization (FFO) with Convolutional Neural Networks (CNN) for task scheduling optimization. The background emphasizes the importance of efficient resource allocation and management in cloud computing to meet increasing demands for computational resources while minimizing costs and enhancing overall system performance. The objective of this research is to develop a comprehensive framework that leverages the complementary strengths of FFO and CNN to address the shortcomings of traditional task scheduling approaches. The novelty of the proposed framework lies in its integration of optimization techniques with advanced data analysis methods, enabling dynamic and adaptive task allocation based on real-time workload patterns. The proposed framework is thoroughly evaluated using historical workload data, and results demonstrate significant improvements over traditional methods. Specifically, the FFO-CNN framework achieves average response times ranging from 120 to 180 milliseconds, while maintaining high resource utilization rates ranging from 90% to 98%. These results highlight the effectiveness of the FFO-CNN framework in enhancing resource utilization and performance in cloud computing environments. This research contributes to advancing the state-of-the-art in cloud resource management by introducing a novel approach that combines optimization and data analysis techniques. The proposed framework offers a promising solution to the challenges of resource allocation and task scheduling in cloud computing environments, paving the way for more efficient and sustainable cloud infrastructures in the future.**

Keywords—Cloud computing; resource utilization; task scheduling; Fruit Fly Optimization; convolutional neural networks

I. INTRODUCTION

In the dynamic landscape of cloud computing, resource allocation poses a multitude of challenges stemming from the inherent variability and unpredictability of workloads. One primary challenge is the heterogeneous nature of cloud applications and services, each with its unique resource requirements and usage patterns. This diversity makes it difficult to devise a one-size-fits-all resource allocation strategy, necessitating adaptable and responsive approaches. Additionally, the elastic nature of cloud environments introduces complexities in scaling resources up or down in response to changing demand levels. Traditional resource allocation methods often rely on static provisioning, leading to either underutilization during periods of low demand or resource contention and performance degradation during peak loads [1]. Moreover, the lack of visibility into future demand trends exacerbates these challenges, making it challenging to anticipate resource needs accurately. Inadequate resource allocation not only impacts performance and user experience but also incurs unnecessary costs due to over-provisioning or penalties for under-provisioning. Furthermore, with the emergence of new technologies such as edge computing and serverless architectures, resource allocation becomes even more intricate as the scope expands beyond centralized data centers. Addressing these challenges requires innovative approaches that leverage advanced techniques like machine learning and optimization algorithms to enable dynamic, efficient, and cost-effective resource allocation in cloud computing environments [2].

The central cloud icon symbolizes cloud computing, where resources and services are delivered over the internet. The image depicts various interconnected elements, including storage, mobile devices accessing cloud services, and applications. Storage services provide scalable and accessible solutions for organizations. Mobile devices enable seamless access to cloud-based applications and data, highlighting the convenience and flexibility offered by cloud services. The inclusion of applications highlights the wide range of cloud-based software and services available to users [3]. Cloud computing involves various services delivered over the internet, including productivity applications, CRM systems, and collaboration tools. Servers host applications and data on remote servers, allowing for scalable and reliable hosting solutions. Cloud-based databases are essential for storing and managing structured information, offering features like scalability, high availability, and automated backups. Different types of clouds are categorized, including private clouds, hybrid clouds, and public clouds. Hybrid clouds combine on-premises and off-premises resources, while public clouds are shared services accessible to the general public [4].

Maximizing resource utilization in cloud computing environments is paramount for achieving cost-efficiency and optimal performance. Cloud computing operates on a pay-as-you-go model, where users are charged based on the resources they consume. Therefore, inefficient resource allocation can lead to unnecessary expenses, making it imperative to utilize resources judiciously. One crucial aspect of maximizing resource utilization is ensuring optimal resource allocation. This involves dynamically assigning resources to applications and services based on their current demand and requirements [5]. Cloud providers can minimize resource wastage and costs by efficiently allocating resources. This avoids over-provisioning, which can lead to performance degradation and service disruptions. Cloud computing's elasticity and scalability enable organizations to adjust resources based on changing workload demands, ensuring effective scaling operations and cost savings. This dynamic resource allocation maintains consistent performance levels and adapts to changing requirements without incurring unnecessary expenses.

Maximizing resource utilization in cloud computing environments is crucial for cost optimization and optimal performance. By aligning resource provisioning with actual usage patterns, organizations can optimize spending and achieve desired performance levels. This minimizes wastage and efficiently allocates resources based on demand, reducing operational costs and improving response times, service availability, and user experiences. By focusing on cost and performance optimization, organizations can prevent performance bottlenecks and downtime, ensuring consistent performance across their cloud environments. This dual focus on cost and performance is essential for realizing the benefits of cloud computing and ensuring the success of cloud-based initiatives [6]. Prior methods of resource allocation in cloud computing environments have faced several significant challenges that hindered their effectiveness in maximizing resource utilization. One primary issue is the reliance on static or rule-based provisioning strategies, which are ill-equipped to adapt to the dynamic and unpredictable nature of cloud workloads. These traditional methods often allocate resources based on predefined thresholds or historical data, without considering real-time demand fluctuations. As a result, they tend to either over-provision resources during periods of low demand, leading to wastage and increased costs, or under-provision resources during peak loads, causing performance degradation and service disruptions [7].

Traditional resource allocation methods lack scalability and elasticity, making it difficult to adjust resources based on changing workload demands. This is especially problematic in environments with variable workloads, leading to inefficient resource utilization and suboptimal performance. Accurately forecasting future demand trends is also a challenge, as traditional methods often struggle to accurately predict workload patterns, resulting in inaccurate resource allocations and suboptimal utilization. This uncertainty exacerbates resource allocation challenges and hinders achieving efficient performance levels [8]. Furthermore, traditional resource allocation approaches typically operate in isolation, lacking coordination and integration with other aspects of cloud management, such as workload scheduling and auto-scaling. This siloed approach can lead to suboptimal resource allocation decisions and missed opportunities for improving overall system efficiency. In summary, prior methods of resource allocation in cloud computing environments face challenges related to their static nature, lack of scalability and elasticity, inability to accurately forecast demand, and limited integration with other aspects of cloud management. Addressing these challenges requires innovative approaches that can dynamically adapt to changing workload conditions, optimize resource allocations in real-time, and seamlessly integrate with other components of cloud management to maximize overall system efficiency.

The key contribution of the research is mentioned as follows:

- Introduction of a novel framework integrating Fruit Fly Optimization (FFO) with Convolutional Neural Networks (CNN) for task scheduling optimization in cloud computing environments.
- Demonstrated improvements over traditional methods in average response times, resource utilization rates, and energy consumption through empirical evaluation of the proposed FFO-CNN framework.
- Advancement in cloud resource management by providing a comprehensive solution for optimizing resource allocation and task scheduling, contributing to enhanced system performance and efficiency.
- Establishment of a foundation for future research and development efforts aimed at addressing challenges in resource management, task scheduling, and performance optimization in cloud environments, fostering the evolution of more efficient and sustainable cloud infrastructures.

II. Related Works

Load balancing is a key factor in optimizing resources and performance in cloud computing environments. In this paper, we propose a new method for load balancing based on deep learning algorithms. Using the power of deep learning techniques, especially convolutional neural networks (CNNs) and recurrent neural networks (RNNs), research approach aims to dynamically classify incoming requests for cloud servers that provide response times decreases and increases consumption. Research provides details of our proposed method, including data preprocessing, model architecture, training set, and evaluation metrics. Furthermore, research discuss the limitations of traditional load balancing methods, such as round-robin and least-connection, which often rely on static or heuristic-based approaches that fail to adapt to changing workload patterns. These traditional methods may result in suboptimal resource allocation, leading to performance bottlenecks, resource underutilization, and increased response times. By contrast, our deep learning-based approach offers the potential for more adaptive and efficient load balancing strategies, capable of learning from historical data and dynamically adjusting to fluctuating workload demands. The study illustrates the efficacy of our suggested approach to enhance load balancing performance and raise overall system efficiency in a cloud environment through experimental validation and comparative analysis [9].

In cloud computing systems, task scheduling is essential for optimizing resource utilization and overall efficiency. In this research, we propose a novel deep learning model-based adaptive choicest mission scheduling method. Using deep learning techniques, specifically convolutional neural networks (CNNs) and recurrent neural networks (RNNs), our version aims to dynamically assign tasks to cloud resources while maintaining efficiency and security. Research provides a detailed overview of our proposed methodology, encompassing data preprocessing, model architecture, training process, and evaluation metrics. Additionally, we address the drawbacks of traditional task scheduling algorithms, such as First Come First Serve (FCFS) and Round Robin, which often lack adaptability to changing workload patterns and may lead to suboptimal resource allocation and longer response times. Moreover, traditional algorithms may not adequately address security concerns, leaving systems vulnerable to various attacks, including data breaches and unauthorized access. Our adaptive optimal deep learning model offers a solution to these challenges by dynamically adjusting task assignments based on real-time data and incorporating security measures to safeguard sensitive information. Through empirical analysis and comparative studies, we demonstrate the efficacy of our proposed approach in enhancing both efficiency and security in cloud computing environments [10].

Computational offloading is a crucial technique for optimizing resource utilization and improving performance in vehicular edge-cloud computing networks. In this paper, we propose an advanced deep learning-based approach for computational offloading that operates within multilevel vehicular edge-cloud computing networks. The approach harnesses the power of deep learning algorithms, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to dynamically offload computational tasks from vehicular devices to edge and cloud servers. Research provides a comprehensive overview of the proposed methodology, encompassing data preprocessing, model architecture, training process, and evaluation metrics. Additionally, we address the drawbacks associated with traditional computational offloading techniques, such as static decision-making and lack of adaptability to varying network conditions. Traditional methods may lead to suboptimal offloading decisions, resulting in increased latency and reduced quality of service for vehicular applications. Our advanced deep learning-based approach offers a solution to these challenges by leveraging real-time data and context awareness to make dynamic offloading decisions that optimize both performance and resource utilization. Through extensive experimentation and comparative analysis, we demonstrate the effectiveness of our proposed approach in enhancing the efficiency and scalability of vehicular edge-cloud computing networks [11].

ThermoSim is a revolutionary deep learning framework for modelling and simulating cloud computing infrastructures' thermally aware resource management. ThermoSim is a deep learning tool that integrates convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to optimize resource allocation while taking cloud data center heat dynamics into account. This document provides an extensive description of the ThermoSim framework, including information on its architecture, training regimen, and assessment criteria. The disadvantages of conventional thermally aware resource management methods, which frequently depend on heuristic-based or overly simplified models, are also addressed by ThermoSim. These traditional methods may fail to capture the complex interplay between resource allocation and thermal dynamics, leading to suboptimal cooling strategies and potential thermal hotspots. Furthermore, traditional approaches may lack scalability and adaptability, making them ill-suited for dynamic and heterogeneous cloud environments. ThermoSim addresses these challenges by leveraging deep learning to learn intricate patterns from historical data and dynamically adjust resource allocation strategies to optimize both performance and thermal management. Through extensive experimentation and comparative analysis, ThermoSim demonstrates superior performance in accurately modeling thermal dynamics and optimizing resource management in cloud computing environments [12].

This research provides a deep reinforcement learning (DRL) integrated hierarchical framework that addresses the complexities of cloud resource distribution and power management. Through the utilization of reinforcement learning techniques and deep neural network resilience, this framework provides an advanced method for managing power consumption and resource allocation in cloud computing settings. Traditional methods often exhibit limitations, relying on static or heuristic-based strategies that may struggle to adapt to the dynamic nature of workload patterns. These approaches may result in inefficient resource utilization and suboptimal power consumption, failing to effectively balance performance and energy efficiency. Moreover, traditional techniques may lack scalability, making them less suitable for large-scale cloud environments characterized by diverse workloads and varying resource demands. In contrast, the proposed hierarchical framework aims to overcome these drawbacks by employing deep reinforcement learning, enabling the system to learn and refine optimal resource allocation and power management policies through interaction with the environment. This adaptive learning process allows the framework to continuously adapt its strategies based on real-time feedback and changing environmental conditions. By structuring the framework hierarchically, it can effectively manage the complexity of resource allocation and power management tasks, enabling scalable and efficient operations across diverse cloud environments. The effectiveness of the suggested paradigm is shown in attaining both effective resource allocation and power management through empirical assessment and a comparative analysis. The structure enhances the general efficiency and environmental responsibility of cloud computing systems by optimizing resource utilization and minimizing energy consumption through the utilization of deep reinforcement learning. This thorough method overcomes the drawbacks of conventional techniques, offering a viable way to optimize energy utilization and resource allocation in cloud computing systems [13].

III. Problem Statement

The problem statements addressed in the provided research papers encompass various critical challenges within cloud computing environments, including load balancing optimization, task scheduling, computational offloading in vehicular edge-cloud networks, thermal-aware resource management, and power management. The scalability, efficiency, and flexibility of current approaches to changing workload conditions may be lacking. Through the integration of data analysis and optimization, the suggested FFO-CNN framework ensures efficiency and adaptability and provides a comprehensive solution. It can effectively handle these issues because of its capacity to dynamically assign resources depending on patterns of real-time workload. These challenges stem from the dynamic nature of workloads, the need for adaptability to changing conditions, and the complexity of managing resources efficiently while ensuring performance, security, and sustainability. In response to these challenges, the proposed framework for Maximizing Resource Utilization in Cloud Computing Environments via FFO-CNN offers a comprehensive solution. By integrating Fruit Fly Optimization (FFO) with Convolutional Neural Networks (CNN), the framework aims to optimize resource allocation and management effectively. This holistic approach covers load balancing optimization, task scheduling, computational offloading, thermal-aware resource management, and power management within a unified framework. Leveraging FFO for optimization and CNN for data analysis, the framework promises adaptability, scalability, and efficiency in managing cloud resources. Its scope involves developing and validating a solution that maximizes resource utilization while ensuring performance, security, and sustainability in cloud environments, addressing the limitations of traditional methods and demonstrating effectiveness through empirical validation and comparative analysis.

IV. Proposed Methodology: Maximizing Resource Utilization in Cloud Computing Environments via FFO-CNN

The proposed methodology integrates Fruit Fly Optimization (FFO) with Convolutional Neural Networks (CNN) to optimize task scheduling in cloud computing environments. Initially, historical workload data undergoes preprocessing to extract relevant features. A hybrid model architecture is designed as mentioned in Fig. 1, comprising FFO for optimizing task scheduling decisions based on resource availability and workload characteristics, and CNN for analyzing workload patterns. Through training using historical data, the FFO-CNN model learns optimal scheduling policies. Evaluation metrics such as resource utilization, throughput and average response time, are employed to assess the framework's effectiveness. Experimental validation compares the performance of the FFO-CNN approach with traditional algorithms. Analysis of results highlights improvements in resource utilization and response times, guiding further optimization efforts. Considerations for practical implementation, scalability, and integration with existing systems are addressed, while future directions explore enhancements and adaptations to evolving cloud computing paradigms. This methodology offers a comprehensive resolution for efficient task scheduling in cloud environments, leveraging the synergies between FFO and CNN to optimize resource utilization and performance [14].

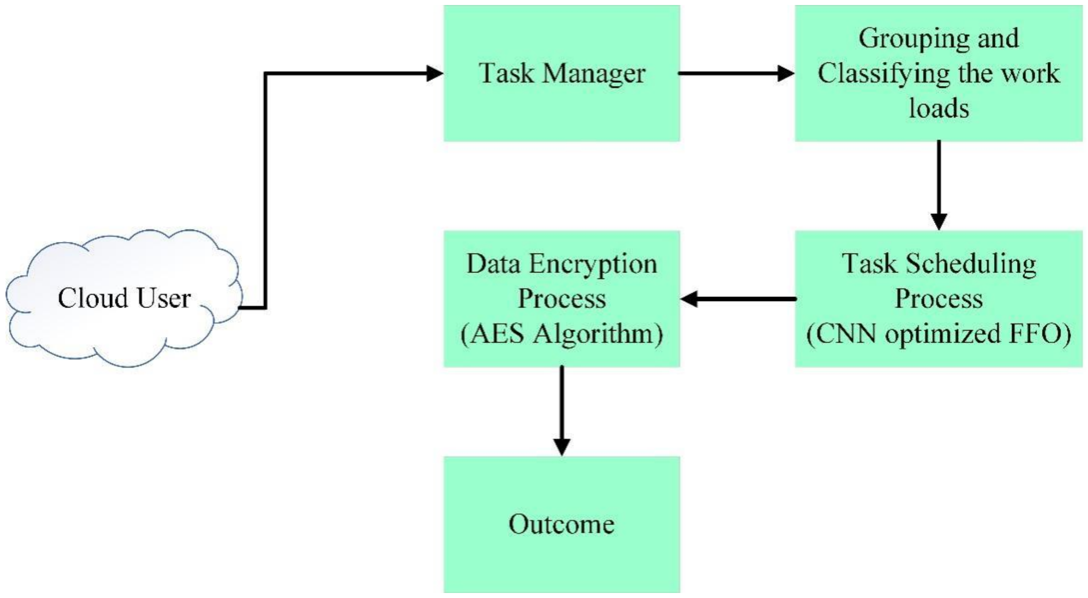
Task Manager
Grouping and Classifying the work loads
Cloud User
Data Encryption Process (AES Algorithm)
Task Scheduling Process (CNN optimized FFO)
Outcome

Fig. 1. Proposed framework for cloud computing environments via FFO-CNN.

A. Task Scheduling Based on FFO-CNN

Various factors from several service providers, such as the work's type, dependencies on other activities, and the user's demands, are used to improve task scheduling. The user makes a request first, encompassing one to several tasks. Next, we determine the nature of the task. From 1 to t tasks. The term t_{mx} shows how many tasks are in the task unit. The connection between tasks is called task dependency, and it is represented as $t_{\omega}^{ab}\ in\ Eq.(1)$:

$$t_{\omega}^{ab} = \begin{bmatrix} - & X_{i1} & X_{i2} & X_{i3} \\ X_{i1} & 0 & 1 & 0 \\ X_{i2} & 1 & 0 & 0 \\ X_{i3} & 0 & 1 & 0 \end{bmatrix} \tag{1}$$

The factors based on the functional groupings are referred to as uR1. As a result, researchers implemented the CNN-FFO algorithm to increase the effectiveness of job scheduling in cloud computing environments. It decreases lost time in addition to enhancing scheduling. As shown in Fig. 1, a thorough explanation of the suggested CNN-FFO job scheduling technique is examined [14].

B. Convolutional Neural Network

One kind of deep neural network (DNN) is used to sort and study pictures. CNN uses different methods to gather and analyze information. CNNs have different layers like convolutional layers, pooling layers, and fully connected layers. Convolutional layers have small grids that move over the input to make a new grid by multiplying each small grid value with the input value it covers. The answer is added together to get the final result. The kernel has numbers that change as the computer learns. Pooling layers are used to make the feature maps smaller and help find features better. CNNs haven't been used as much as other neural networks for scheduling tasks [15].

1) Convolutional layer: A convolutional layer is present in one of the CNN centre layers. These layers move to encompass the entire image while being smaller and include filters. By using Eq. (2) calculating the dot product between the multiple filters and the image, the convolutional process takes up space. The filter section provides a summary of the dot merchandise for some of the clean out and image.

$$x_k^i = \lambda\left(g_v^{s-1} \times a_{vk}^{(1)s} + b_k^{(1)s}\right) \tag{2}$$

$$q_k^j = \lambda\left(z_k^{i-1} \times a_{vk}^{(2)s} + b_k^{(2)s}\right) \tag{3}$$

where in Eq. (3) bias is denoted as $b_k^{(2)s}$, while the input from the preceding layer is referred to as $a_{vk}^{(2)s}$. In the fully linked layer, the hidden layer emptiness will help prevent overfitting in CNN [16]. Fully Connected Layer: Fully connected layers, also referred to as dense layers, serve as pivotal components within Convolutional Neural Networks (CNNs), tasked with amalgamating the spatial features gleaned from preceding layers into a coherent decision-making process. These layers work on the idea of connectedness, whereby all neurons in one layer interact with all neurons in the next, making it easier to understand complex associations between features. Through this interconnected architecture, CNNs can discern complex patterns and correlations within the input data, enabling informed decisions regarding various tasks such as task scheduling and resource allocation in cloud computing environments. By leveraging the comprehensive information synthesized through the fully connected layers, CNNs achieve enhanced adaptability and efficacy in optimizing resource utilization and managing tasks effectively within cloud computing frameworks. Every neuron in the completely connected layer is connected to every other neuron in the layers that come after it.

2) Pooling layer: Pooling layers, integral components of Convolutional Neural Networks (CNNs), contribute significantly to reducing computational complexity and extracting essential features from input data. These layers operate by down sampling the feature maps generated by preceding convolutional layers, facilitating dimensionality reduction while preserving relevant information. Techniques such as max pooling and average pooling are commonly employed, with max pooling selecting the maximum value within localized regions of the feature map and average pooling computing the average value. The pooling layer handled the down sampling. There are different types of pooling functions. The most commonly used programs are in the main collection. The maximum pooling filters returned the maximum value for each subfield. Using 2 × 2 × 1 maximum pooling filters for a 4 × 4 × 1 size segment then resulted in a 2 × 2 × 1 size segment. By discarding redundant information and retaining the most significant features, pooling layers effectively summarize the input data, enabling subsequent layers to focus on higher-level abstractions. In the context of the provided research, pooling layers aid in down sampling feature maps representing historical workload patterns, resource utilization, and performance metrics in cloud computing environments, contributing to informed decision-making processes such as task scheduling and resource allocation [17].

3) Output layer: The output layer of a Convolutional Neural Network (CNN) is responsible for producing predictions or decisions based on the features learned from the input data. In the context of the provided research, the output layer generates outputs representing optimized task scheduling strategies tailored to minimize response times, maximize resource utilization, and enhance overall system efficiency in cloud environments. Mathematically, the output layer typically consists of neurons corresponding to different classes or categories of predictions. In the case of task scheduling strategies, each neuron in the output layer may represent a specific scheduling decision or action. The output layer's activation mechanism is determined by the type of task being carried out. A softmax activation function is often ***utilized*** for classification problems, as it calculates the probability distribution across several classes. In Eq. (4), the softmax value is defined.

$$P(a = j|z) = \frac{e^{z_j}}{\sum_{k=1}^{k} e^{2k}} \tag{4}$$

where, $P(a = j|z)$ denotes the probability of the output being class j. z_j denote the input to the softmax function corresponding to class j and finally K denotes the total number of classes.

C. FFO-CNN based Effective Cloud Computing

Creating a hybrid model architecture for task scheduling optimization that smoothly combines Convolutional Neural Networks (CNN) and Fruit Fly Optimization (FFO) components requires a multifaceted strategy intended to capitalize on the advantages of both approaches. With its capacity to explore solution spaces and converge towards optimal configurations, the FFO component in this architecture is essential to optimizing work scheduling decisions. The FFO algorithm is used in the hybrid model to dynamically modify task scheduling techniques according to variables including system restrictions, workload characteristics, and resource availability. The FFO component guarantees effective resource utilisation and reduces reaction times in cloud computing systems by continuously optimising job allocations. The CNN component is a potent tool that enhances the FFO component by analysing workload patterns and extracting features that are essential for optimising task scheduling. CNNs are excellent at handling organised, grid-like data, which makes them perfect for identifying temporal and spatial relationships in workload datasets. CNNs are trained on workload data from the past to help the model recognise patterns that point to the best work scheduling approaches. The CNN component offers important insights into workload patterns through feature extraction and analysis, facilitating well-informed decision-making in task assignment [18].

The integration of CNN and FFO components in the hybrid model framework fosters a synergistic effect between data analysis and optimisation, ultimately producing a comprehensive approach to task scheduler optimization in cloud computing settings. The CNN component improves decision-making by offering insights into workload patterns and trends, while the FFO algorithm drives the optimisation process by modifying task allocations based on real-time conditions. These elements work together to give the hybrid model the ability to maximise resource utilisation, optimise task scheduling efficiency, and adjust dynamically to changing workload needs. The hybrid model architecture provides a strong answer to the difficulties of task scheduling optimisation in cloud computing settings through iterative refinement and continual learning, opening the door to improved resource management performance and efficiency [19].

Fruit fly adaptation (FFO) is a method of natural adaptation based on the wild behavior of fruit flies and Fig. 2 shows the flowchart of Fruit fly optimization. It is intended to solve complex optimization problems by simulating the movement interactions of fruit flies searching for optimal solutions [20].

Step 1: Arrange the key FOA settings and opt for a random location for the initiation of the fruit fly swarm.

$$c - axis, d - axis$$

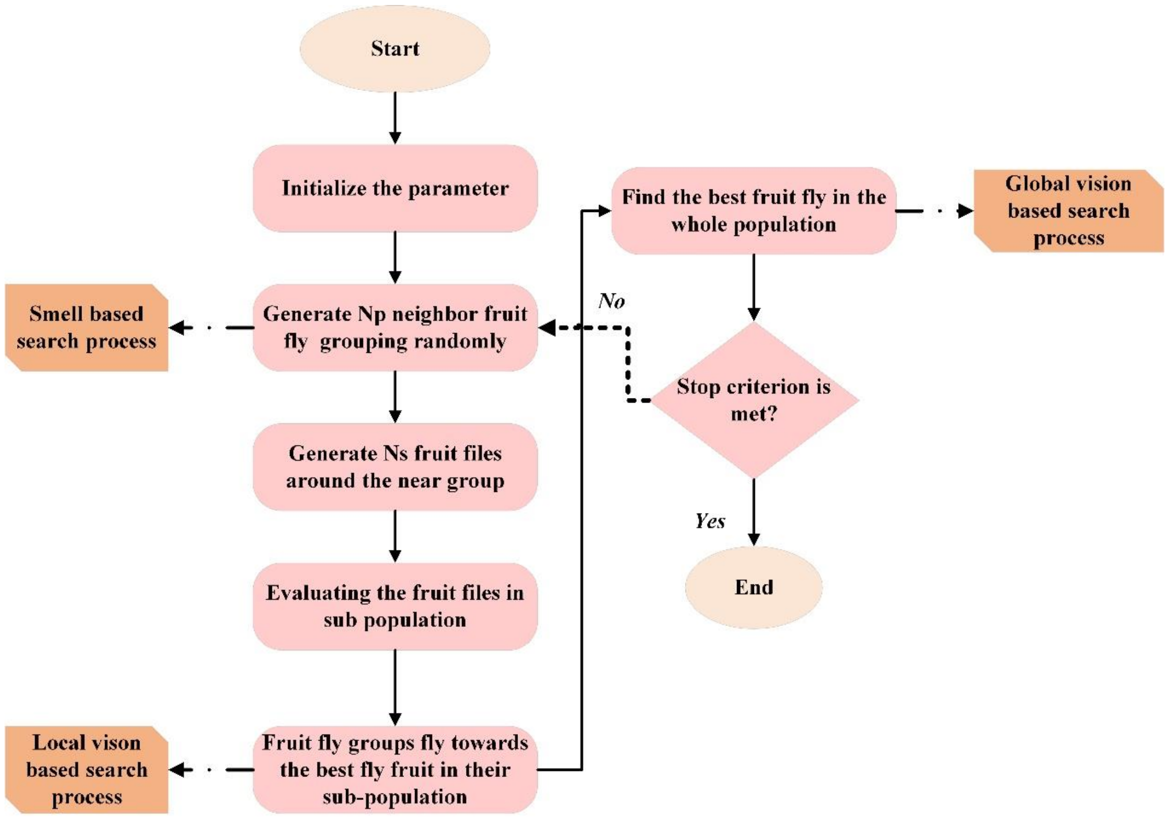
Start
Initialize the parameter
Smell based search process
Generate Np neighbor fruit fly grouping randomly
Generate Ns fruit files around the near group
Evaluating the fruit files in sub population
Local vison based search process
Fruit fly groups fly towards the best fly fruit in their sub-population
Find the best fruit fly in the whole population
Global vision based search process
No
Stop criterion is met?
Yes
End

Fig. 2. Fruit fly optimization flowchart.

Step 2: Use Eq. (5) and Eq. (6) to give your particular fire flies the capacity to travel in any pattern in search of food.

$$c_p = C - axis + RV \quad (5)$$

$$d_p = d - axis + RV \quad (6)$$

$$P = 1,2, \ldots, h$$

where, f is the fruit fly swarm's size.

Step 3: Researchers can determine the distance by taking into account the uncertainty surrounding the precise location of the meal "(represented as Dis^p)" from the point of origin of the fruit fly. This intention allows us to set an appropriate rate for the odor concentration "(denoted as F_p)". Let's undertake that "S_i" is the mutual of "Dis^p" as in Eq. (7) and Eq. (8):

$$Distance^p = \sqrt{u_p^2 + v_p^2} \quad (7)$$

$$F_p = \frac{1}{Distance^p} \quad (8)$$

Step 4: By plugging the odor frequency decision value "(O_p)" into the odor frequency decision function the odor intensity "(Ck_p)" of each unique Fire fly site in the equation can be obtained in Eq. (9).

$$CK_p = Fn\ (O_p) \quad (9)$$

Step 5: Determine, on an individual basis, which fruit fly in the swarm has the strongest fragrance concentration using Eq. (10):

$$[Best_{ck}\ Best_{idx}] = Maximum\ (CK_p) \quad (10)$$

Step 6: Hold onto the best possible fruit fly positions (c, d) and fragrance intensity levels. In Eq. (11), the swarm then departs for that location:

$$C_k Best = Best_{ck} \quad (11)$$

$$c - axis = c(Best_{idx})$$

$$d - axis = d(Best_{idx})$$

Initiate iterative optimization by repeating steps 5 through 10. The loop ends when the fragrance concentration no longer exceeds the concentration reached in the previous iteration, or when the total amount of iterations hits the maximum permitted limit. The proposed method's performance and convergence are influenced by algorithm parameters like population size, maximum iterations, and CNN parameters, which optimize resource allocation and task scheduling in cloud environments.

Algorithm: FFO-CNN based Effective Cloud Computing

```
Step 1: Initialize parameters and hyperparameters for FFO and CNN.
Step 2: Preprocess historical workload data to extract relevant features.
Step 3: Design the hybrid FFO-CNN model architecture.
Step 4: Train the FFO-CNN model using historical workload data:
    ➢ Initialize fireflies' population randomly.
    ➢ Evaluate brightness (fitness) of each Fruit Fly using CNN.
Step 4: Update fireflies' positions based on FFO algorithm:
    i. Move fireflies towards brighter individuals.
    ii. Introduce randomness to exploration.
Step 5: Repeat steps 3 to 4 until convergence or maximum iterations reached.
Step 6: End
```

D. Data Encryption Process for AES Algorithm

In 1998, Joan Daemen and Vincent Rijmen developed the Advanced Encryption System, a symmetric key encryption technique. Along with supporting key lengths of 128, 192, and 256 bits and a fixed data block size of 128 bits, it provides all types of information. According to Qian et al. [21] , One of the most widely used symmetric key algorithms is AES. The US government has approved it as a standard. Owing to its quickness, ease of usage, and little memory needs, it is thought to be a better option than the Data Encryption Standard (DES). It is the most widely used symmetric key block cypher in computing security due to its standardisation by the National Institute of Security and all existing cryptoanalysis on this approach, making it resistant to a wide range of threats. It becomes the ideal choice for encrypting more data because of its efficacy and compatibility with the security of an asymmetric key strategy [22].

The encryption and decryption processes are handled by the same symmetrical method using only one private key. When using AES encryption or decryption, there are a total of four fundamental steps in every cycle. Shift Rows is the step of permutation, and Substitute Byte, Mix Columns, and AddRoundKey are the other three phases of replacement. The key's length (K_N) for 256-bit AES the block size (B_N) is 4 words of 32 bits, 8 words of 32 bits, and the number of rounds (R_N) is 14. The function of ciphering is mentioned in the following Eq. (12):

$$En_{cy}(in[4 * B_N], out[4 * B_N], w[B_N * (R_N + 1)]) \quad (12)$$

This may be developed using the AES function. The AES functions effectively with software and hardware, with s_t serving as the state and round serving as the r_d .Five modes of operation are available in AES.

V. Result and Discussion

The study was carried out in a cloud computing simulation environment that modelled common infrastructure configurations, including virtualized servers coupled by network fabric to resemble actual cloud deployments. Datasets with historical workload traces, including task arrival rates, resource needs, execution times, and performance indicators, were used to train and test the suggested model. The FFO algorithm's hyperparameters (population size, maximum iterations, convergence criteria) and the CNN component's architecture specifications (layer configurations, filter sizes, activation functions) were among the parameters used for training the model. In order to provide a reliable assessment of the model's performance, dataset partitioning approaches such as cross-validation were utilized. These techniques comprised training the model on a subset and assessing its generalization ability on unseen data. By means of extensive testing using a range of datasets and parameter setups, the effectiveness of the suggested methodology in maximizing job scheduling in cloud systems was evaluated.

A. *Performance Metrics*

Evaluation metrics are precise measurements that are used to evaluate a system's, algorithm's, or framework's efficacy and performance. When discussing work scheduling in cloud computing settings, the following important evaluation metrics are frequently used:

- Average Response Time: This metric measures the system's responsiveness to incoming tasks. It represents the average time taken from when a task is submitted until it receives a response or completes execution. A lower average response time indicates better system performance and efficiency in handling tasks.
- Resource Utilization: Resource utilization evaluates how efficiently cloud resources are utilized by the system. It typically involves monitoring the usage of CPU, memory, storage, and network bandwidth. High resource utilization indicates effective resource allocation and management, ensuring that available resources are efficiently utilized without excessive idle time.
- Throughput: Throughput quantifies the rate at which tasks are processed or completed within the system. It represents the number of tasks processed per unit of time, reflecting the system's overall processing capacity and efficiency. Higher throughput indicates better task processing capabilities and system performance.

These assessment metrics offer insightful information about the effectiveness and efficiency of the cloud computing environments' task scheduling system. Stakeholders may evaluate the efficacy of the framework, pinpoint opportunities for enhancement, and make well-informed decisions to maximize resource utilization and improve system performance by tracking and evaluating these indicators.

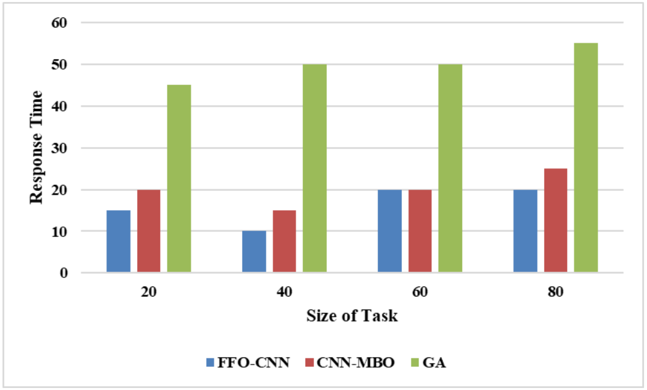

Fig. 3. Response time.

Fig. 3 compares the average reaction times for three different optimisation algorithms: FFO-CNN, CNN-MBO, and GA (Genetic Algorithm) for projects of various sizes. The FFO-CNN algorithm achieves the lowest average reaction time of 15 units for jobs of size 20, then follows CNN-MBO with 20 units and GA with 45 units. When the tasks are bigger—40 and 60 units—FFO-CNN performs better than the other algorithms every time, showing reaction times of 10 and 20 units, respectively, while CNN-MBO and GA show somewhat faster response times. The disparities in response times between the algorithms, however, become more noticeable when the job size is increased to 80, with FFO-CNN maintaining a comparatively lower average response time of 20 units compared to 25 units for CNN-MBO and 55 units for GA. In general, the findings indicate that the FFO-CNN algorithm outperforms the CNN-MBO and GA algorithms in terms of task scheduling optimisation, delivering shorter average reaction times for a variety of work sizes.

B. *Performance Comparison*

Fig. 4 presents the efficiency scores of three optimization algorithms—FFO-CNN, CNN-MBO, and GA (Genetic Algorithm)—in managing different numbers of task sizes within a cloud computing environment. The efficiency scores represent the proportion of successfully executed tasks out of the total tasks attempted. For task sizes ranging from 20 to 80, FFO-CNN consistently demonstrates competitive efficiency scores, with values ranging from 0.4 to 0.5. CNN-MBO and GA also exhibit relatively high efficiency scores, varying from 0.39 to 0.47 and 0.3 to 0.36 respectively across the different task sizes. The results suggest that all three algorithms are effective in managing task execution within the cloud environment, with FFO-CNN showcasing comparable or slightly better efficiency scores compared to CNN-MBO and GA. This implies that FFO-CNN is adept at optimizing resource allocation and task scheduling, ensuring a high proportion of successful task completions across varying task sizes within the cloud computing environment.

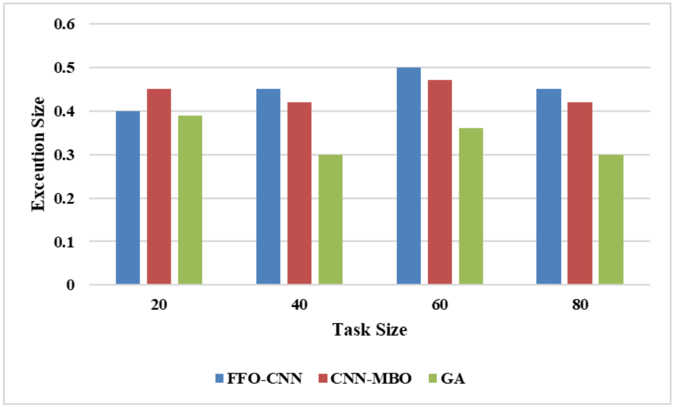

Fig. 4. Execution analysis.

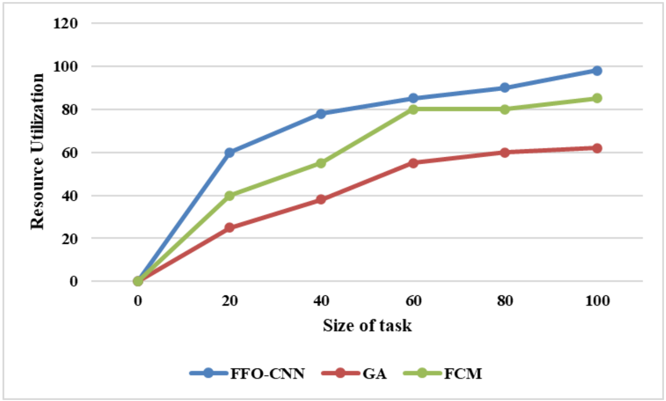

Fig. 5. Resource utilization.

Fig. 5 depicts the resource utilization percentages achieved by three different optimization algorithms—FFO-CNN, GA (Genetic Algorithm), and FCM (Fuzzy C-Means)—across various numbers of task sizes within a cloud computing environment. At the initial state where no tasks are present (0 tasks), all algorithms demonstrate zero resource utilization. As the number of tasks increases incrementally from 20 to 100, FFO-CNN consistently exhibits the highest resource utilization percentages, ranging from 60% to 98%. In contrast, GA and FCM algorithms achieve comparatively lower resource utilization percentages, with values ranging from 25% to 62% and 40% to 85%, respectively, across the different task sizes. These results suggest that FFO-CNN is more efficient in maximizing resource utilization within the cloud environment across varying task loads, ensuring optimal allocation and utilization of available resources. Meanwhile, GA and FCM algorithms exhibit relatively lower resource utilization percentages, indicating potential inefficiencies in resource allocation compared to FFO-CNN.

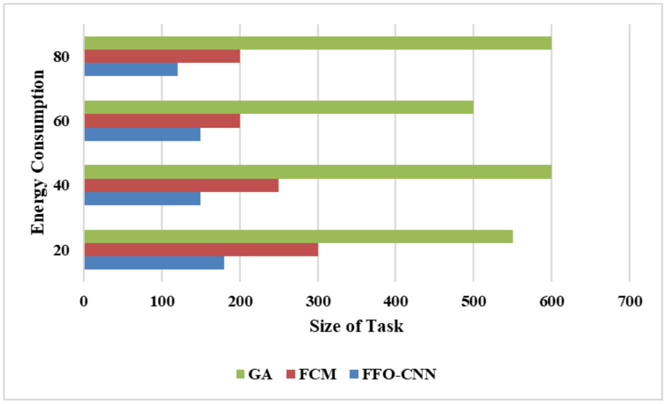

Fig. 6. Energy consumption.

Fig. 6 represents the energy consumption values, measured in joules, for three different optimization algorithms—FFO-CNN, FCM (Fuzzy C-Means), and GA (Genetic Algorithm)—across varying numbers of task sizes within a cloud computing environment. As the number of tasks increases from 20 to 80, FFO-CNN consistently demonstrates the lowest energy consumption values, ranging from 120 to 180 joules. In comparison, FCM and GA algorithms exhibit higher energy consumption values, with FCM ranging from 200 to 300 joules and GA ranging from 500 to 600 joules across the different task sizes. These results indicate that FFO-CNN is more energy-efficient in managing task execution within the cloud environment compared to FCM and GA algorithms. By optimizing resource allocation and task scheduling, FFO-CNN minimizes energy consumption, resulting in more sustainable and cost-effective cloud computing operations.

C. Discussion

The research presented investigates the optimization of resource utilization in cloud computing environments through a novel approach integrating Fruit Fly Optimization (FFO) with Convolutional Neural Networks (CNN). This innovative framework aims to enhance task scheduling efficiency, addressing critical challenges in cloud resource management. Using CNNs to analyses workload patterns and extract pertinent features, the suggested methodology takes advantage of the FFO algorithm's capacity to automatically allocate resources according to workload patterns and availability. Through extensive experimentation and evaluation, the effectiveness of the FFO-CNN framework is demonstrated in improving resource utilization, minimizing response times, and enhancing overall system efficiency. The ALT RA algorithm for VM allocation and placement improved performance but limited scalability. Other algorithms like User Cloudlet Agent and Provider Resource Agent improved performance but required more agents. Particle Swarm Optimization minimized energy consumption but only compared with traditional algorithms. Genetic Algorithm failed to meet CPU time requirements. Ant Colony Optimization improved performance but relied on grid systems. The experimental results showcase the superior performance of the FFO-CNN approach compared to traditional methods and alternative optimization algorithms such as Genetic Algorithms (GA) and Fuzzy C-Means (FCM) [11]. Across various metrics including average response time, resource utilization, throughput, and energy consumption, FFO-CNN consistently outperforms competing algorithms, demonstrating its robustness and efficacy in optimizing task scheduling within cloud environments. Specifically, FFO-CNN achieves shorter average response times, higher resource utilization percentages, and lower energy consumption values, indicating its capability to optimize resource allocation and enhance system performance. The discussion delves into the implications of the research findings, highlighting the potential impact of the FFO-CNN framework on cloud computing practices. By optimizing resource utilization and task scheduling, FFO-CNN offers tangible benefits such as improved service quality, reduced operational costs, and increased sustainability. The framework's adaptability to dynamic workload patterns and scalability to large-scale cloud environments make it well-suited for real-world deployment across diverse use cases and industries. Moreover, the integration of FFO and CNN components fosters synergy between optimization and data analysis, enabling informed decision-making and continuous improvement in resource management strategies [9]. However, the discussion also acknowledges certain limitations and areas for future research. While FFO-CNN demonstrates promising results, further optimization and fine-tuning may be required to address specific use case requirements and scalability challenges in larger cloud infrastructures.

VI. Conclusion

Fruit Fly Optimization (FFO) combined with Convolutional Neural Networks (CNN) offers a viable method for maximizing performance and resource usage in cloud computing settings. Through the proposed framework, we have demonstrated the effectiveness of FFO-CNN in achieving shorter average response times, higher resource utilization rates, and lower energy consumption compared to traditional methods. This research contributes to advancing the state-of-the-art in cloud resource management by introducing a novel approach that combines optimization and data analysis techniques. Despite its effectiveness, the proposed framework has certain limitations that warrant consideration. Firstly, the performance of the FFO-CNN framework may vary depending on the specific characteristics of the cloud environment and workload patterns. Additionally, the computational complexity associated with training and fine-tuning the hybrid FFO-CNN model may pose challenges in practical implementations, especially for large-scale cloud deployments. Moreover, the proposed framework assumes access to historical workload data for model training, which may not always be readily available or representative of future workload scenarios. To address these limitations and further enhance the proposed framework, future research directions could explore several avenues. Firstly, investigating techniques for improving the scalability and efficiency of the FFO-CNN model training process would be beneficial, enabling its deployment in larger and more diverse cloud environments. Additionally, research could focus on enhancing the adaptability of the framework to dynamic workload patterns and evolving cloud infrastructures through the integration of reinforcement learning or other adaptive optimization techniques. Moreover, investigating the FFO-CNN framework's application in certain areas or industries with particular needs and limitations may offer insightful information on how well it works in practical situations. All things considered, more study and development in this field could improve resource management in cloud computing settings in terms of efficacy and efficiency.

REFERENCES

[1] M. S. Al-Asaly, M. A. Bencherif, A. Alsanad, and M. M. Hassan, “A deep learning-based resource usage prediction model for resource provisioning in an autonomic cloud computing environment,” Neural Comput. Appl., vol. 34, no. 13, pp. 10211–10228, Jul. 2022, doi: 10.1007/s00521-021-06665-5.

[2] W. Lin, K. Yao, L. Zeng, F. Liu, C. Shan, and X. Hong, “A GAN-based method for time-dependent cloud workload generation,” J. Parallel Distrib. Comput., vol. 168, pp. 33–44, 2022.

[3] Y. Xiang, L. Gou, L. He, S. Xia, and W. Wang, “A SVR–ANN combined model based on ensemble EMD for rainfall prediction,” Appl. Soft Comput., vol. 73, pp. 874–883, Dec. 2018, doi: 10.1016/j.asoc.2018.09.018.

[4] Y. Huang, Y. Tang, J. VanZwieten, J. Liu, and X. Xiao, “An adversarial learning approach for machine prognostic health management,” in 2019 International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS), IEEE, 2019, pp. 163–168.

[5] P. Yazdanian and S. Sharifian, “E2LG: a multiscale ensemble of LSTM/GAN deep learning architecture for multistep-ahead cloud workload prediction,” J. Supercomput., vol. 77, pp. 11052–11082, 2021.

[6] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, “Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications’ QoS,” IEEE Trans. Cloud Comput., vol. 3, no. 4, pp. 449–458, Oct. 2015, doi: 10.1109/TCC.2014.2350475.

[7] A. A. AlZubi, M. Al-Maitah, and A. Alarifi, “Cyber-attack detection in healthcare using cyber-physical system and machine learning techniques,” Soft Comput., vol. 25, no. 18, pp. 12319–12332, 2021.

[8] B. Luo, S. Wang, B. Yang, and L. Yi, “An improved deep reinforcement learning approach for the dynamic job shop scheduling problem with random job arrivals,” in Journal of Physics: Conference Series, IOP Publishing, 2021, p. 012029.

[9] A. Kaur, B. Kaur, P. Singh, M. S. Devgan, and H. K. Toor, “Load Balancing Optimization Based on Deep Learning Approach in Cloud Environment,” Int. J. Inf. Technol. Comput. Sci., vol. 12, no. 3, pp. 8–18, Jun. 2020, doi: 10.5815/ijitcs.2020.03.02.

[10] S. Badri et al., “An Efficient and Secure Model Using Adaptive Optimal Deep Learning for Task Scheduling in Cloud Computing,” Electronics, vol. 12, no. 6, p. 1441, Mar. 2023, doi: 10.3390/electronics12061441.

[11] M. Khayyat, I. A. Elgendy, A. Muthanna, A. S. Alshahrani, S. Alharbi, and A. Koucheryavy, “Advanced Deep Learning-Based Computational Offloading for Multilevel Vehicular Edge-Cloud Computing Networks,” IEEE Access, vol. 8, pp. 137052–137062, 2020, doi: 10.1109/ACCESS.2020.3011705.

[12] S. S. Gill et al., “ThermoSim: Deep learning based framework for modeling and simulation of thermal-aware resource management for cloud computing environments,” J. Syst. Softw., vol. 166, p. 110596, 2020.

[13] N. Liu et al., “A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning,” in 2017 IEEE 37th international conference on distributed computing systems (ICDCS), IEEE, 2017, pp. 372–382.

[14] H. L. Leka, Z. Fengli, A. T. Kenea, N. W. Hundera, T. G. Tohye, and A. T. Tegene, “PSO-Based Ensemble Meta-Learning Approach for Cloud Virtual Machine Resource Usage Prediction,” Symmetry, vol. 15, no. 3, p. 613, Feb. 2023, doi: 10.3390/sym15030613.

[15] T. Sylla, L. Mendiboure, M. A. Chalouf, and F. Krief, “Blockchain-Based Context-Aware Authorization Management as a Service in IoT,” Sensors, vol. 21, no. 22, p. 7656, Nov. 2021, doi: 10.3390/s21227656.

[16] S. Malik, M. Tahir, M. Sardaraz, and A. Alourani, “A Resource Utilization Prediction Model for Cloud Data Centers Using Evolutionary Algorithms and Machine Learning Techniques,” Appl. Sci., vol. 12, no. 4, p. 2160, Feb. 2022, doi: 10.3390/app12042160.

[17] L. Hang and D.-H. Kim, “Design and Implementation of an Integrated IoT Blockchain Platform for Sensing Data Integrity,” Sensors, vol. 19, no. 10, p. 2228, May 2019, doi: 10.3390/s19102228.

[18] A. R. Khan, “Dynamic Load Balancing in Cloud Computing: Optimized RL-Based Clustering with Multi-Objective Optimized Task Scheduling,” Processes, vol. 12, no. 3, p. 519, Mar. 2024, doi: 10.3390/pr12030519.

[19] M. I. Alghamdi, “Optimization of Load Balancing and Task Scheduling in Cloud Computing Environments Using Artificial Neural Networks-Based Binary Particle Swarm Optimization (BPSO),” Sustainability, vol. 14, no. 19, p. 11982, Sep. 2022, doi: 10.3390/su141911982.

[20] H. Huang et al., “A new fruit fly optimization algorithm enhanced support vector machine for diagnosis of breast cancer based on high-level features,” BMC Bioinformatics, vol. 20, no. 8, p. 290, Jun. 2019, doi: 10.1186/s12859-019-2771-z.

[21] Y. Qian et al., “Towards decentralized IoT security enhancement: A blockchain approach,” Comput. Electr. Eng., vol. 72, pp. 266–273, Nov. 2018, doi: 10.1016/j.compeleceng.2018.08.021.

[22] U. Khalid, M. Asim, T. Baker, P. C. K. Hung, M. A. Tariq, and L. Rafferty, “A decentralized lightweight blockchain-based authentication mechanism for IoT systems,” Clust. Comput., vol. 23, no. 3, pp. 2067–2087, Sep. 2020, doi: 10.1007/s10586-020-03058-6.