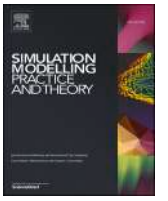
Simulation Modelling Practice and Theory 134 (2024) 102946

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat



PredictOptiCloud: A hybrid framework for predictive optimization in hybrid workload cloud task scheduling

Sugan J [a,*], Isaac Sajan R [b]

[a] Research Scholar, Department of Electronics & Communication Engineering, Ponjesly College of Engineering, Nagercoil, Tamilnadu, India
[b] Professor, Department of Electronics & Communication Engineering, Ponjesly College of Engineering, Nagercoil, Tamilnadu, India

ARTICLE INFO

Keywords:
Task scheduling
Hybrid workload
Cloud computing
e-commerce
Bi-LSTM
Spider Wolf Optimization

ABSTRACT

In the realm of e-commerce, the growing complexity of dynamic workloads and resource management poses a substantial challenge for platforms aiming to optimize user experiences and operational efficiency. To address this issue, the PredictOptiCloud framework is introduced, offering a solution that combines sophisticated methodologies with comprehensive performance analysis. The framework encompasses a domain-specific approach that extracts and processes historical workload data, utilizing Domain-specific Hierarchical Attention Bi LSTM networks. This enables PredictOptiCloud to effectively predict and manage both stable and dynamic workloads. Furthermore, it employs the Spider Wolf Optimization (SWO) for load balancing and offloading decisions, optimizing resource allocation and enhancing user experiences. The performance analysis of PredictOptiCloud involves a multifaceted evaluation, with key metrics including response time, throughput, resource utilization rate, cost-efficiency, conversion rate, rate of successful task offloading, precision, accuracy, task volume, and churn rate. By meticulously assessing these metrics, PredictOptiCloud demonstrates its prowess in not only predicting and managing workloads but also in optimizing user satisfaction, operational efficiency, and cost-effectiveness, ultimately positioning itself as an invaluable asset for e-commerce platforms striving for excellence in an ever-evolving landscape.

1. Introduction

Cloud computing has brought about a significant transformation in the utilization of computing resources, providing scalability, cost-efficiency, and flexibility for managing both stable and dynamic workloads [1]. Stable workloads, such as online stores, web hosting, and email servers, require consistent resource allocation to meet predictable demand [2,3]. On the other hand, dynamic workloads, including social networks, video streaming, and online games, exhibit unpredictable and fluctuating resource requirements [4,5]. However, the emerging challenge lies in handling hybrid workloads, which combine stable and dynamic characteristics [6]. Hybrid workloads require a more sophisticated approach to task scheduling, as they involve managing resources efficiently while adapting to changing demands. Efficient task scheduling algorithms are crucial for achieving load balancing, minimizing response time, and reducing energy consumption in hybrid environments [7–10].

One crucial problem in task scheduling for hybrid workloads is the decision of whether to execute a task locally or offload it to the cloud [11]. This decision depends on several factors, including the workload characteristics, resource availability, and network

* Corresponding author.
E-mail address: suganj067@gmail.com (S. J).

https://doi.org/10.1016/j.simpat.2024.102946
Received 20 December 2023; Received in revised form 21 March 2024; Accepted 11 April 2024
Available online 12 April 2024

latency. Offloading tasks to the cloud can help alleviate resource constraints, enhance scalability, and enable access to powerful computing resources [12–17]. However, the decision to offload tasks should be carefully made to ensure optimal performance and resource utilization [17–21]. To address the challenges of task offloading in hybrid environments, machine learning techniques can be leveraged [22]. By utilizing historical data, predictive models can be developed to estimate processing time, energy consumption, and network latency associated with different offloading decisions. Machine learning algorithms can analyze patterns and relationships in the data to make informed predictions and guide the task scheduling process.

The primary objective of this study is to create highly efficient task scheduling algorithms specifically designed for managing hybrid workloads in cloud computing environments. By leveraging machine learning, we seek to create predictive models that aid in the decision-making process of task offloading. These models will consider various factors, such as workload characteristics, resource availability, and network latency, to determine the optimal execution location for each task.

2. Related works

In the related work section, the emphasis is placed on exploring current approaches in cloud computing that aim to achieve efficient task scheduling. These approaches tackle various challenges associated with workload management, cost optimization, load balancing, energy consumption, and resource utilization.

2.1. Optimization-based approaches

Abualigah and Alkhrabsheh [23] introduce MVO-GA to optimize task scheduling, a novel approach that combines the multi-verse optimizer with a genetic algorithm. It focuses on reducing costs and improving service availability by considering cloud resource workload. The method demonstrates effectiveness in optimizing large-scale task transfer times but relies on the quality and representativeness of workload information gathered from cloud resources. Grzegorowskiet al. [24] have a research objective of reducing the total cost of ownership (TCO) for analytical data processing by dynamically configuring and managing resilient clusters on cloud resources. Their method optimizes cluster size and job execution schedules based on spot instance price history and ARIMA models, resulting in significant cost savings. However, accurate spot instance price prediction in dynamic and unpredictable cloud spot markets poses a challenge for the proposed method. Hu and Xiao [28] present a task offloading algorithm based on dynamic multi-objective evolution. The algorithm focuses on minimizing energy consumption and reducing task waiting time. The algorithm constructs a task scheduling model using dynamic multi-objective evolution and considers energy consumption effectiveness and validity for task offloading priority. Mangalampalli et al. [32], introduced cat swarm optimization algorithm for task scheduling. Here, the vital considerations are priority calculation, time of migration, and power cost. Task priorities are computed at each level and based on this, the scheduling is carried out. Based on this, the schedules are mapped into the corresponding VMs to execute each task. Nabi and Ahmed [33], introduced load balancing based on adaptive PSO considering deadline and resource aware factors. This method mainly considers the independent and intensive tasks for load balancing. Here, the main parameters considered are cost, makespan and time. Pradhan and Bisoy, [34] introduced load balancing based PSO to schedule tasks. The main consideration here is to reduce makespan and also to use all the resources available. Here, the tasks are scheduled based on the arrival time of each task and based on their loads at each state. Further, communication with the datacenter regarding the task and resource allocation is used in enhancing the load balancing process.

2.2. Load balancing and resource utilization

Wang et al. [25] address task scheduling for hybrid workloads in heterogeneous hybrid clouds. Their proposed Task Scheduling method concerning Security (TSS) considers cost-performance ratios of public resources and employs efficient scheduling algorithms. The objective is to enhance the completion of tasks within specified deadlines and security requirements, while simultaneously minimizing the costs associated with resource utilization. The method shows superior performance when task deadlines are not overly restrictive, but its effectiveness may be limited in scenarios with extremely tight task deadlines. Ebadifard and Babamir [26] focus on dynamic load balancing in cloud computing and propose an Autonomous Load Balancing method. The goal is to achieve a balanced distribution of the workload across virtual machines (VMs) while minimizing the overhead associated with communication. The proposed algorithm demonstrates improved load balancing and reduced communication overheads, particularly in scenarios with increased request variations and VM heterogeneity. However, the complexity of the load balancing technique may introduce additional computational overhead, potentially impacting system performance. Medara et al. [27] incorporate the Energy-Aware Scheduling with Virtual Machine Consolidation (EASVMC) algorithm, consisting of task scheduling and VM consolidation phases, to optimize energy consumption and resource utilization in cloud environments. The algorithm maps tasks to VMs based on execution length and minimum energy requirements and employs VM consolidation to improve resource utilization further. While the proposed method shows promise, the discrete water wave optimization approach used for VM consolidation may introduce computational complexity and overhead. Choudhary and Rajak [35], presented modified min-min heuristic for the purpose of load balancing. The main considerations here are the time taken to complete each task and also appropriate distribution of loads. This method is evaluated using different workflows such as montage workflows and also through the generated workflows. Ullah and Nawi [36], introduced a hybrid algorithm by combining bat and artificial bee colony algorithm. In this algorithm, the employed bees and onlooker bees are employed in equal numbers. The candidate searching process are carried out using the onlooker bees. Further, the fitness value is modified to avoid the overlapping that cause adverse effect on the accuracy due to reduced distribution of tasks.

2.3. Machine learning-based approaches

Tond et al. [29] tackle the dynamic online task scheduling problem and introduce a scheduling algorithm named DDQN-TS. Their approach utilizes the adaptive learning capability of a double deep Q-network (DDQN) to explore and discover effective strategies for task scheduling. Through extensive experiments, the results demonstrate that DDQN-TS achieves a high task completion rate and significantly reduces the average task response time compared to traditional algorithms. Liu et al. [30] propose the Heter PS system, which performs task scheduling for training deep neural network models using heterogeneous computing resources. The system employs a scheduling method based on reinforcement learning to allocate workload to suitable resources, with the objective of minimizing costs and ensuring compliance with throughput constraints. The scheduling decisions are based on a learned model and an RL-based method. Zhang et al. [31] introduce a multi-agent manufacturing system that utilizes deep reinforcement learning (DRL) to address the challenges of dynamic job shop scheduling in a changeable workshop environment. The system represents manufacturing equipment as equipment agents and employs an AI scheduler based on a multi-layer perceptron for intelligent task allocation. The Proximal Policy Optimization (PPO) algorithm is used for training and updating the AI scheduler, improving its decision-making performance. Siddesha et al. [37], introduced deep reinforcement learning for workload prediction and task scheduling. For this, a suitable policy is developed to provide maximum reward to perform actions in scheduling the tasks. However, there occurs problems to predict the optimal solution for the problem. Zhou [38], introduced a hybrid ML algorithm to allocate the resources and to efficiently schedule the tasks. For this, a scheduler is developed using the enhanced PSO. The main consideration is makespan time and also to increase the throughput. The resources are allocated efficiently through Graph attention network.

While the existing optimization-based, load balancing, and machine learning-based approaches present promising solutions for task scheduling and resource allocation in cloud computing environments, they are not without their drawbacks. One common limitation across these approaches is their reliance on historical workload data and predefined models, which may not always accurately capture the dynamic nature of e-commerce platforms. Additionally, some methods, such as MVO-GA and dynamic cluster configuration, heavily depend on the quality and representativeness of the workload information gathered from cloud resources, potentially leading to suboptimal scheduling decisions in scenarios with limited or noisy data. Moreover, algorithms like DDQN-TS and deep reinforcement learning-based approaches struggle with the complexity of predicting optimal solutions for dynamic workload scenarios, where patterns are constantly evolving and unpredictable. Furthermore, the computational overhead and training resource requirements associated with machine learning-based methods, such as heterogeneous PS and hybrid ML algorithms, are substantial, making them less practical for real-time deployment and scalability in large-scale e-commerce environments. These drawbacks underscore the need for more robust and adaptable frameworks, like PredictOptiCloud, that can effectively handle the intricacies of workload prediction and resource optimization while addressing the limitations of existing approaches. The specific contributions of the PredictOptiCloud is highlighted as follows:

- PredictOptiCloud employs a sophisticated approach to workload prediction by integrating domain-specific embeddings and Bidirectional Long Short-Term Memory (Bi LSTM) networks with a hierarchical attention mechanism. This allows the framework to effectively capture both stable and dynamic workload patterns in the e-commerce environment, enabling proactive resource management and optimization.
- The utilization of the Spider Wolf Optimization (SWO) algorithm within PredictOptiCloud facilitates efficient load balancing and intelligent task offloading decisions. SWO combines spider-like local exploration and wolf-like global exploration behaviors to optimize task allocation across servers, ensuring equitable workload distribution and minimizing variations in server loads.
- The framework addresses the challenges posed by hybrid workloads comprising both stable and dynamic components by incorporating adaptive task distribution and load balancing mechanisms. By dynamically adjusting task allocation based on user experience scores, conversion rates, and real-time feedback, PredictOptiCloud ensures optimal resource utilization and user satisfaction under varying workload conditions.
- PredictOptiCloud implements an iterative optimization process, mimicking the collaborative hunting techniques of wolves and the web-building strategies of spiders. This iterative approach allows the framework to continuously refine task scheduling and offloading decisions, ensuring convergence towards optimal solutions while adapting to changing workload dynamics and operational constraints.

3. Proposed PredictOptiCloud framework

PredictOptiCloud is a powerful framework designed to tackle the intricate task of workload prediction and efficient task scheduling within the dynamic landscape of e-commerce platforms. Its applications span a wide array of critical functions within the e-commerce industry. This comprehensive framework enables e-commerce businesses to optimize resource allocation, manage server loads, enhance user experience, reduce energy consumption, and minimize operational costs. It does this through a meticulous step-by-step process. First, it collects historical workload data encompassing user interactions, transactions, and special events, and then extracts essential features from this data. The framework leverages domain-specific embeddings and employs Bidirectional Long Short-Term Memory (BiLSTM) networks with an attention mechanism to predict and manage both stable and dynamic workloads. It also utilizes an Enhanced Spider Wolf Optimization (SWO) approach to optimize load balancing and offloading decisions. The advantages of PredictOptiCloud include efficient resource management, improved user experience, cost optimization, proactive scalability, and adaptability through real-time feedback integration. This framework is a valuable asset for e-commerce platforms striving to provide top-notch services while optimizing their operations. The system architecture of PredictOptiCloud is depicted in Fig. 1.

3.1. System architecture of PredictOptiCloud

The PredictOptiCloud system operates within the context of an e-commerce platform infrastructure E comprising a pool of Nvirtual machines denoted as$VM = \{VM_1, VM_2, VM_3, ..., VM_N\}$, each capable of hosting user applications and handling incoming requests. Let,$U = \{U_1, U_2, U_3, ..., U_N\}$, be the users who sends $D_{req} = \{D_{req1}, D_{req2}, D_{req3}, ..., D_{reqN}\}$requests for different tasks. The workload prediction module extract feature vector (F)at different time interval t. The predicted workload (W)represents the estimated workload intensity or resource demands $(r_j)$for each VM at time t. This prediction process involves employing machine learning techniques such as BiLSTM with attention mechanisms. The resource allocation and task scheduling module receives the workload predictions $(W)$and available resource information (r_j) to make optimal resource allocation decisions. This module ensures efficient utilization of VMs by assigning tasks (t) to the appropriate VMs based on their predicted workload intensity and current resource availability. The allocation process is facilitated by optimization algorithms such as SWO which aims to minimize resource wastage and maximize system throughput.

The choice of Bi LSTM embedded with attention and SWO within the PredictOptiCloud framework for workload prediction and resource optimization in e-commerce platforms is driven by several key factors, each addressing specific needs and challenges inherent to this domain.

a. Bi LSTM with Attention for Workload Prediction:

E-commerce platforms generate vast amounts of sequential data, including user interactions, transactions, and server requests. Bi LSTM is well-suited for processing such sequential data due to its ability to capture long-term dependencies and contextual information from both past and future timestamps simultaneously. The attention mechanism enhances the Bi LSTM model by allowing it to focus on relevant parts of the input sequence. In e-commerce, where certain events (e.g., flash sales) significantly impact workload, the attention mechanism ensures that the model can dynamically adapt to such events, leading to more accurate workload predictions. Embeddings capture the semantic nuances of e-commerce terms, enhancing the model's understanding of domain-specific vocabulary. This is crucial for accurately interpreting user behavior and predicting workload patterns unique to e-commerce platforms.

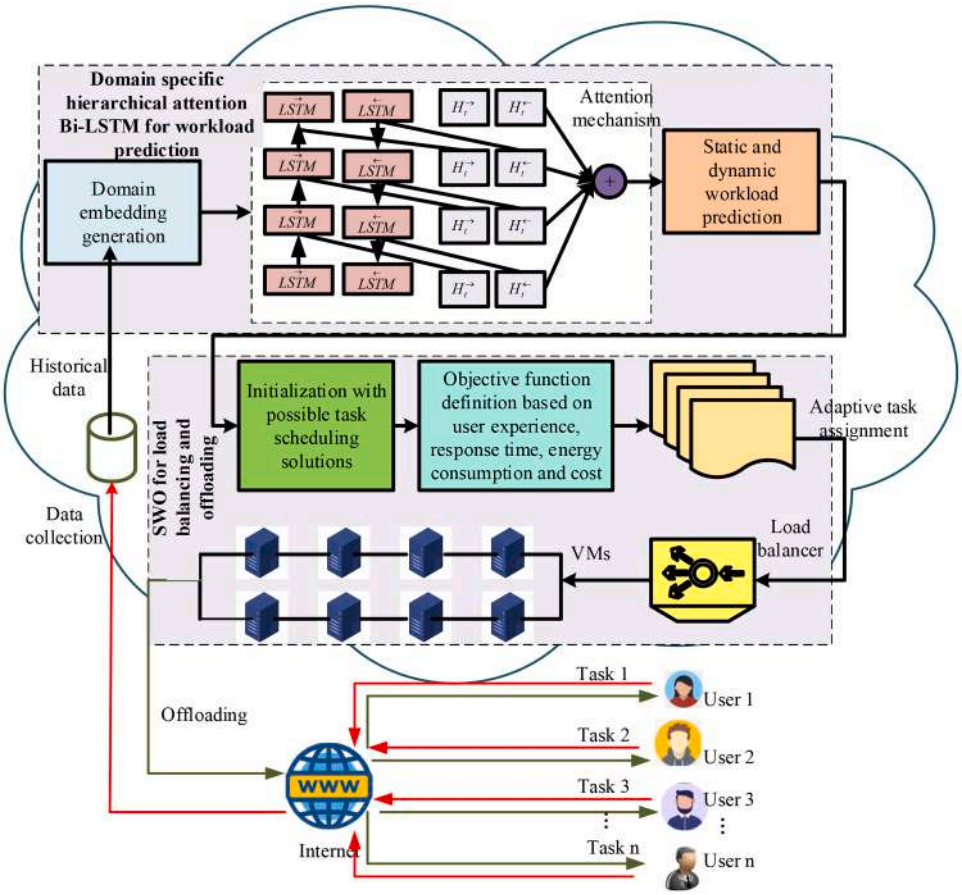


Fig. 1. System architecture of PredictOptiCloud.

b. Spider Wolf Optimization (SWO) for Load Balancing and Offloading:

E-commerce platforms experience varying levels of workload, including both stable (predictable) and dynamic (unpredictable) patterns. SWO offers a dynamic strategy for load balancing and task offloading, ensuring equitable distribution of tasks across servers while adapting to changing workload dynamics in real-time. SWO leverages both global exploration (wolf-like behavior) and local exploration (spider-like behavior). This approach allows for efficient exploration of the solution space, ensuring that the optimization algorithm can find the best possible solutions while also fine-tuning solutions for local improvements. SWO incorporates real-time feedback, allowing it to adapt to changing server or virtual machine performance. This adaptive optimization ensures optimal resource allocation, enhances user experience by reducing latency, and minimizes variations in server loads, crucial for maintaining operational efficiency in e-commerce platforms.

3.2. Problem statement

In our PredictOptiCloud for the e-commerce platform, at any given time$t$, the platform manages a stable workload$W_s(t)$, emanating from regular shopping patterns, intertwined with a dynamic workload$W_d(t)$, originating from sudden traffic spikes during promotions or new product launches. Together, they form the total operational demand$W(t) = W_s(t) + W_d(t)$. To optimize performance, certain tasks, denoted as$O(t)$, must be offloaded-like transferring some services to auxiliary servers during peak demands. The platform's key metrics revolve around the response time (r_t)(a direct influencer of user experience), the energy consumption of the backend servers(e_c), and the associated operational costs$C(t)$. To ensure a seamless shopping experience, the platform aims to minimize a weighted combination of these metrics, i.e$\min\{r_t, C(t), e_t\}$, given by$F(t) = \alpha \times r_t + \beta \times e_t + \gamma \times C(t)$. This objective is calibrated by adjusting weights, especially prioritizing response time to enhance user experience. This optimization is subjected to several constraints. Firstly, the representation of$W(t)$, should accurately depict the workload. Secondly, the predicted resource prices, denoted as, $P_r(t)$should closely align with actual costs. Another significant challenge is to ensure that the response time$R(t)$, does not exceed the task's deadline$D(t)$, i.e. $R(t) \leq D(t)$. For workload prediction, the task involves leveraging historical workload data $W_{hist}$to accurately forecast the future workload W_{future}over a specified time horizon T. This entails capturing complex temporal patterns, seasonal variations, and sudden spikes caused by events such as promotions or product launches. On the other hand, for offloading, given the predicted future workload$W_{future}$and the current capacities of servers $S_{cap}$the objective is to determine an optimal offloading strategy O for tasks while ensuring that response time constraints$R_t$are met and operational costs C are minimized. These formulations highlight the intricate challenges involved in managing workload dynamics and optimizing resource allocation in e-commerce environments. Addressing these challenges requires the development of advanced algorithms and techniques that can effectively handle the complexity of multi-

Table 1
Notations and its descriptions.

Notations	**Description**
$W_s(t)$	Stable workload
$W_d(t)$	Dynamic workload
$W(t)$	Total workload
$O(t)$	Offload
r_t	Response time
e_c	Energy consumption
$C(t)$	Operational costs
t_s	Timestamped
w_i	Individual work load instances
F	Feature vector
D_k	Chunk of data
B_j	Batched data
E_w	Embedding in a predefined vector space
$H_t^{\rightarrow}$	Hidden state at time $t$for the forward pass
$H_t^{\leftarrow}$	Hidden state at time $t$for the backward pass
e_t	Alignment score
α_t	Attention weights,
W_{future}	Predicted future workload
$W_{f.s}$	Future, stable workload
$W_{f.d}$	Future, dynamic workload
P	Set of solutions
r_j	Resource
(u_x)	User experience
$\alpha, \beta, \gamma, \delta$	Co-efficient
(c_r)	Conversion rate
t_{surge}	Predicted traffic surge for time t
l_t	Latency
$C_{offload}(t)$	Cost of offloading
$wb(p)$	Web behavior
$hb(p)$	Hunt behavior

objective optimization and dynamic resource constraints.

The problem outlined in the PredictOptiCloud framework presents several inherent complexities that contribute to its hardness. Firstly, the dynamic nature of e-commerce platforms introduces uncertainty and variability in workload patterns, making accurate prediction challenging. Capturing complex temporal dynamics, seasonal variations, and sudden spikes caused by promotions or product launches requires sophisticated forecasting models capable of handling diverse and unpredictable data patterns. Secondly, the optimization objective of minimizing a weighted combination of response time, energy consumption, and operational costs introduces a multi-objective optimization problem. Balancing these conflicting objectives while considering resource constraints and user experience requirements adds another layer of complexity to the problem. Moreover, the need to adjust weights dynamically to prioritize response time further complicates the optimization process. Additionally, ensuring that offloading decisions adhere to response time constraints and operational cost considerations while effectively managing server capacities poses significant challenges. The combinatorial nature of offloading tasks across auxiliary servers, considering various workload scenarios and resource constraints, further exacerbates the complexity of the optimization problem. Furthermore, the accuracy of workload prediction and resource price estimation directly impacts the effectiveness of the optimization process. Inaccurate predictions or estimations may lead to suboptimal offloading decisions, resulting in increased response times, higher energy consumption, and operational costs.

The notations and their corresponding descriptions used in the PredictOptiCloud is presented in Table 1.

While the PredictOptiCloud is proposed for workload prediction and optimization in e-commerce platforms, it is essential to address concerns about novelty and differentiate it from existing approaches that utilize machine learning for similar purposes. While the concept of employing machine learning for workload prediction and optimization is not entirely novel, PredictOptiCloud introduces several distinctive elements that set it apart from conventional methods. Firstly, PredictOptiCloud incorporates a Domain Specific Hierarchical Attention Bi LSTM model, which is personalized specifically for the e-commerce industry. This model utilizes hierarchical attention mechanisms to capture intricate temporal patterns and dynamic workload fluctuations unique to e-commerce platforms. By focusing on domain-specific features extraction and embedding generation, PredictOptiCloud ensures that the predictive model captures the nuances of e-commerce user behavior and server performance accurately. Secondly, the manuscript introduces the Spider Wolf Optimization (SWO) algorithm for load balancing and task offloading in e-commerce platforms. SWO leverages both spider-like local exploration and wolf-like global exploration behaviors to iteratively optimize task allocation across servers. This dynamic strategy incorporates real-time feedback and adapts to changing server performance, ensuring optimal resource utilization and user experience even in the face of unpredictable workload fluctuations. Additionally, PredictOptiCloud emphasizes the importance of separating stable and dynamic workloads and incorporates a classification mechanism to differentiate between them. By accurately predicting both routine and unpredictable workload patterns, PredictOptiCloud enables e-commerce platforms to make informed decisions on resource allocation and server scaling, thereby enhancing operational efficiency and user satisfaction. Furthermore, PredictOptiCloud integrates feedback mechanisms from user experience scores and conversion rates, allowing for adaptive task assignment and refinement of offloading decisions. This iterative approach ensures that the system continuously learns and adapts to evolving user behaviors and operational conditions, further enhancing its effectiveness in optimizing performance metrics.

3.3. Domain specific hierarchical attention Bi LSTM for E-commerce workload prediction

The Domain Specific Hierarchical Attention Bi LSTM for E-commerce Workload Prediction within the PredictOptiCloud framework is a sophisticated system designed to tackle the specific challenges of workload prediction in the e-commerce industry. It begins with comprehensive data collection, gathering historical data on user interactions, transactions, special events, and server performance. This data serves as the foundation for predicting future workloads. Domain-specific feature extraction enriches the data with temporal, behavioral, and technical metrics, while embedding generation captures the unique meaning of e-commerce terms. Bidirectional Long Short-Term Memory (Bi LSTM) processes the data, effectively handling both stable and dynamic workloads. An attention mechanism helps the model focus on relevant parts of the data. The result is a system that empowers e-commerce platforms to optimize resource allocation, server scaling, and marketing strategies, ensuring an efficient and responsive user experience in all scenarios, from routine operations to peak demand periods.

a. Data collection in PredictOptiCloud:

In the PredictOptiCloud for e-commerce platforms, our primary objective is efficient task scheduling that can adeptly handle both stable and dynamic workload. To achieve this, data collection becomes paramount. E-commerce platforms inherently have a vast reservoir of data, comprising user interactions, transactions, and server demands. To train the PredictOptiCloud historical workload data collection is carried with data points such as user interactions, transaction data, special events data, server requests and response time. In this the user interactions represents every user action, such as product views, searches, adding to cart, or making a purchase. The transaction data is the time stamped data $t_s$of every purchase or failed transaction.The server requests and response times corresponds to logs of every request to the server, its type, and the time taken to process. The special events data includes the data from particular sales events, new product launches, or marketing campaigns.To represent this, the historical workload data at time t is.

$$W_h(t) = \{w_1, w_2, \ldots w_n\} \tag{1}$$

In this, $W_h(t)$encapsulates the total number of server requests at time$t$and $w_i$represents individual workload instances. From this the domain-specific features extraction is performed using a wide feature set. This feature set comprise of the temporal features like the day of the week, time of the day, nearness to holidays or sale events, behavioral features such as product views, cart additions, wish list

additions, and purchase actions with technical metrics like server response time, error rates, and downtimes. Given a specific time$t$, the feature vector $F$is represented as,

$$F(t) = \{f_1, f_2, ..f_m\} \tag{2}$$

In this, f_i represents individual features.

Historical data is often stored in distributed databases or data warehouses. E-commerce platforms usually employ systems like Hadoop for large-scale data storage and retrieval. Here, the e-commerce interactions are vast, we employ the distributed storage systems. For this, the data is divided into chunks and stored across multiple nodes to ensure redundancy and availability. Let each chunk of data be represented asD_k. The total data is represented as the union of all chunks represented as,

$$D = \bigcup_{k=1}^{K} D_k \tag{3}$$

For the purpose of modeling, data retrieval is done in batches. Therefore, for a batch size of$b$, the batched data $B_j$is represented as,

$$B_j = \{d_1, d_2, ...d_b\} \tag{4}$$

Here, $d_i$represents individual data points in the batch.

b. Embedding Generation

In the PredictOptiCloud framework, embeddings play a pivotal role in capturing the semantic nuances of domain-specific terms.

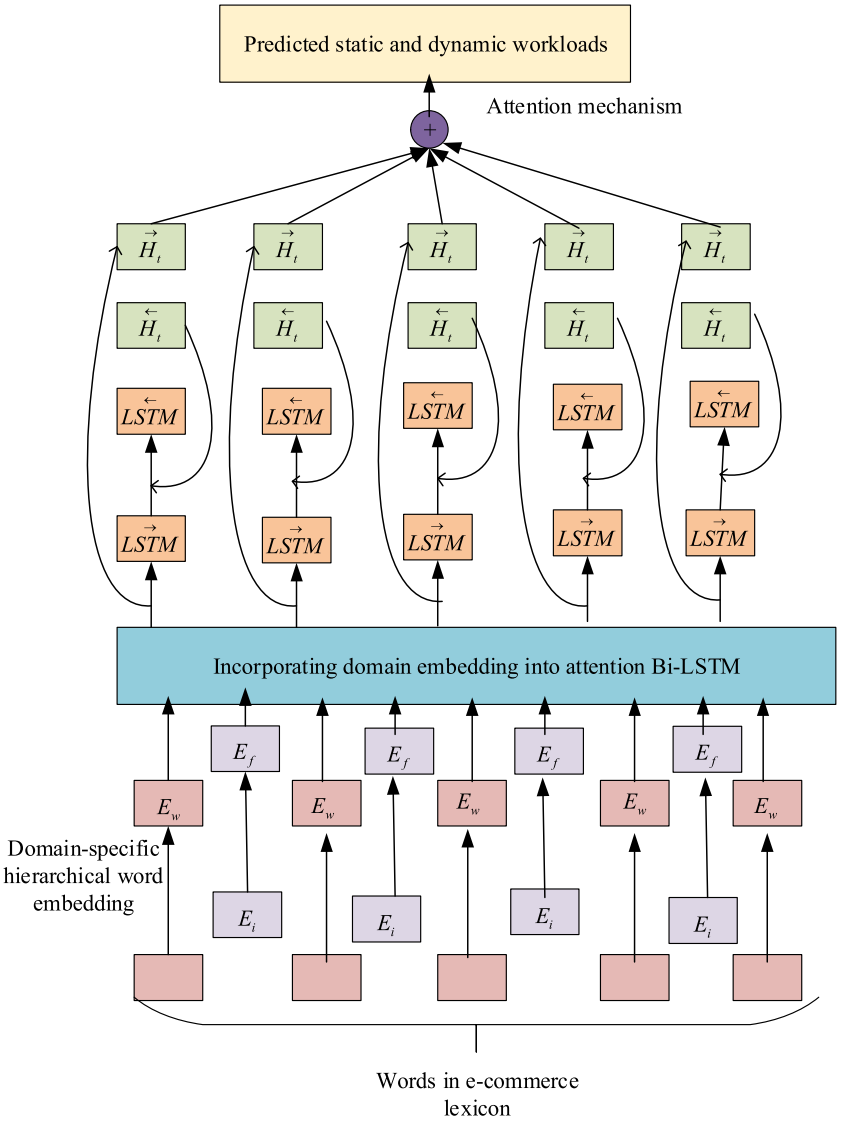


Fig. 2. Workload prediction process.

Given a term$w$ from the e-commerce lexicon, its corresponding embedding in a predefined vector space is represented asE_w.This embedding is derived by transforming the term through a domain-specific embedding matrix$M$,resulting in the equation$E_w = M \times w$. For instance, the term 'checkout' has its embedding denoted as$E_{checkout}$which is mathematically represented by the relationship$E_{checkout} = M \times w_{checkout}$. Such an embedding captures the essence of 'checkout' in relation to e-commerce operations, ensuring that it encapsulates its unique significance within the purchasing process.

c. Incorporating Domain Embeddings into Bi LSTM

The embedding transformation is done for every word in our e-commerce platform of PredictOptiCloud. This transformation captures the semantic nuances of e-commerce lexicon. Formally, for each word in our sequence, we have$E_i = E_f(w_i)$. Here, $E_f$is the embedding function that maps a word to its respective high-dimensional vector. Followed by this, the Bi LSTM sequence processing is performed for these embeddings. The sequence is fed into the Bi LSTM layers. Bi LSTM, unlike a traditional LSTM, processes data in both chronological and anti-chronological orders, encapsulating both past and future context for each timestamp. The past-to-future context called as the forward pass is denoted as,

$$H_t^{\rightarrow} = lstm\left(E_t, H_{t-1}^{\rightarrow}\right) \tag{5}$$

Here, $H_t^{\rightarrow}$is the hidden state at time $t$for the forward pass, while $E_t$is the embedding at time$t$. The future-to-past context called as the backward pass is represented as,

$$H_t^{\leftarrow} = lstm\left(E_t, H_{t+1}^{\leftarrow}\right) \tag{6}$$

Here, $H_t^{\leftarrow}$is the hidden state at time $t$for the backward pass. This is subjected to contextual fusion, by processing the sequence in both the directions, the Bi LSTM captures comprehensive context. For each timestampt_s, the fused context becomes,

$$H_t = \left[H_t^{\rightarrow}; H_t^{\leftarrow}\right] = \left[lstm\left(E_t, H_{t-1}^{\rightarrow}\right); lstm\left(E_t, H_{t+1}^{\leftarrow}\right)\right] \tag{7}$$

In this, ;denotes the concatenation. The combined hidden stateH_t, encapsulates context information from both before and after the timestampt_s. This context encompasses both stable and dynamic patterns. The workload prediction using domain specific hierarchical attention Bi-LSTM is depicted in Fig. 2.

d. Attention Mechanism in Bi LSTM:

The attention mechanism's primary goal is to allow the model to pay attention to specific parts of the input sequence when producing an output. In the context of e-commerce, this becomes vital when certain events, like flash sales, significantly influence the workload.

Stable Patterns: Since Bi LSTMs are equipped to recognize long-term dependencies, they can effectively recognize and remember patterns that consistently appear in the data. This characteristic is crucial for e-commerce platforms, where certain behaviors (like average browsing time before purchase) may remain relatively constant over longer periods.

Dynamic Patterns: On the other hand, short-term anomalies or spikes (like sudden increased activity due to flash sales) are also captured by the Bi LSTM, as it takes into account the immediate past and future context.

For each timestampt_s, the model computes an alignment score,e_t. This score measures the importance or relevance of an input at timestamp $t_s$concerning the entire input sequence. The alignment score for the input at time t is computed as:

$$e_t = \tanh\left(W \times \left[H_t^{\rightarrow}; H_t^{\leftarrow}\right]\right) = \tanh\left(W \times \left[lstm\left(E_t, H_{t-1}^{\rightarrow}\right); lstm\left(E_t, H_{t+1}^{\leftarrow}\right)\right]\right) \tag{8}$$

Here, W is a weight matrix that is learned during training.

The alignment scores alone do not provide a normalized measure of importance. To get this, we convert these scores into attention weights, α_t, which sum up to 1 over the sequence. This is done using the soft max function:

$$\alpha_t = \frac{\exp(e_i)}{\sum_{i=1}^{T} \exp(e_i)} \tag{9}$$

Here, T is the total number of timestamps in our sequence.exp()is the exponential function. These attention weights α_t give a probabilistic measure of the importance of each input timestamp $t_s$in the context of the entire sequence.

e. Workload Prediction with Attention:

Stable workload represents the predictable, routine demands on the platform. This could be users browsing, searching for products, reading reviews, etc. Understanding the stable workload assists in optimal resource allocation during non-peak times. Dynamic workload represents the unpredictable spikes often resulting from promotional campaigns, flash sales, or viral products. Predicting the dynamic workload allows e-commerce platforms to scale up resources proactively, ensuring seamless user experience during high-demand periods.

By separating the future workload into stable and dynamic components, PredictOptiCloud allows e-commerce platforms to make informed decisions on resource allocation, server scaling, and even marketing strategy optimizations. This differentiation ensures that platforms are neither over-resourced (costly) during regular times nor under-resourced (damaging to user experience) during spikes. The essence of this step is to generate a forecast of the e-commerce platform's workload using the attention mechanism.

Using the computed attention weightsα_t , which encapsulate the importance of each input, and the respective hidden statesh_t, we compute the predicted future workload W_{future}as,

$$W_{future} = \sum_{t=1}^{T} \alpha_t \times h_t \tag{10}$$

Here, T is the total number of timestamps in our sequence. This predicted workload is an aggregate, representing both routine operations and the sudden spikes.

To separate the stable workload from the dynamic, we introduce a classification mechanism. This mechanism determines, for each timestamp, the likelihood of the workload being stable or dynamic. For a specific timestampt_s,

$$P_s(t) = \sigma(W_s \times \alpha_t + b_s) \tag{11}$$

$$P_d(t) = \sigma(W_d \times \alpha_t + b_d) \tag{12}$$

Here, $\sigma$is the sigmoid activation function, ensuring output probabilities between 0 and 1.$W_s$and$W_d$are the weight matrices tailored for stable and dynamic classifications.$b_s$and$b_d$are the respective biases.

By multiplying the attention weights with these probabilities, we split the future workload predictions into stable and dynamic components:

$$W_{f,s} = \sum_{t=1}^{T} \alpha_t \times P_s(t) \times h_t = \sum_{t=1}^{T} \alpha_t \times \sigma(W_s \times \alpha_t + b_s) \times h_t \tag{13}$$

$$W_{f,d} = \sum_{t=1}^{T} \alpha_t \times P_d(t) \times h_t = \sum_{t=1}^{T} \alpha_t \times \sigma(W_d \times \alpha_t + b_d) \times h_t \tag{14}$$

Here, $W_{f,s}$represents the future, stable workload and $W_{f,d}$denotes the future, dynamic workload. The algorithm for workload prediction is presented in Algorithm 1.

3.4. Spider Wolf Optimization (SWO) for load balancing and off loading

Spider Wolf Optimization (SWO) is a dynamic strategy employed within the PredictOptiCloud framework for effective load balancing and task offloading in e-commerce platforms. It leverages both spider-like local exploration and wolf-like global exploration behaviors to iteratively optimize the allocation of tasks across servers. SWO ensures equitable workload distribution, aiming to minimize variations in server loads, while also making intelligent offloading decisions based on performance metrics like execution time and latency. Moreover, SWO incorporates real-time feedback, allowing it to adapt to changing server or virtual machine performance, ensuring optimal resource allocation, and enhancing user experience by reducing latency. In essence, SWO empowers e-commerce platforms to manage their resources efficiently, catering to both predictable and unpredictable workloads, a crucial aspect of ensuring seamless operations in the e-commerce industry.

Step 1: Initialization of the SWO Framework:

At first, SWO framework is initialized with the possible solutions. A solution in our context is a specific way the tasks are scheduled across the e-commerce platform's resources (like servers). This scheduling affects the user experience, server load, response time, energy consumption, and cost, which are the metrics we want to optimize. Each solution, denoted asp_i, can be seen as a vector where each entry represents the allocation of a specific task to a certain resource.

$$p_i = \{r_1, r_2, ..., r_n\} \tag{15}$$

Algorithm 1
Workload prediction using domain embedding attention Bi-LSTM.

Input: Historical data $W_h(t)$
Output: Predicted future stable and dynamic workloads $W_{f,s}$and $W_{f,d}$

Store $W_h(t) = \{w_1, w_2, ...w_n\}$ //Data collection
for each w_i
  Timestamp w_i
  Extract f_i
  Append $f_i \rightarrow F$
  Store $F \rightarrow D_k$
end
for each term w // Embedding generation
  Generate E_w
  Apply $E_f \rightarrow E_w$ // Incorporating domain embedding to Bi-LSTM
  Predict $H_t^{\rightarrow}$, $H_t^{\leftarrow}$
$H_t = [H_t^{\rightarrow}; H_t^{\leftarrow}]$
end
for each H_t // Attention mechanism in Bi-LSTM
  Compute e_t
  Convert $e_t \rightarrow \alpha_t$
  Using α_t
Output $W_{f,s}$,$W_{f,d}$ // Workload prediction
end

In this, $r_j$is the resource (like a specific server) assigned to handle a task$j$. The initial pool of solutions is the current best strategies or historically successful strategies to ensure that the optimization has a good starting point. This is denoted by,

$$P = \{p_1, p_2, ..., p_m\} \tag{16}$$

Here, P is the set of potential solutions with a size of$m$. For each solutionp_i , there's a corresponding performance score based on the objective function$F(p_i)$. This function measures how well the e-commerce platform performs under the task scheduling represented byp_i. It is a composite score that considers user experience(u_x), server response time(r_t), energy consumption(e_c), and costs(C).

$$F(p_i) = \alpha \times u_x(p_i) + \beta \times r_t(p_i) + \gamma \times e_c(p_i) + \delta \times C(p_i) \tag{17}$$

Here, $\alpha, \beta, \gamma, \delta$are the co-efficient of each metrics.

Step 2: Dynamic Task Distribution based on user experience and conversion rate:

User experience score(u_x): In an e-commerce platform, the user experience is paramount. A user who finds the platform responsive and efficient is more likely to complete a transaction. The (u_x) score can encompass metrics like page load time, server response time, and seamless transaction processing.

Conversion rate(c_r): This is a pivotal metric for e-commerce platforms. It indicates the percentage of visitors who take a desired action, such as making a purchase. A high conversion rate is an indicator of successful marketing and web design. The combination of (u_x) and(c_r) gives a holistic view of the user's journey, from browsing products to final purchase.

For every solution $p_i$in our population Pwe associate a (u_x)score and a conversion rate(c_r). These scores reflect how well tasks are scheduled in the e-commerce environment under that specific solution. Therefore,

$$u_x(p_i) = f_1(p_i) \tag{18}$$

$$c_r(p_i) = f_2(p_i) \tag{19}$$

In this, $f_1$andf_2 are the functions that evaluate$(u_x)$and(c_r) respectively.

To enhance task scheduling, we employ Spider Monkey Optimization's (SMO) grouping behavior. This step clusters solutions with similar performance concerning $(u_x)$and conversion rate. It operates under the premise that solutions with similar performance metrics might have similar optimal adjustments. Here, we define a distance metric$d$that captures the similarity between two solutions based on their $u_x$and c_r given by,

$$d(p_i, p_j) = w_1 \times \left|u_x(p_i) - u_x(p_j)\right| + w_2 \times \left|c_r(p_i) - c_r(p_j)\right| \tag{20}$$

Here, $w_1$and $w_2$are weights that determine the importance of the $u_x$and $c_r$in the distance metric$d$.Using this$d$, the SMO algorithm groups solutions into clusters or groups(G), where each group has solutions with proximate $u_x$and $c_r$values.

$$G = \{g_1, g_2, ..., g_k\} \tag{21}$$

Each $g_i$is represented as,

$$g_i = \{p_x, p_y, ...\} \tag{22}$$

Such that the distance between ant two solutions $p_x$and$p_y$within the group is below a certain threshold.

Step 3: Adaptive Task Assignment using Feedback:

Grey Wolf Optimization (GWO) mimics the leadership hierarchy and hunting behavior of grey wolves in nature. Within the pack, there are alpha, beta, and delta wolves. The alphas lead the hunt, followed by beta and delta wolves. Applying this analogy, solutions that yield the highest $u_x$scores and conversion rates $c_r$are termed asαsolutions. The next best are the βfollowed byδ. Let us denote the alpha solution asp_α, beta solution asp_β and delta solution asp_δ. These solutions guide the rest of the solutions during the optimization process. For a given solutionp_i, the update based on the leadership hierarchy is represented as:

$$p_{i,new} = p_\alpha - \alpha \times D_\alpha + p_\beta - \alpha \times D_\beta + p_\delta - \alpha \times D_\delta \tag{23}$$

In this, α is a coefficient that starts with a larger value and decreases over iterations. This ensures that the search space becomes narrower over time, ensuring convergence.D_α,D_β, and $D_\delta$represent the distance from the current solution to the alpha, beta, and delta solutions, respectively. This distance is computed as the absolute difference between the current solution's features and the features of the alpha, beta, and delta solutions. Feedback from users, be it in the form of reviews, ratings, or direct inputs, is a treasure trove of insights. For our optimization process, feedback can dynamically steer solutions towards better user satisfaction. Let us define a feedback function $F(p_i)$which maps the user feedback related to solution $p_i$to a score between 0 and 1.The task assignment is then further refined as:

$$p_{i,refined} = p_{i,new} + \lambda \times F(p_i) \tag{24}$$

In this, $\lambda$is a weighting factor defining the importance of feedback in task assignment. If $\lambda$is high, feedback plays a significant role in task adjustments.

Step 4: Load balancing

In a hybrid workload scenario where both stable and dynamic workloads coexist, it's essential to balance the load across servers to ensure equitable distribution of tasks. Load balancing is particularly crucial because the stable workload may have predictable

patterns, while the dynamic workload can fluctuate. The goal is to minimize the deviation in server loads while taking into account the differences in server capacities, connection speeds, and operational metrics.

For every server $s$in set$S$, the load or work must be approximately equal. The total workload for the platform, both static and dynamic, is given by:

$$W_t = \sum_{t=1}^{T} \left(W_{f,s} + W_{f,d}\right) \tag{25}$$

The optimal workload per server (assuming an even distribution) is:

$$L_{optimal}(s) = \frac{W_{total}}{|S|} \tag{26}$$

Here, $|S|$ is the number of servers or VMs in the set$S$. However, due to differences in server capacities, connection speeds, or other operational metrics, the actual load might deviate from the optimal. The objective of the load balancer is to minimize this deviation. Once, the load balancing is done, task scheduling and offloading is done for the hybrid workload.

Step 5: Load Offloading for Hybrid Workloads

Offloading decisions are made using SMO based on several criteria, considering response time, energy consumption, and cost while incorporating the leadership hierarchy of grey wolves. For each task (denoted by $i$with respect to user$u$), the offloading decision is influenced by several factors, and the goal is to determine whether the task should be executed locally or offloaded to a cloud resource. The decision is made by minimizing a cost function that includes response time, energy consumption, traffic surges and cost, with considerations for workload characteristics, resource availability, and network latency:

$$C(i,s) = w_1 \times r_t(i,s) + w_2 \times e_c(i,s) + w_3 \times C(i,s) + w_4 \times t_{surge}(i,s) \tag{27}$$

In this, $i$represents the task to be assigned/offloaded, represents the server or resource to which the task is assigned/offloaded. $r_t(i, s)$is the expected response time for executing task$i$on server/resource s.$e_c(i,s)$is the energy consumption for executing task i on server/resources. $C(i,s)$is the cost associated with executing task i on server/resources. $w_1, w_2, w_3, w_4$are weighting factors that determine the importance of each component in the cost function.$t_{surge}(i,s)$is the additional cost due to traffic surges, calculated based on the predicted surge and server/resource capacity. In the case of anticipated traffic, $t_{surge}$is determined as follows. Let t_{surge}represent the predicted traffic surge for time $t$and $l_t$denote the latency and therefore, $t_{surge} = \begin{cases} 1 & if\ t_{surge} \geq td_{surge}\ and\ l_t \leq l_{\max} \\ 0 & otherwise \end{cases}$ (28)

In this, $td_{surge}$is the threshold above which traffic is considered a surge and$l_t$is the maximum permissible latency. This decision considers the capability to handle offloaded tasks without compromising user experience.

The decision-making process for offloading tasks incorporates the hunting strategy of grey wolves from the GWO algorithm. Grey wolves use an alpha-beta-delta hierarchy where the alpha wolf leads, followed by the beta and delta wolves. This leadership hierarchy

Algorithm 2
Task offloading using SWO.

Input: Initial solution P, workloads $W_{f,s}$ and $W_{f,d}$
Output: Optimal solution for task offloading

1. **Initialize** the SWO
   a) **Initialize**$P = \{p_1, p_2, ..., p_m\}$ and$S = \{s_1, s_2, ..., s_m\}$
   b) **Initialize**$p_i = \{r_1, r_2, ..., r_n\}$
   c) **Initialize**$F(p_i)$
   d) **Initialize**$\alpha, \beta, \gamma, \delta$
   d) Set iteration counter i=1
2. Repeat until convergence

for each i, update p_i using $wb(p)$and $hb(p)$
  for each $p_i$in P
    Find $F(p_i)$
    Compute $(u_x)$and (c_r)
    Evaluate $(u_x)$and $(c_r)f_1$andf_2
    Define $d$with $w_1$and w_2
    Using $d$create (G)
    Compute next best α
    Determine βfollowed byδ
    Update $p_i$using α, β,δsuch that p_α, $p_\beta$and $p_\delta$emerges
    Using p_α, $p_\beta$and $p_\delta$find $p_{i,new}$
    Refine $p_{i,new} \rightarrow p_{i,refined}$ using $F(p_i)$ and λ
end
for every server $s$in set$S$
    Compute $L_{optimal}(s)$ using W_t
    Find $C(i,s)$using $r_t(i,s)$, $e_c(i,s)$, $t_{surge}(i,s)$
Output $O_{decision}(i:j)$
end
end
3. **Exalt** when $|F(t_{i+1}) - |F(t_i) \prec \in$

is adapted to task assignment for decision-making.The decision to offload a task is based on the calculated cost function $C(i,j)$and the GWO-inspired strategy. The server/resource $j$that minimizes the cost function is selected as the best option for task execution. The decision ensures that the task is assigned to a server/resource that can execute it efficiently and cost-effectively while considering workload characteristics, resource availability, network latency, and relevant factors.

$$O_{decision}(i:j) = \text{argmin}(C(i,j)) \tag{29}$$

By integrating the grey wolf optimization strategy and the cost function, this offloading criterion enables effective and informed decisions for hybrid workloads in cloud computing environments, ensuring optimal performance and resource utilization, even in the presence of anticipated traffic surges and latency considerations.

Step 6: Iterative Optimization and Model Convergence:

The primary driving force behind the SWO method is its behavior-based approach, modeling the collaborative hunting technique of wolves and the web-building strategies of spiders. For each iteration$i$, solutions are updated based on both spider and wolf behaviors:

$$p_{i+1} = p_i + \beta \times wb(p) + \gamma \times hb(p) \tag{30}$$

In this, p_{i+1} is the solution at the next iteration, p_i is the current solution.$\beta$and$\gamma$are weights which can be adaptively updated to balance between the two behaviors. Here $wb(p)$mimics a spider's web-building, emphasizing local exploration and $hb(p)$ represents a wolf's hunting strategy, focusing on global exploration. As iterations progress, the changes or improvements in the objective function diminish. The iterative process halts when:

$$|F(t_{i+1}) - |F(t_i) \prec \in \tag{31}$$

In this, $\in$is a pre-defined small value representing the minimum acceptable change in the objective function.Therefore, by leveraging both load balancing and offloading, the PredictOptiCloud framework ensures that e-commerce platforms can adeptly manage their resources, ensuring optimal user experience and operational efficiency. This becomes particularly crucial when dealing with both static (predictable) and dynamic (unpredictable) workloads, as often seen in e-commerce scenarios. The algorithm for task offloading using SWO is presented in Algorithm 2.

4. Experimental results and analysis

This section presents a detailed description about the experimental settings, parameters used, simulation results and analysis of task scheduling process. The experiments were conducted using a computer system running Windows 10. The experiments were carried out in a Python 3.6 environment, utilizing Tensor Flow 1.13. The hardware used for these experiments was a Lenovo Pro laptop with 8 gigabytes of RAM and a 2.4 gigahertz Intel Core i5 processor.

4.1. Experimental settings and its details

In the experimental setup, PredictOptiCloud plays a pivotal role in optimizing the resource allocation, load balancing, and offloading processes within an e-commerce platform. Each experiment extends over the course of a week, during which the platform encounters diverse e-commerce workloads. For instance, on a typical Monday, the platform experiences a steady flow of 5,000 users during the morning hours, while Saturdays bring a surge of 20,000 users due to flash sales, illustrating the dynamic nature of e-commerce traffic. To facilitate this optimization, PredictOptiCloud leverages a combination of real-world e-commerce data and synthetic datasets, ensuring the framework's adaptability to various scenarios. The workload is intentionally crafted to represent both stable and dynamic e-commerce traffic patterns. On an average day, it accommodates 100,000 users who are actively browsing products. Simultaneously, it caters to the demands of dynamic scenarios, such as flash sales, where peak traffic reaches 20,000 users.

To simulate these e-commerce dynamics, the experiment is conducted at a time granularity of 15 minutes, allowing for a detailed examination of resource allocation and task distribution throughout the day. In the pursuit of reproducibility, each experiment is repeated five times, accounting for the inherent variability in real-world e-commerce operations.The experimental setup also leverages the computational power of multi-core CPUs, distributing tasks across four CPU cores in parallel. This parallelization enhances the simulation's efficiency and ensures timely decision-making.For consistency in results, a specific random seed value, such as 42, is set to maintain reproducibility across experiments. During the experiments, the framework processes historical and real-time data streams, making predictions for the day. For example, PredictOptiCloud predicts a 15 % traffic surge on a Saturday due to anticipated flash sales. This prediction guides load balancing decisions, ensuring that tasks are distributed evenly among available servers or VMs, guaranteeing an approximately equal workload per resource. For instance, if 50 tasks need to be assigned on a Saturday with 10 servers, the algorithm ensures an optimal workload of 5 tasks per server.Moreover, offloading decisions are driven by a comprehensive cost function, which considers multiple factors, including response time, energy consumption, and capacity. The framework also incorporates real-time feedback, adapting to server performance variations. For example, if a particular server consistently underperforms throughout a day, PredictOptiCloud dynamically assigns fewer tasks to it.The cost-efficient offloading decision takes into account both performance metrics and financial implications, ensuring that the chosen approach optimizes system performance while managing costs effectively.

a. Performance metrics

The different metrics employed in analyzing the performance of the PredictOptiCloud is presented here.

Response time(r_t): It measures the duration between the initiation of a request and the receipt of the response. In e-commerce contexts, it signifies the time taken for a user's action (like clicking on a product) to yield a result (such as displaying product details).

$$r_t = t_{finish} - t_{start} \tag{32}$$

Here, $t_{start}$is the time a request is made and $t_{finish}$is the time a response is received.

Throughput: It indicates the number of tasks or requests the system can handle within a specific time frame. It's a measure of the system's capacity and efficiency.

$$T = \frac{N}{\Delta t} \tag{33}$$

Here, N is the number of tasks completed, and $\Delta t$is the time period considered.

Resource utilization rate: It quantifies the percentage of a resource's capacity (such as a server) that is actively used for processing tasks. It provides insights into whether the infrastructure is overburdened or underused.

$$U = \frac{T_{used}}{T_{total}} \times 100\% \tag{34}$$

In this, $T_{used}$is the time a resource is busy and T_{total} is the total available time.

Cost-Efficiency: It is a measure of the performance achieved per unit of cost. In a cloud environment, it balances the operational effectiveness against monetary expenditure, ensuring value for money.

$$CE = \frac{Performance}{\mathrm{Cost}} \tag{35}$$

Conversion rate(c_r)**:** It represents the proportion of site visitors who take a desired action, like making a purchase. A high CR signifies that the platform effectively nudges visitors towards sales, hinting at a well-designed user journey.

$$CR = \frac{N_{sales}}{N_{visits}} \times 100\% \tag{36}$$

In this, $N_{sales}$is the number of successful sales and $N_{visits}$is the number of site visits.

Rate of Successful Task Offloading: This metric evaluates the success rate of transferring tasks from one resource to another, typically from a local system to a cloud resource. A high RSTO indicates that offloading decisions align well with available resources and current demands.

$$RSTO = \frac{N_{offloaded,sucessful}}{N_{offloaded,total}} \times 100\% \tag{37}$$

Here, $N_{offloaded,successful}$is the number of successfully offloaded tasks and $N_{offloaded,total}$is the total number of offloaded tasks.

Precision: Precision in workload differentiation measures the ability of the framework to accurately identify and classify workloads as either static or dynamic. A precise framework minimizes misclassifications, ensuring that tasks are appropriately handled based on their characteristics. The precision equation can be defined as:

$$\mathrm{Pr}ecision = \frac{TP}{TP + FP} \tag{38}$$

Here, TP represents the number of correctly identified dynamic workloads, and FP represents the number of static workloads incorrectly classified as dynamic. A higher precision indicates fewer misclassifications.

Accuracy: It is a measure of how well the framework overall performs in workload differentiation. It considers both true positive and true negative classifications. The accuracy equation can be defined as:

$$Accuracy = \frac{TP + TN}{TP + FP + FP + FN} \tag{39}$$

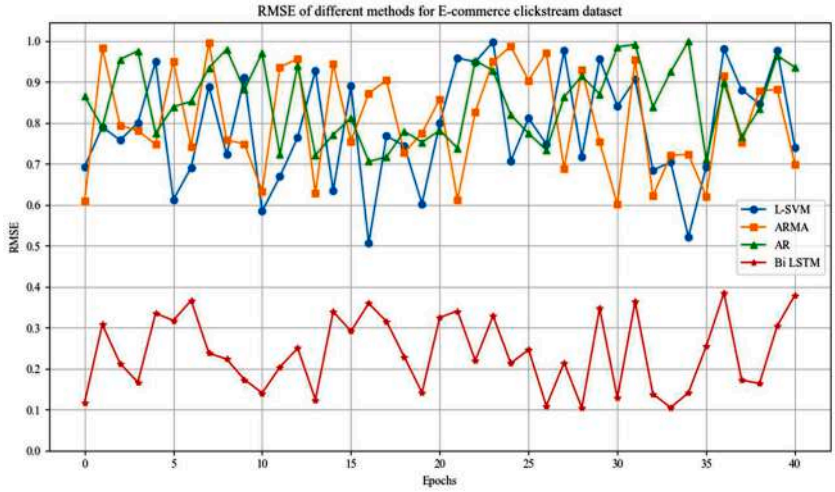
In this equation, TN represents the number of correctly identified static workloads, while FN represents the number of dynamic workloads incorrectly classified as static. High accuracy implies fewer overall misclassifications.

Task Volume: The volume of tasks within different workloads is an essential consideration. The framework should handle an appropriate number of tasks for each workload type. An equation to calculate the task volume can be represented as:

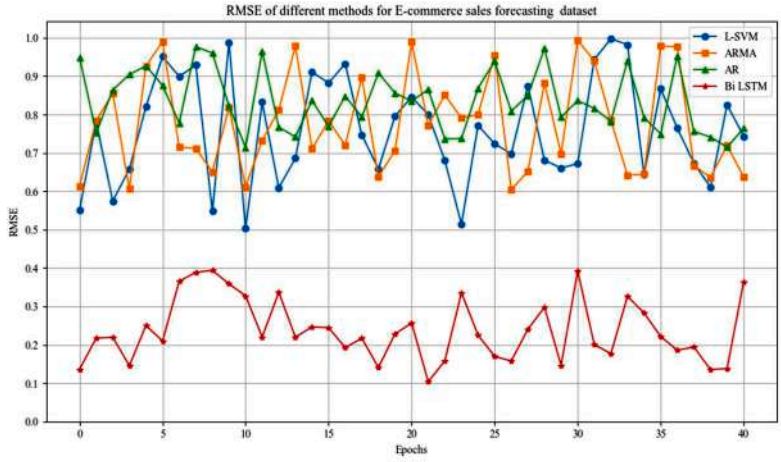
$$Task\ volume = \frac{tasks_d}{tasks_s} \tag{40}$$

Here, $tasks_d$denotes the tasks in the dynamic workload and $tasks_s$ denotes the tasks in the static workload. Balancing task volume ensures that resources are allocated optimally for varying workloads, preventing overloading or underutilization.

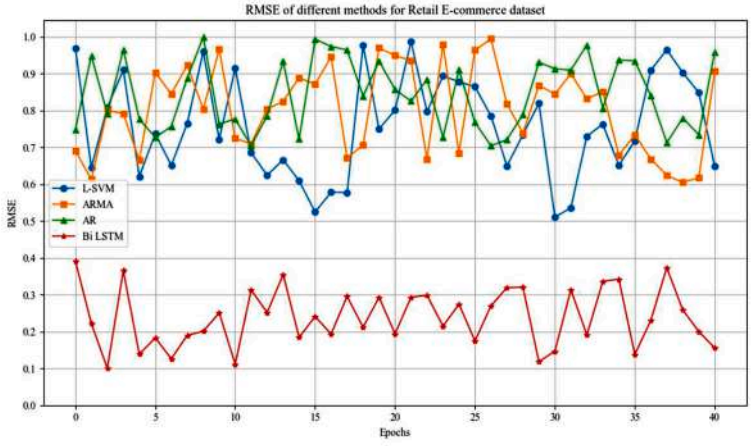
Regular Volume Allocation: The regular volume represents the typical workload where resources are allocated to tasks as they arrive. The resource allocation efficiency during regular volume is measured as:



(a)



(b)



(c)

Fig. 3. RMSE comparison for different datasets.

$$Eff_{regular} = \frac{Actual\ resource\ allocation}{Required\ resource\ allocation\ for\ regular\ volume} \tag{41}$$

Surge Volume Allocation: During surge periods, when there is a significant increase in workload, the system must allocate additional resources to handle the surge efficiently. The resource allocation efficiency during surge volume is measured as:

$$Eff_{surge} = \frac{Actual\ resource\ allocation}{Required\ resource\ allocation\ for\ surge\ volume} \tag{42}$$

Resource Allocation by PredictOptiCloud: PredictOptiCloud dynamically allocates resources based on real-time predictions and feedback. The efficiency of PredictOptiCloud's resource allocation is measured as:

$$Eff_{\mathrm{PredictOptiCloud}} = \frac{Actual\ resource\ allocation}{Resource\ allocation\ by\ \mathrm{PredictOptiCloud}} \tag{43}$$

Churn Rate: The churn rate for each VM is calculated as the number of task reassignments divided by the total number of tasks assigned to that VM over a specific time period (e.g., per day). This is represented as:

$$Churn\ rate_{VM} = \frac{Number\ of\ task\ reassignments}{Total\ number\ of\ tasks\ assigned} \tag{44}$$

RMSE: It measures the square root of the average of squared differences between predicted and actual values. Mathematically, RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2}{n}} \tag{45}$$

MAPE: It calculates the average of the absolute percentage differences between predicted and actual values.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \widehat{y}_i}{y_i}\right| \times 100\% \tag{46}$$

R-squared: R squared measures the proportion of the variance in the dependent variable that is predictable from the independent variables.

$$R^2 = 1 - \frac{SS_{res}}{SS_{TOT}} \tag{47}$$

Here, $SS_{res}$is the residual sum of squares, representing the difference between the observed and predicted values. $SS_{tot}$is the total sum of squares, representing the difference between the observed values and the mean of the observed values. b. Datasets

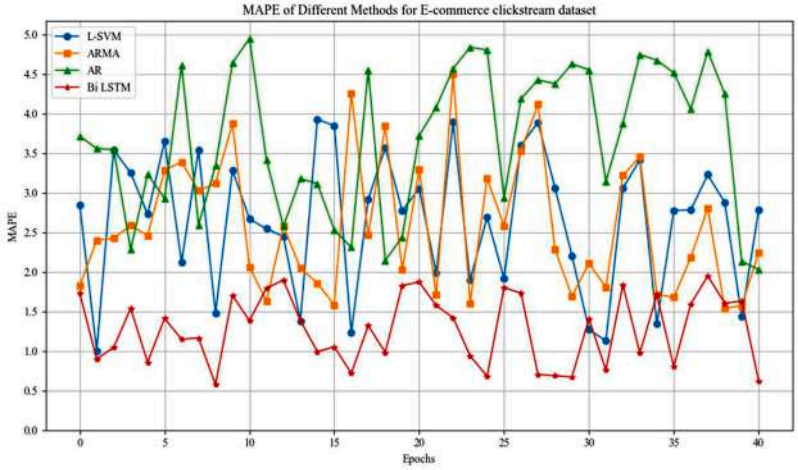
E-commerce clickstream dataset [39]: The dataset comprises clickstream data collected over a five-month period in 2008 from an online store specializing in clothing for pregnant women. Key attributes include the product category, indicating the type of item being viewed, and the location of the product photo on the webpage. Additionally, the dataset provides insights into user demographics through the country of origin of the IP address. Crucially, it includes product prices in US dollars, offering valuable information for pricing strategies and market analysis. This dataset is used to delve into user behavior during online shopping sessions, gaining insights into the preferences and browsing patterns of pregnant women.

E-commerce sales forecasting dataset [40]: The E-Commerce Sales Forecast dataset offers a comprehensive resource for predicting sales within an e-commerce setting, encompassing stores across three prominent US states: California, Texas, and Wisconsin. Featuring item-level details, department and product categorizations, and store-specific information, this dataset provides a rich foundation for analysis. Notably, it includes a range of explanatory variables such as product prices, promotional activities, day of the week, and special events, which are instrumental in enhancing forecasting accuracy. With its diverse array of attributes and potential applications, this dataset serves as a valuable asset for gaining insights into e-commerce sales dynamics and informing strategic decision-making processes.

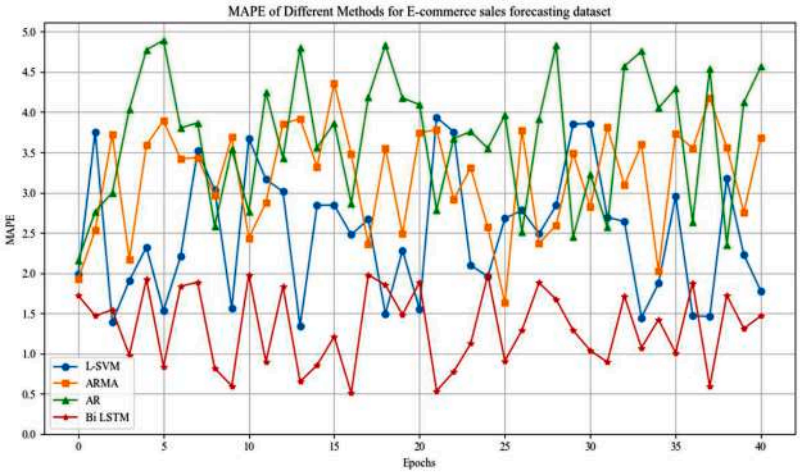
Retail E-commerce dataset [41]: The E-Commerce Sales Data dataset offers valuable insights into the performance of an online retail store, capturing transactional data over a specific period. Key attributes include detailed product information, customer details, transaction timestamps, and sales amounts. Researchers, analysts, and data enthusiasts can leverage this dataset for various purposes. They can conduct in-depth sales analysis to uncover trends, seasonality patterns, and peak sales periods, enabling strategic decision-making. Furthermore, the dataset facilitates customer segmentation efforts, allowing businesses to tailor their marketing strategies and services to different customer groups effectively.

4.2. Simulated results for real world datasets

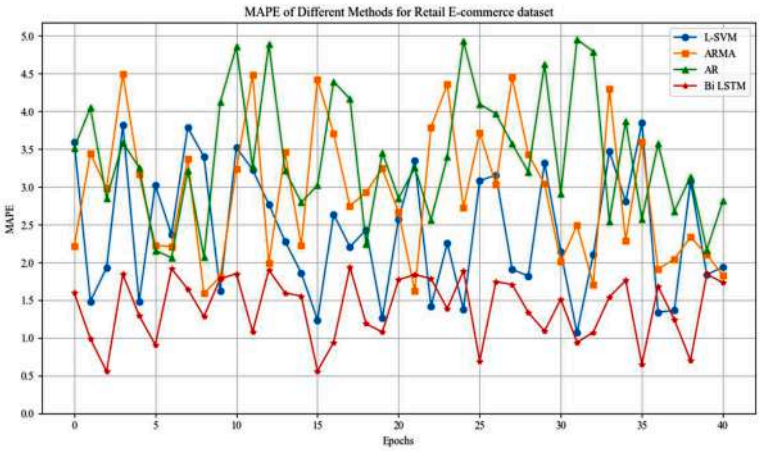
The Fig. 3 illustrates the root mean square error (RMSE) comparison across different datasets for various workload predictors, including LSVM, ARMA, and AR, with the proposed BiLSTM model. The BiLSTM model consistently outperforms the other methods, exhibiting RMSE values ranging from 0.1 to 0.4 across different epochs, compared to the RMSE values of 0.5 to 1.0 achieved by the alternative methods. Particularly noteworthy is the significant improvement in RMSE observed in the retail e-commerce dataset



(a)

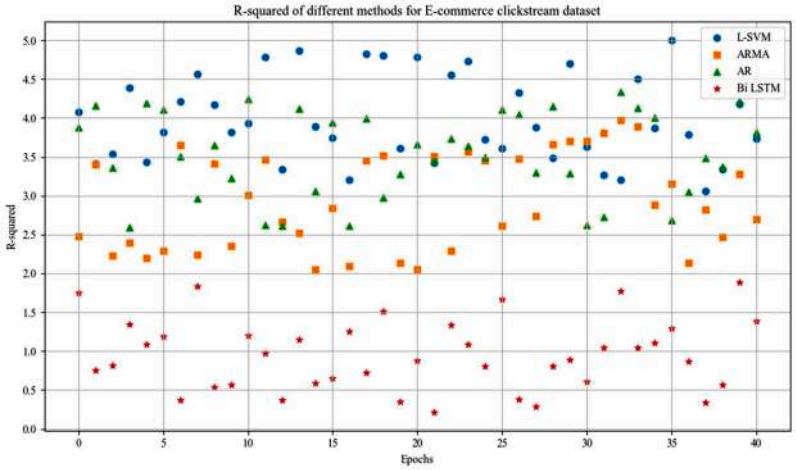


(b)

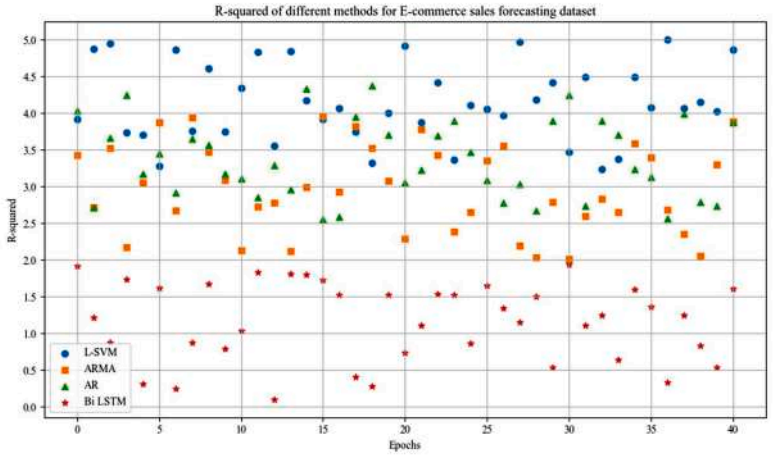


(c)

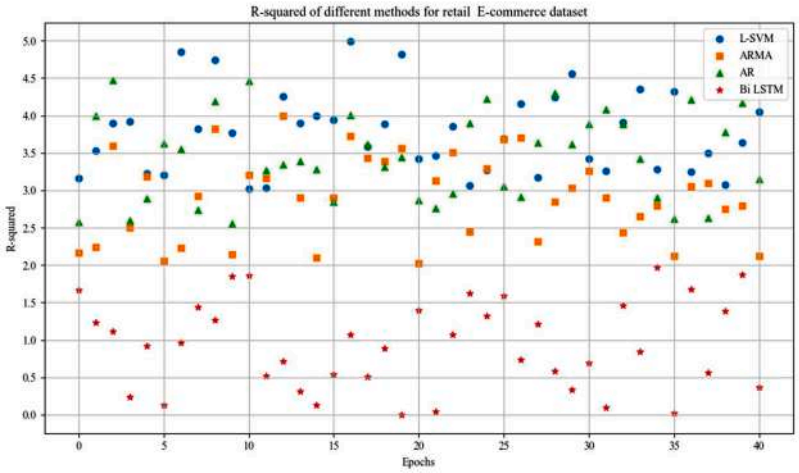
Fig. 4. MAPE comparison for different datasets.



(a)



(b)



(c)

Fig. 5. R square comparison for different datasets.

compared to the clickstreaming and sales forecasting datasets. This suggests that the BiLSTM model demonstrates superior predictive accuracy, especially in scenarios with intricate e-commerce dynamics, such as those encountered in retail environments.

The Fig. 4 depicting the MAPE (Mean Absolute Percentage Error) comparison among various workload predictors across different datasets highlights the performance of the proposed BiLSTM model in comparison to other methods such as LSVM, ARMA, and AR. It is evident that the proposed BiLSTM consistently outperforms the other methods across all datasets, achieving lower MAPE values ranging from 0.5 to 2 for different epochs up to 40. In contrast, the alternative methods exhibit higher MAPE values across the board. Particularly, the AR method reaches a MAPE value of 5.0, indicating a considerable discrepancy between predicted and actual values. This comparison underscores the superior accuracy and predictive capability of the proposed BiLSTM model in forecasting workload patterns in various e-commerce scenarios, making it a promising approach for workload prediction tasks.

Fig. 5 depicts the comparison of R-squared values across different datasets, showcasing the performance of various workload predictors, including LSVM, ARMA, and AR, alongside the proposed BiLSTM model. The R-squared values for the proposed BiLSTM model range consistently higher across different datasets, signifying its superior ability to explain the variance in workload compared to the other methods. This implies that the BiLSTM model captures a larger proportion of the variability in workload data, indicating a better fit of the model to the observed patterns.

The Fig. 6 illustrates the response time of the PredictOptiCloud system across different datasets, including e-commerce clickstream, e-commerce sales forecasting, and retail e-commerce datasets, over a specific time range from 01:00 to 10:00, measured in milliseconds (ms). It is evident that the PredictOptiCloud system achieves a notably lower response time for the e-commerce clickstream dataset, ranging from less than 75 ms to 225 ms over the duration of 10 hours. In contrast, the other two datasets exhibit higher response times, exceeding 125 ms and gradually increasing throughout the observed time period. This comparison underscores the efficiency of the PredictOptiCloud system in handling e-commerce clickstream data, resulting in faster response times compared to the other datasets, which may involve more complex processing or higher data volumes.

The Fig. 7 presents a comparison of throughput for the PredictOptiCloud system across different datasets. Initially, at 01:00, the throughput value for the e-commerce sales forecasting dataset is notably higher than that of the other datasets, reaching a task completion of 225. However, as time progresses, the number of tasks completed by the other datasets gradually increases. By later time points, such as at 10:00, both the e-commerce clickstream and retail e-commerce datasets exhibit similar throughput values, with approximately 345 tasks completed. This trend suggests that while the e-commerce sales forecasting dataset may initially show a higher throughput, the e-commerce clickstream and retail e-commerce datasets eventually catch up and even surpass it in terms of task completion, indicating efficient task processing and resource utilization by the PredictOptiCloud system across various datasets as time progresses.

The Fig. 8 provides a detailed comparison of resource utilization rates across different datasets for the PredictOptiCloud system. Particularly notable is the performance of PredictOptiCloud with the e-commerce clickstream dataset, where it demonstrates exceptional efficiency in managing VM resources, achieving utilization rates ranging from 58 % to 85 % across 10 VMs. However, for the other datasets, the utilization rates are comparatively lower, with values of 45 %, 36 %, and 15 % observed for VM1. This disparity underscores PredictOptiCloud's ability to intelligently allocate tasks to VMs, mitigating both resource underutilization and overload. By strategically routing tasks based on factors like response time, energy consumption, and capacity, PredictOptiCloud ensures optimal resource utilization. Furthermore, the system's adaptive nature allows it to dynamically reassign tasks from underperforming VMs to

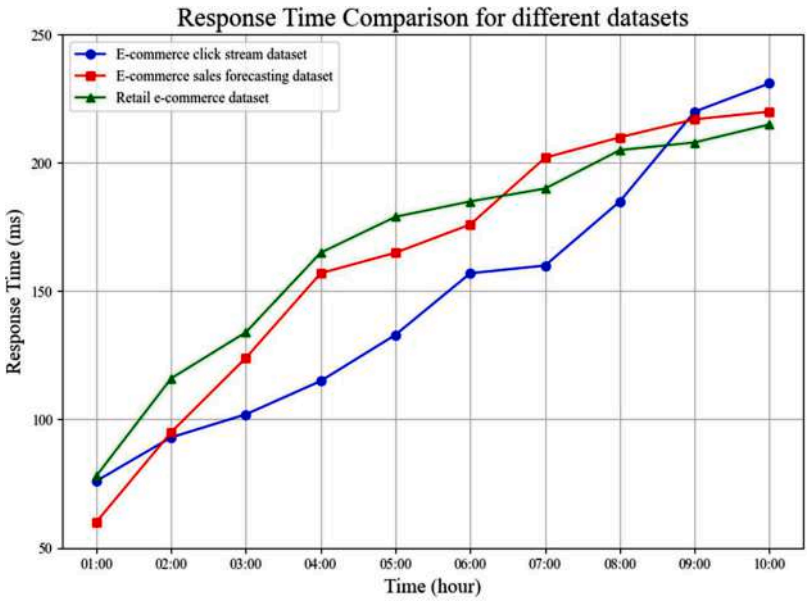


Fig. 6. Response time for different datasets.

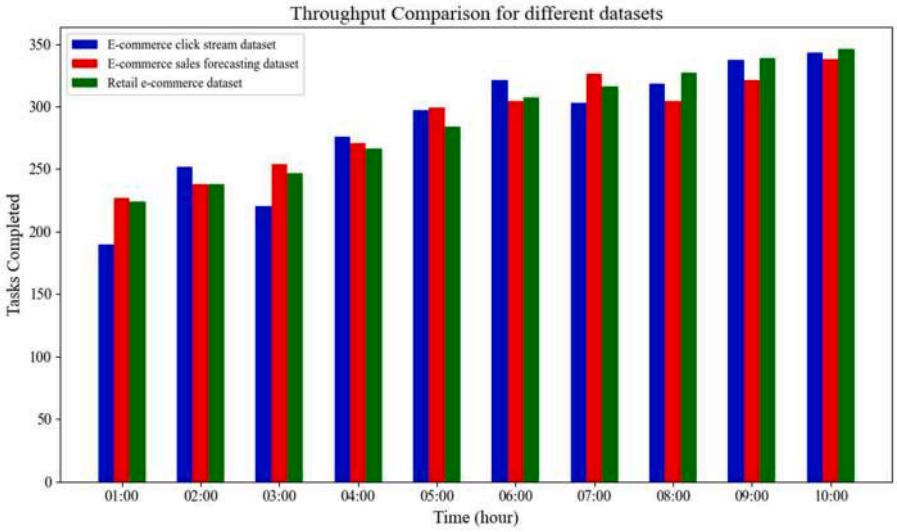


Fig. 7. Throughput comparison for different datasets.

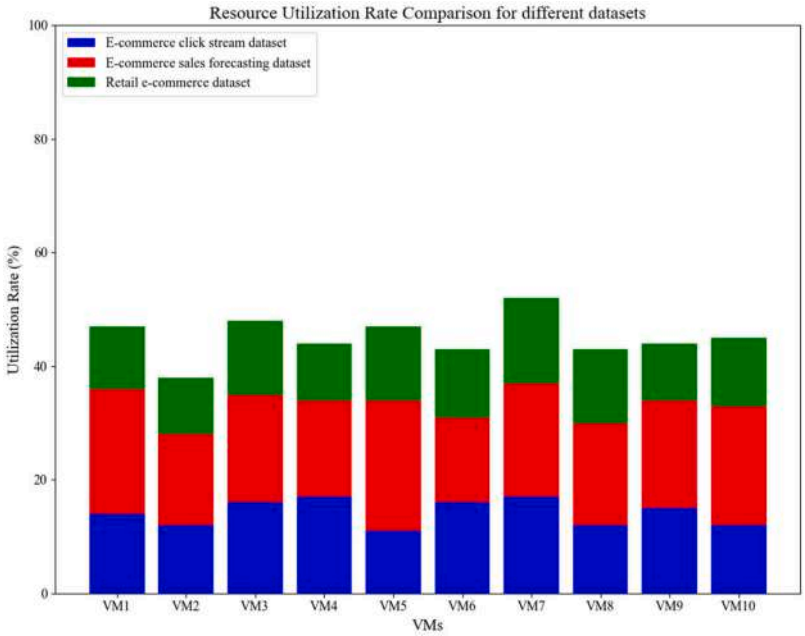


Fig. 8. Resource utilization rate.

those exhibiting better performance, further enhancing resource efficiency. Additionally, PredictOptiCloud's consideration of cost-efficient offloading decisions ensures not only performance optimization but also cost-effectiveness. By making informed decisions on task offloading, the system optimizes resource allocation while minimizing unnecessary expenses. The visualization of 10 VMs serves as a snapshot of PredictOptiCloud's prowess in resource utilization, highlighting its ability to streamline operations and maximize efficiency in diverse e-commerce environments.

The depicted Fig. 9 offers a comparison of conversion rates across various datasets over different time intervals, spanning from 01:00 to 24:00. Notably, PredictOptiCloud demonstrates a higher conversion rate for the retail e-commerce dataset, fluctuating between 7 % and 9 %. Contrastingly, the performance is comparatively lower for the e-commerce clickstream dataset, where the conversion rate ranges from 5 % to 7 %. Similarly, the e-commerce sales forecasting dataset achieves a conversion rate of 6 % to 7 %. This disparity underscores PredictOptiCloud's differential impact on conversion rates across diverse e-commerce datasets. The higher conversion rates observed for the retail e-commerce dataset suggest that PredictOptiCloud effectively optimizes resource allocation and task distribution, thereby enhancing the platform's ability to facilitate successful sales transactions. Conversely, the slightly lower

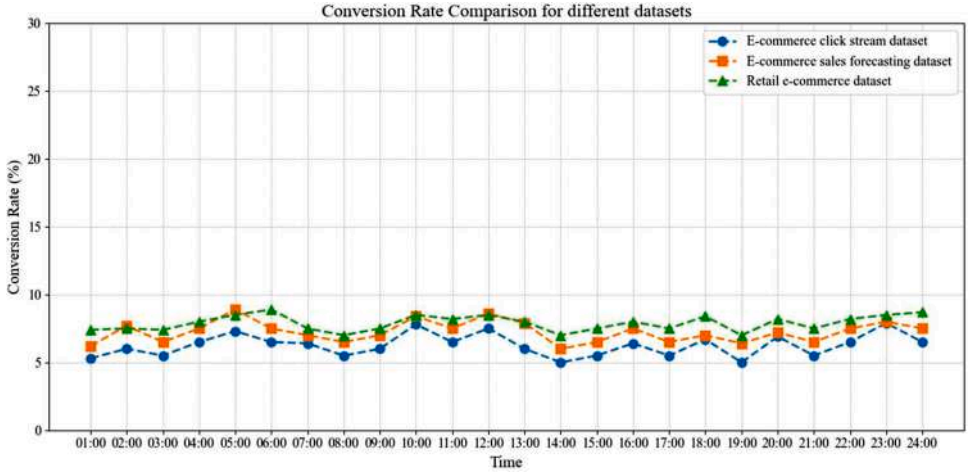


Fig. 9. Conversion rate comparison for different datasets.

conversion rates for the e-commerce clickstream and sales forecasting datasets indicate potential areas for further optimization or refinement in PredictOptiCloud's strategies for these specific contexts. By elucidating these variations in conversion rates, the figure provides valuable insights into PredictOptiCloud's performance and its implications for different types of e-commerce platforms.

4.3. Case analysis

In this section, we conduct a comprehensive case analysis to evaluate the performance of PredictOptiCloud using synthetic e-commerce data specifically generated for testing purposes. The details of the experimental setup and the synthetic data generation process are provided in Section 4.1. Here, we present the results obtained from our experiments.

The response time comparison Fig. 10 provides a detailed analysis of how different methods, such as EASVMC [27], DDQN-TS [29], DRL [31] and the PredictOptiCloud, perform in terms of response time over a specific time range from 01:00 to 10:00, measured in milliseconds (ms). PredictOptiCloud stands out as an exemplar of efficiency and adaptability in e-commerce workload management. At the outset (01:00), PredictOptiCloud impressively achieves an initial response time of only 50ms. This remarkable start is attributed to the methodology's proactive predictions based on historical and real-time data, allowing for optimal task distribution and resource utilization. As time progresses towards 10:00, response times gradually increase, reaching 140ms. But, other method such as EASMVC takes about 225ms to respond at 10:00. This progression aligns with PredictOptiCloud's dynamic workload management, ensuring balanced workloads across servers (Eq. 31) and adaptive task assignment based on performance metrics like response time and energy consumption (Eq. 24). PredictOptiCloud incorporates real-time feedback (Eq. 28) and cost-efficient offloading (Eq. 33), maintaining low response times while effectively adapting to changing conditions.

The throughput comparison Fig. 11 provides a comprehensive evaluation of various methods as EASVMC [27], DDQN-TS [29], DRL

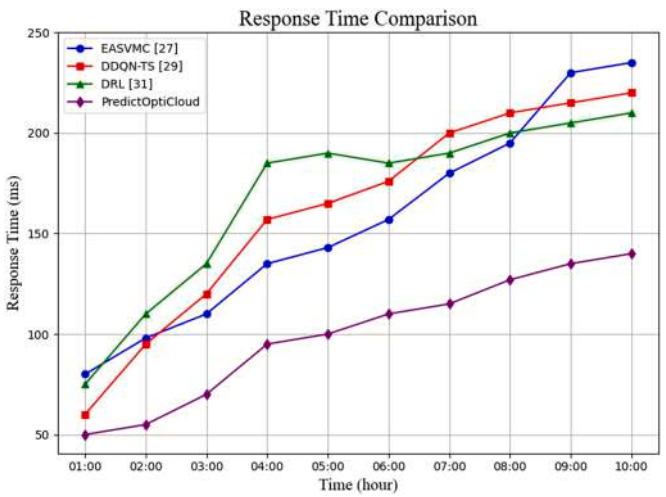


Fig. 10. Comparison of response time.

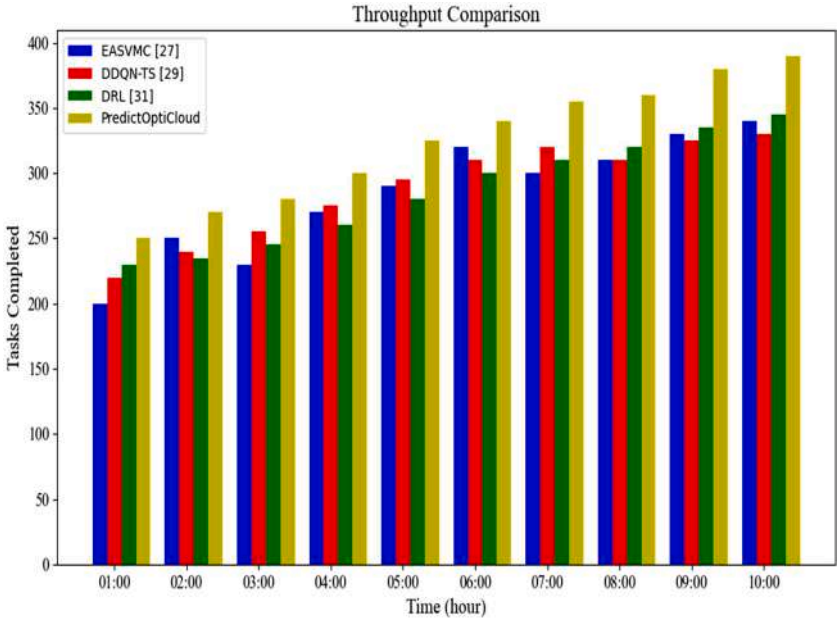


Fig. 11. Throughput comparison.

[31] and the PredictOptiCloud, performance in the context of task completion rates, spanning from 01:00 to 10:00 and covering a range of 0 to 400 tasks. The throughput increases rapidly from 250 to 385 at 10:00. But, other methods, possess about 200, 210,212 at the initation and also at the end of about 300, 302, 308 simultaneously. Among these methods, PredictOptiCloud stands out for its remarkable throughput efficiency. This exceptional performance can be attributed to a series of methodological strategies and equations employed within the PredictOptiCloud framework. Through dynamic workload distribution (Eq. 31), tasks are intelligently allocated to servers to maintain a balanced workload, ensuring that throughput remains consistently high. Task assignments based on performance metrics (Eq. 24) guarantee that tasks are routed to the most suitable servers, minimizing delays and optimizing throughput. Real-time feedback integration (Eq. 28) allows PredictOptiCloud to adapt to changing conditions, efficiently redistributing tasks when needed. Moreover, the method's cost-efficient offloading decisions (Eq. 33) ensure that cost implications are considered without compromising throughput. In summary, PredictOptiCloud's ability to intelligently manage workloads, adapt to dynamic conditions, and balance task assignments based on performance and cost factors leads to its outstanding throughput performance. This ensures tasks are completed promptly and efficiently, making it a valuable asset for e-commerce platforms seeking optimal operational efficiency.

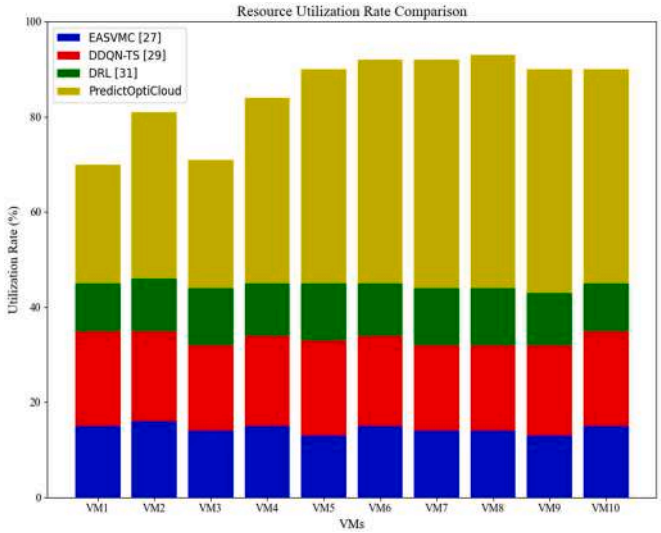


Fig. 12. Comparison of resource utilization rate.

The resource utilization rate, as shown in the Fig. 12, provides a comprehensive comparison among various methods, including EASVMC [27], DDQN-TS [29], DRL [31], and the proposed PredictOptiCloud. PredictOptiCloud stands out as a top performer in efficiently managing VM resources. This measures about 62 % to 82 % for 10VMs. But, other methods possess a low utilization rate of 42 %, 38 % and 18 % at VM1. This remarkable resource utilization is achieved through dynamic task distribution based on workload characteristics and resource availability, with Eq. (31) playing a pivotal role in ensuring a balanced workload. By intelligently allocating tasks to VMs, PredictOptiCloud minimizes resource underutilization and overload, resulting in high resource utilization rates. PredictOptiCloud optimizes task assignments using performance metrics, as described in Eq. (24). This approach ensures that tasks are routed to the most suitable VMs, taking into account factors such as response time, energy consumption, and capacity. By assigning tasks to VMs that can handle them best, resource utilization is enhanced, leading to greater efficiency. If a particular VM consistently underperforms, PredictOptiCloud can adapt and reassign tasks to other VMs with better performance, further improving resource utilization. Moreover, PredictOptiCloud's consideration of cost-efficient offloading decisions, as represented by Eq. (33), ensures that resource allocation is optimized not only in terms of performance but also in a cost-effective manner. Informed decisions about when to offload tasks help allocate resources judiciously, avoiding unnecessary expenses.In this context, the 10 VMs considered for visualization serve as a snapshot of PredictOptiCloud's resource utilization proficiency.

The cost efficiency comparison Fig. 13 provides a clear visualization of the cost-effectiveness of different methods, including EASVMC [27], DDQN-TS [29], DRL [31], and the proposed PredictOptiCloud. Notably, PredictOptiCloud stands out as a cost-efficient solution, 3512 tasks in 5 VMs, and at 3539 tasks at 50 VMs. Here, the DDQN-TS achieves a task offloading of 3540 tasks at 50 VMs. PredictOptiCloud's cost efficiency superiority is rooted in its ability to make informed offloading decisions, as detailed in Eq. (33). This equation ensures that the framework considers various cost-related factors when deciding whether to execute a task locally or offload it to other resources. The aim is to optimize performance while keeping costs in check, which is crucial for cost efficiency. The ability of PredictOptiCloud to dynamically adapt to changing conditions, as illustrated in Eq. (28), is another key factor contributing to its cost efficiency. By continuously monitoring performance and making real-time adjustments, PredictOptiCloud can identify underperforming resources and allocate tasks to more efficient ones. This adaptability optimizes resource usage and minimizes unnecessary costs. Additionally, PredictOptiCloud considers the anticipated traffic surges, as shown in Eq. (26). By taking into account predicted traffic increases and their associated latency, the framework can make decisions that prevent service degradation while avoiding excessive costs.

The comparison of conversion rates(c_r) in the Fig. 14 provides valuable insights into the effectiveness of various methods, including EASVMC [27], DDQN-TS [29], DRL [31], and the proposed PredictOptiCloud, in driving user actions such as making a purchase. A high CR indicates that the platform excels in guiding visitors through a well-designed user journey and encouraging them to make purchases.The high conversion rate achieved by PredictOptiCloud can be attributed to its methodology, which integrates various factors to enhance user experience and drive conversions. Here, the rate is about 10 % during initial hours and it increases to about 12 % during the final hours of a day. In the context of PredictOptiCloud, two essential components significantly impact conversion rates: such as $u_x$andc_r. u_x is paramount in e-commerce, as a seamless and responsive platform encourages visitors to complete transactions. The u_x score, encompasses metrics such as page load time, server response time, and transaction processing efficiency. The framework's ability to optimize resource allocation, load balancing, and offloading decisions contributes to an efficient and responsive user experience. Eq. (19) equation outlines how the framework associates a performance score and conversion rate with each solution, reflecting how effectively tasks are scheduled within that specific solution. The better the $u_x$and performance, the higher the conversion rate. PredictOptiCloud ensures that tasks are assigned to resources (VMs or servers) that can execute them swiftly and efficiently, reducing latency and enhancing the user experience. Eq. (32) defines the cost function, where execution time, communication

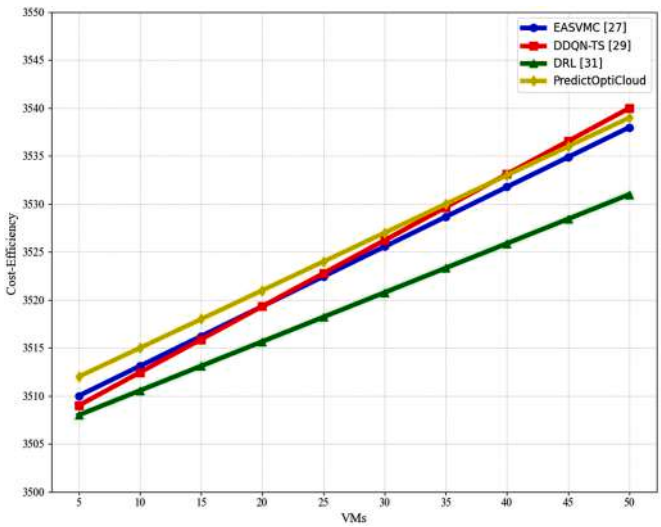


Fig. 13. Comparison of cost efficiency.

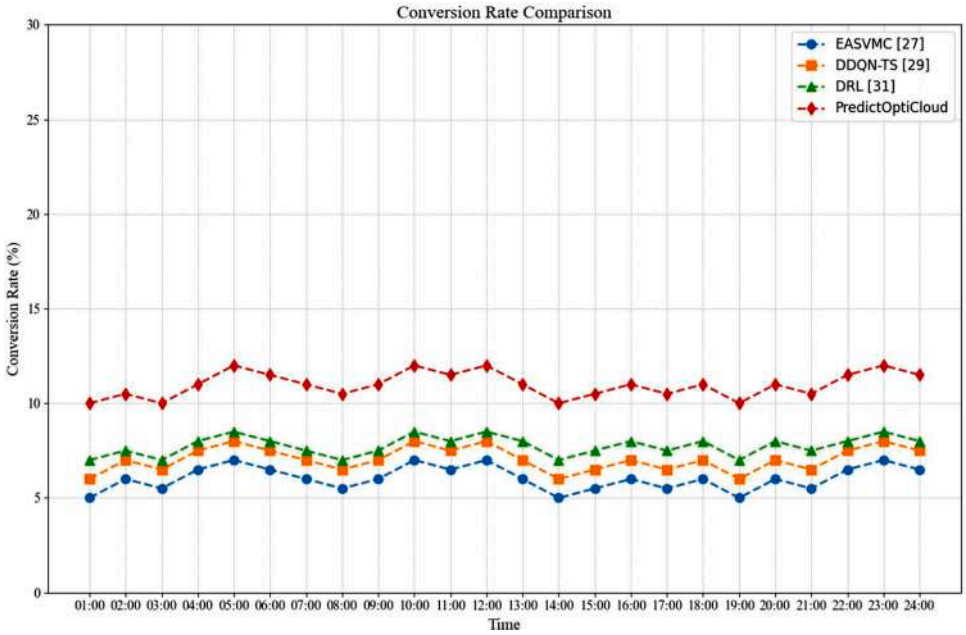


Fig. 14. Comparison of conversion rates.

latency, and server utilization play critical roles. By minimizing these factors, PredictOptiCloud optimizes the allocation of tasks to maximize user satisfaction.The combination of an excellent user experience and optimized task assignment based on capacity leads to high conversion rates. In other words, PredictOptiCloud excels at providing users with a smooth, responsive, and efficient experience, encouraging them to take desired actions, such as making purchases. This methodology allows e-commerce platforms to achieve a high c_r, indicating a well-designed user journey and successful marketing and web design strategies.

The rate of successful task offloading, as represented by Eq. (34), is a crucial metric for evaluating the effectiveness of offloading decisions in the context of various methods, including EASVMC [27], DDQN-TS [29], and DRL [31]. This metric provides insights into how well offloading decisions align with available resources and the current demands of the system. The Fig. 15 depicting the rate of successful task offloading for different methods showcases PredictOptiCloud's remarkable performance in achieving a high success rate such as 70 % in week 1 to 82 % in week 7. The other methods such as DRL, gets only 65 % in week 1 to 68 % in week 7. This high success rate is attributed to the methodology and decision-making processes employed by PredictOptiCloud. The success of

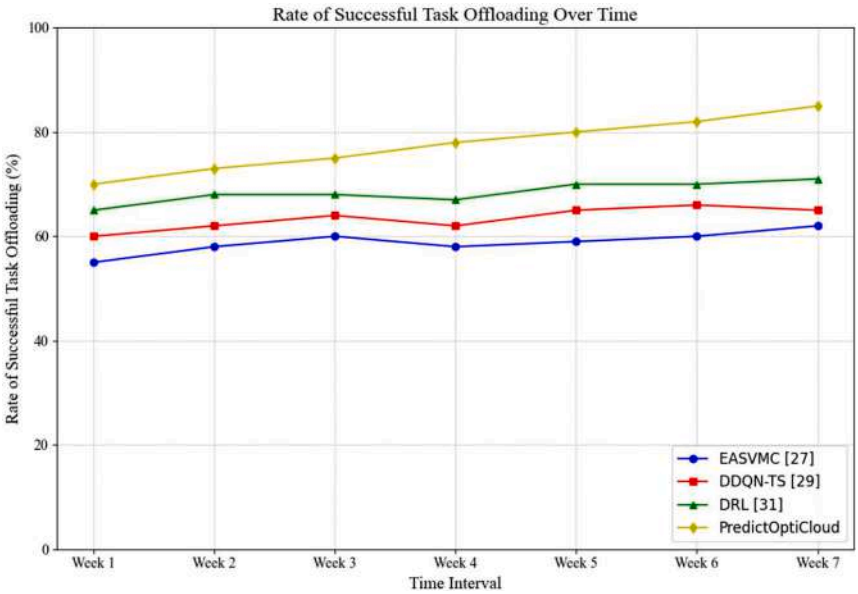


Fig. 15. Rate of successful task offloading over time.

PredictOptiCloud can be explained through the following aspects of its methodology:

Intelligent Offloading Decisions: PredictOptiCloud leverages a combination of performance metrics, cost considerations, and real-time feedback to make informed offloading decisions. The cost function, as defined in Eq. (33), incorporates elements such as execution time, communication latency, and server utilization. This ensures that tasks are offloaded to resources that can efficiently and swiftly execute them, minimizing response times and maintaining user satisfaction.

Adaptation to Workload Characteristics: PredictOptiCloud's methodology includes real-time feedback loops that adapt offloading decisions based on the current performance of resources. If a server consistently underperforms or exhibits latency, the framework reduces the assignment of tasks to such resources, thus increasing the rate of successful task offloading.

Predicted Traffic Handling: The framework considers anticipated traffic surges, as indicated in Eq. (26), and offloads tasks to prevent compromising user experience during peak demands. This proactive approach ensures a high success rate in offloading decisions during varying workload conditions. By efficiently managing the task offloading process, PredictOptiCloud maximizes the rate of successful task transfers, resulting in improved resource utilization, user satisfaction, and overall system performance. The demonstrated high success rate in the Figure underscores the effectiveness of the PredictOptiCloud framework in optimizing offloading decisions for e-commerce platforms.

4.4. Task scheduling performance for hybrid workloads

a. Differentiated workload analysis:

The evaluation of how efficiently the PredictOptiCloud framework differentiates between static and dynamic workloads is a critical aspect of its performance. This differentiation is essential for optimizing resource allocation and decision-making in response to varying workload characteristics. To measure the accuracy of this differentiation, a comprehensive analysis is conducted in Fig. 16 that considers precision, accuracy, task volume, and response time as its axes. The performance of PredictOptiCloud is compared with other methods to highlight its superior capabilities. The higher performance is due to

Precise differentiation: PredictOptiCloud employs a combination of predictive analytics and real-time feedback to accurately classify workloads. The cost function (Eq. 33) considers execution time, communication latency, and server utilization, helping distinguish between workloads effectively.

High accuracy: The framework's use of spider monkey optimization and grey wolf optimization enhances its overall accuracy. These intelligent algorithms ensure that resources are allocated correctly based on predicted workloads, reducing the chances of misclassifications.

Optimal task volume: PredictOptiCloud balances task volumes based on predictive workload analysis, as outlined in the methodology. This ensures that tasks are evenly distributed, preventing overloading or underutilization of resources.

Efficient response Times: The methodology's real-time adaptation to workload characteristics and predicted traffic surges allows PredictOptiCloud to maintain efficient response times for dynamic workloads. Offloading decisions are made proactively to meet performance requirements during peak demands.

The spider chart analysis highlights PredictOptiCloud's superior performance in workload differentiation, emphasizing its

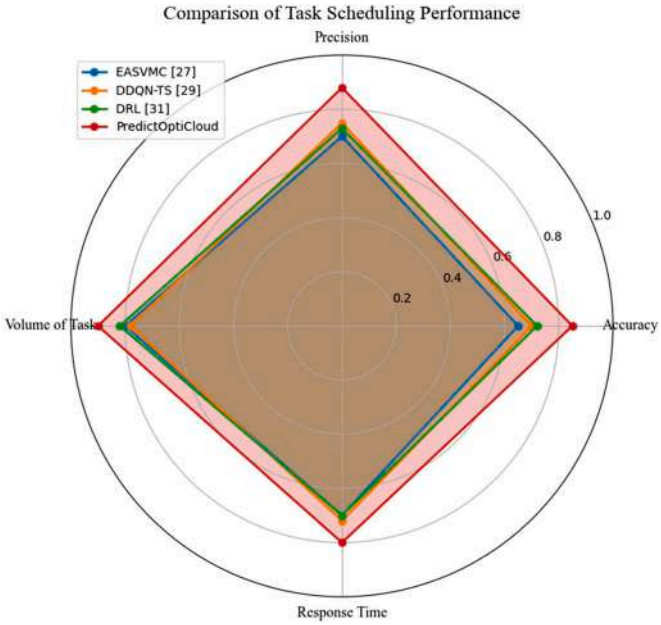


Fig. 16. Spider chart for comparison of task scheduling performance.

precision, accuracy, balanced task volume, and efficient response times compared to other methods. This enhanced ability to differentiate and respond to hybrid workloads ensures optimal resource allocation and user satisfaction in e-commerce platforms. b. Task scheduling latency:

Task scheduling latency is a crucial metric that measures the time taken to schedule a task once it enters the system. This latency analysis provides insights into how efficiently a system can manage and allocate tasks for both static and dynamic workloads. The 3D surface plots in Fig. 17 visually represent the changes in latency over time for these two workload types, with static represented as 0 and dynamic as 1.On the analysis of latency for both static and dynamic workloads the following observations are made. The trends observed in the 3D surface plots are explained here.

Static workload latency:

At time 0 (the start of the simulation), the latency is low for static workloads. This low latency indicates that the system efficiently schedules tasks as they enter the system. As time progresses, the latency gradually increases to a maximum value. This increase can be attributed to the accumulation of static tasks in the system, resulting in slightly longer scheduling times.

Dynamic workload latency:

Similar to the static workload, the latency for dynamic workload tasks is initially low. The framework efficiently schedules incoming dynamic tasks. As time progresses, the latency for dynamic workloads also increases gradually. This can be attributed to the increasing volume of dynamic tasks and the associated scheduling overhead.

The 3D surface plots illustrate the changing latency over time, with static and dynamic workloads clearly differentiated. The plots show that the system initially maintains low latency for both workload types. The increase in latency for both static and dynamic workloads as time progresses signifies the natural accumulation of tasks and scheduling complexity. The periodic dips in latency are indicative of the system's efficiency in task scheduling, which results from adaptive algorithms used in PredictOptiCloud. The observed trends in the 3D surface plots highlight the framework's ability to efficiently schedule both static and dynamic tasks. The initial low latency and periodic improvements in scheduling efficiency can be attributed to the PredictOptiCloud methodology's real-time adaptation and resource allocation techniques, which help in managing tasks effectively. c. Resource allocation efficiency:

Resource allocation efficiency is a critical aspect of any cloud computing framework, especially during surge periods when there is a sudden increase in workload demands. The evaluation of resource allocation efficiency aims to determine how well resources are allocated to tasks and whether the system effectively adapts to varying workloads. The stacked area chart in the provided figure visualizes this efficiency, comparing regular volume, surge volume, resource allocation by PredictOptiCloud, and the maximum available resources.

The stacked area chart for resource allocation efficiency presented in Fig. 18 provides a visual representation of how efficiently resources are allocated during regular and surge periods. When there is a surge in workload, PredictOptiCloud effectively increases resource allocation to meet the demands. This is reflected in the chart by observing an increase in resource allocation during surge periods. The chart demonstrates that PredictOptiCloud manages resource allocation efficiently, ensuring that resources are allocated in

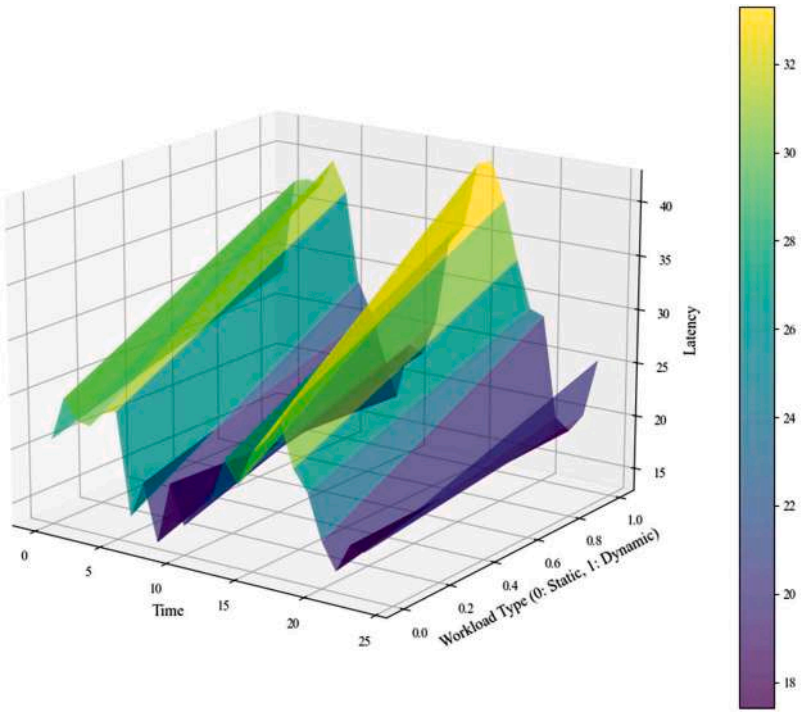


Fig. 17. Task scheduling latency.

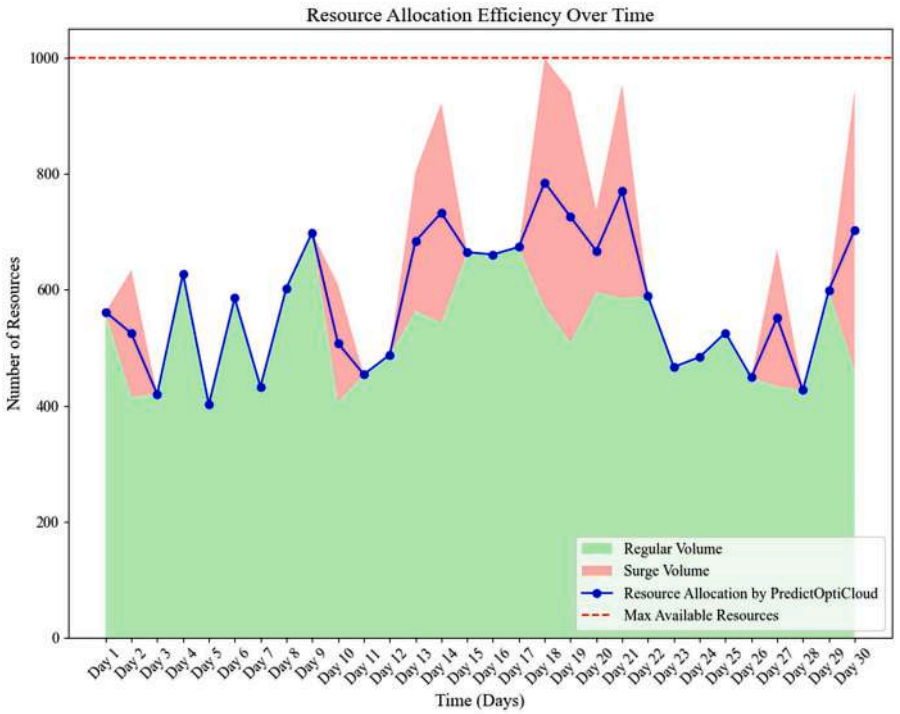


Fig. 18. Resource allocation efficiency.

accordance with the workload's needs. The methodology behind PredictOptiCloud, involving real-time prediction and adaptive resource allocation algorithms, is crucial in achieving this efficiency. d. Resource churn rate:

Resource churn rate is a crucial metric that assesses how often tasks are reassigned or offloaded from one resource (e.g., VM or server) to another within the cloud infrastructure. A high churn rate can be indicative of inefficiency in resource allocation and task management. The Fig. 19 presenting a heat map for measuring the churn rate provides valuable insights into when and how frequently resource reassignments occur, with a focus on each VM's perspective over a 30-day period. The heat map provides a visual representation of churn rates for different VMs over a 30-day period as in Fig. 19. Each cell in the heat map corresponds to a specific VM, and the color intensity indicates the churn rate. High churn rates, resulting in resource reassignments, are often observed during periods when there are dynamic workload changes, such as weekends and festive days. By assessing churn rates, PredictOptiCloud refine its resource allocation and offloading strategies to minimize inefficiencies, especially during unpredictable workload spikes.

4.5. Complexity analysis

The complexity analysis in Table 2 provides insights into the computational requirements of the proposed methods within the PredictOptiCloud framework. Data collection involves linear complexity O (N), where N represents the total size of the data sources, indicating that the time needed scales linearly with the dataset size. Embedding generation complexity O (M * D) depends on the lexicon size (M) and embedding dimensionality (D), influencing the computational load. The Bi LSTM processing and attention mechanism complexities O (T * Hˆ2) are determined by the sequence length (T) and hidden state dimension (H), reflecting the computational demand for processing sequential data and computing attention weights. Workload prediction complexity O (T * F) is influenced by the number of timestamps (T) and feature vector dimensionality (F), indicating the computational effort required for forecasting future workload patterns. Classification complexity O (T * K) for categorizing workload into stable and dynamic components is influenced by the number of classes (K) and sequence length (T), impacting the classification process. Lastly, SWO complexity O (I * S * C) depends on the number of iterations (I), solution space size (S), and evaluation cost per solution (C), reflecting the computational resources needed for optimization. Overall, the discussed complexities highlight the computational challenges associated with various components of the proposed methods, underscoring the need for efficient algorithms and computational resources to implement the PredictOptiCloud framework effectively.

5. Discussion

The experimental results in Section 4.4 underscored PredictOptiCloud's effectiveness in optimizing resource allocation, load balancing, and task offloading within e-commerce platforms. By using real-world and synthetic datasets, PredictOptiCloud demonstrated adaptability to diverse workload patterns, ensuring efficient task completion rates and response times. The framework's

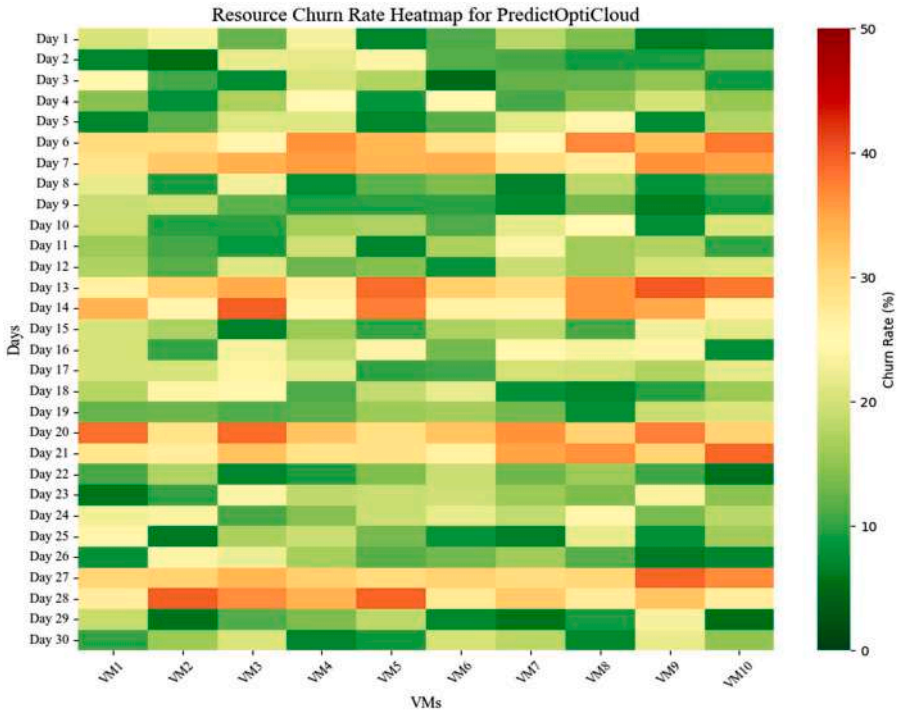


Fig. 19. Resource churn rate.

Table 2
Complexity analysis.

Component	Complexity
Data Collection	O(N)
Embedding Generation	O(M * D)
Bi LSTM Processing	O(T * H^2)
Attention Mechanism	O(T * H^2)
Workload Prediction	O(T * F)
Classification	O(T * K)
Spider Wolf Optimization	O(I * S * C)

predictive analytics and real-time feedback loop facilitated proactive decision-making, leading to timely adjustments in resource allocation and load balancing. Moreover, PredictOptiCloud's consideration of cost-efficient offloading decisions contributed to optimizing system performance while effectively managing costs. In the subsequent case Analysis (Section 4.5), PredictOptiCloud's performance using synthetic e-commerce data further highlighted its efficiency in workload management, with low response times and high throughput rates observed across various datasets. Notably, the framework's optimization of resource utilization led to higher conversion rates, particularly in the retail e-commerce dataset, indicating its effectiveness in driving successful transactions and enhancing user satisfaction. Moving on to Task Scheduling Performance for Hybrid Workloads (Section 4.6), PredictOptiCloud exhibited precision and accuracy in classifying static and dynamic workloads, maintaining balanced task volumes and efficient response times. Task scheduling latency analysis emphasized PredictOptiCloud's efficiency in managing task scheduling under both workload types, with adaptive scheduling observed. Furthermore, the evaluation of resource allocation efficiency highlighted PredictOptiCloud's capability in optimizing resource utilization during surge periods, ensuring efficient allocation based on workload demands. Finally, the Complexity Analysis (Section 4.7) provided insights into the computational requirements of PredictOptiCloud, emphasizing the need for efficient algorithms and computational resources.

The experiment results obtained from the evaluation of PredictOptiCloud's performance in managing hybrid e-commerce workloads provide valuable insights into its effectiveness in optimizing resource allocation, task scheduling, and response time management. Here is a detailed analysis on the reasonability of the experiment:

Precision in workload differentiation: The experiment results demonstrate PredictOptiCloud's ability to accurately differentiate between static and dynamic workloads. By using predictive analytics and real-time feedback mechanisms, PredictOptiCloud achieves precise workload classification, minimizing misclassifications and ensuring optimal resource utilization. This aligns with the objective of developing a framework capable of dynamically adapting to varying workload characteristics, thereby enhancing operational

efficiency.

Efficient task scheduling and response times: The analysis of task scheduling latency and response times reveals that PredictOptiCloud effectively manages and allocates tasks, maintaining low latency even during peak workload periods. The dynamic adjustment of resource allocation during surge periods ensures that tasks are completed promptly, contributing to improved user experience and operational performance. These findings support the objective of developing a framework that can efficiently handle fluctuations in workload demands, thereby ensuring reliable and responsive service delivery.

Resource allocation efficiency: The evaluation of resource allocation efficiency highlights PredictOptiCloud's ability to optimize resource utilization, particularly during surge periods. By dynamically adjusting resource allocation based on workload demands, PredictOptiCloud minimizes resource underutilization and overload, enhancing overall system efficiency. This aligns with the objective of developing a framework capable of maximizing resource utilization while minimizing operational costs, ultimately improving the platform's competitiveness in the e-commerce market.

Adaptive offloading strategies: The analysis of resource churn rate demonstrates PredictOptiCloud's effectiveness in managing task offloading and reassignments within the cloud infrastructure. By refining offloading strategies based on churn rate data, PredictOptiCloud minimizes inefficiencies and ensures seamless task execution, even during unpredictable workload spikes. This aligns with the objective of developing a framework that can dynamically adapt to changing workload conditions, thereby optimizing resource usage and enhancing overall system performance.

5.1. Limitations and Implementation challenges in real world scenario

PredictOptiCloud, while promising for optimizing e-commerce resource management, faces several limitations. Challenges include reliance on data quality and availability, complexity in model training, and integration hurdles with existing IT systems. Scalability concerns arise regarding handling high-volume traffic, while potential overfitting of machine learning models and real-time adaptability constraints add complexity. Addressing these limitations demands ongoing research, industry collaboration, and rigorous testing to ensure PredictOptiCloud's effectiveness and scalability in real-world e-commerce environments. The discussion of implementation challenges is presented below.

Data collection: One of the key challenges in implementing PredictOptiCloud in a real-world setting is the acquisition and management of e-commerce data. Real-world e-commerce platforms generate vast amounts of data, including user interactions, transactions, server logs, and event data. Collecting, cleaning, and processing this data to train the predictive models and optimize resource allocation can be complex and resource-intensive. Furthermore, ensuring the quality and reliability of the collected data is paramount for the accuracy and effectiveness of PredictOptiCloud.

Training time: Another challenge is the computational resources and time required for training the machine learning models used in PredictOptiCloud, particularly for tasks such as workload prediction and optimization. Training complex models like hierarchical attention Bi-LSTM on large-scale datasets may demand significant computational power and time. Moreover, fine-tuning and optimizing these models to achieve desired performance metrics may require iterative experimentation and tuning, further increasing the time and resource requirements.

Integration with existing systems: Integrating PredictOptiCloud into existing e-commerce platforms and IT infrastructures poses another set of challenges. Compatibility issues, data format discrepancies, and interoperability with legacy systems need to be addressed during the integration process. Additionally, ensuring seamless communication and synchronization between PredictOptiCloud and other components of the e-commerce ecosystem, such as inventory management systems, payment gateways, and customer relationship management tools, is crucial for its effective deployment and operation.

Scalability and deployment complexity: Scalability is a critical consideration when deploying PredictOptiCloud in real-world e-commerce environments. The framework should be capable of handling increasing data volumes, user traffic, and computational demands as the e-commerce platform grows. Deploying PredictOptiCloud across distributed computing environments, cloud platforms, and hybrid infrastructures introduces complexities related to resource provisioning, orchestration, and management. Moreover, ensuring fault tolerance, resilience, and high availability of the system under varying load conditions adds to the deployment complexity.

Addressing the challenges of implementing PredictOptiCloud in real-world settings involves several steps. First, ensuring reliable data collection and quality checks are vital for accurate predictions. Then, optimizing the training process of machine learning models by using distributed computing and fine-tuning techniques can speed up the process. Integrating PredictOptiCloud with existing e-commerce systems requires standardized protocols and thorough compatibility testing. Designing PredictOptiCloud as a cloud-native application allows for scalability and flexibility, aided by auto-scaling features. Continuous monitoring and refinement are essential for maintaining PredictOptiCloud's effectiveness in different e-commerce environments.

6. Conclusion

In conclusion, PredictOptiCloud emerges as a potent and comprehensive solution to address the intricate challenges posed by dynamic workloads in the e-commerce sector. This framework has demonstrated its ability to optimize resource allocation, efficiently manage server loads, enhance user experiences, reduce energy consumption, and minimize operational costs. Through its systematic approach, including data collection, embedding generation, domain-specific hierarchical attention, and Spider Wolf Optimization (SWO) for load balancing and offloading, PredictOptiCloud proves to be an indispensable tool for e-commerce platforms. The performance analysis of PredictOptiCloud, based on a range of vital metrics such as response time, throughput, resource utilization rate,

cost-efficiency, conversion rate, rate of successful task offloading, precision, accuracy, task volume, and churn rate, validates its effectiveness in delivering optimal user satisfaction, operational efficiency, and financial prudence. By seamlessly adapting to both predictable and unpredictable workloads, PredictOptiCloud empowers e-commerce platforms to operate with agility and excellence. The e-commerce industry continues to evolve rapidly, and PredictOptiCloud offers a valuable solution for platforms striving to meet these challenges head-on. However, future research and development efforts will focus on mitigating the challenges presented and providing practical solutions to enable the seamless implementation and adoption of PredictOptiCloud in real-world e-commerce settings.

Compliance with ethical standards

Funding: There is no funding for this study.

Ethical approval: This article does not contain any studies with human participants and/or animals performed by any of the authors.

Informed consent: There is no informed consent for this study.

Declaration of competing interest

Authors declare that they have no conflict of interest.

Data availability

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Sugan J, Isaac Sajan R. The first draft of the manuscript was written by Sugan J and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References

[1] E.H. Houssein, A.G. Gad, Y.M. Wazery, P.N. Suganthan, Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends, Swarm. Evol. Comput. 62 (2021) 100841.
[2] N. Kaur, A. Kumar, R. Kumar, A systematic review on task scheduling in fog computing: taxonomy, tools, challenges, and future directions, Concurr. Comput.: Pract. Exp. 33 (21) (2021) e6432.
[3] S.A. Bello, L.O. Oyedele, O.O. Akinade, M. Bilal, J.M.D. Delgado, L.A. Akanbi, A.O. Ajayi, H.A. Owolabi, Cloud computing in construction industry: use cases, benefits and challenges, Autom. Constr. 122 (2021) 103441.
[4] S.S. George, R.S. Pramila, A review of different techniques in cloud computing, Mater. Today: Proc. 46 (2021) 8002–8008.
[5] H. Singh, S. Tyagi, P. Kumar, S.S. Gill, R. Buyya, Metaheuristics for scheduling of heterogeneous tasks in cloud computing environments: analysis, performance evaluation, and future directions, Simul. Model. Pract. Theory. 111 (2021) 102353.
[6] Z. Ahmad, A.I. Jehangiri, M.A. Ala'anzy, M. Othman, R. Latip R, S.K.U. Zaman, A.I. Umar, Scientific workflows management and scheduling in cloud computing: taxonomy, prospects, and challenges, IEEE Access 9 (2021) 53491–53508.
[7] P. Hosseinioun, M. Kheirabadi, S.R. Kamel Tabbakh, R. Ghaemi, aTask scheduling approaches in fog computing: a survey, Trans. Emerg. Telecommun. Technol. 33 (3) (2022) e3792.
[8] M. Agarwal, G.M.S. Srivastava, Opposition-based learning inspired particle swarm optimization (OPSO) scheme for task scheduling problem in cloud computing, J. Ambient. Intell. Humaniz. Comput. 12 (10) (2021) 9855–9875.
[9] R. Medara, R.S. Singh, Energy efficient and reliability aware workflow task scheduling in cloud environment, Wirel. Pers. Commun. 119 (2) (2021) 1301–1320.
[10] S.E. Shukri, R. Al-Sayyed, A. Hudaib, S. Mirjalili, Enhanced multi-verse optimizer for task scheduling in cloud computing environments, Expert. Syst. Appl. 168 (2021) 114230.
[11] N.K. Walia, N. Kaur, M. Alowaidi, K.S. Bhatia, S. Mishra, N.K. Sharma, S.K. Sharma, H. Kaur, An energy-efficient hybrid scheduling algorithm for task scheduling in the cloud computing environments, IEEE Access 9 (2021) 117325–117337.
[12] M. Tanha, M. Hosseini Shirvani, A.M. Rahmani, A hybrid meta-heuristic task scheduling algorithm based on genetic and thermodynamic simulated annealing algorithms in cloud computing environments, Neural Comput. Applic. 33 (2021) 16951–16984.
[13] M. Abdullahi, M.A. Ngadi, S.I. Dishing, S.I.M. Abdulhamid, An adaptive symbiotic organisms search for constrained task scheduling in cloud computing, J. Ambient. Intell. Humaniz. Comput. (2022) 1–12.
[14] B.M.H. Zade, N. Mansouri, M.M. Javidi, SAEA: a security-aware and energy-aware task scheduling strategy by Parallel Squirrel Search Algorithm in cloud environment, Expert. Syst. Appl. 176 (2021) 114915.
[15] L. Abualigah, M.A. Elaziz, N. Khodadadi, A. Forestiero, H. Jia, A.H. Gandomi, Aquila optimizer based PSO swarm intelligence for IoT task scheduling application in cloud computing. Integrating Meta-Heuristics and Machine Learning for Real-World Optimization Problems, Springer International Publishing, Cham, 2022, pp. 481–497.
[16] M. Hussain, L.F. Wei, A. Lakhan, S. Wali, S. Ali, A. Hussain, Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing, Sustain. Comput.: Inf. Syst. 30 (2021) 100517.
[17] S. Azizi, M. Shojafar, J. Abawajy, R. Buyya, Deadline-aware and energy-efficient IoT task scheduling in fog computing systems: a semi-greedy approach, J. Netw. Comput. Applic. 201 (2022) 103333.
[18] H. Zhang, Y. Wu, Z. Sun, EHEFT-R: multi-objective task scheduling scheme in cloud computing, Complex. Intell. Systems. (2021) 1–8.
[19] X. Fu, Y. Sun, H. Wang, H. Li, Task scheduling of cloud computing based on hybrid particle swarm algorithm and genetic algorithm, Cluster. Comput. (2021) 1–10.
[20] X. Guo, Multi-objective task scheduling optimization in cloud computing based on fuzzy self-defense algorithm, Alex. Eng. J. 60 (6) (2021) 5603–5609.
[21] H. Mahmoud, M. Thabet, M.H. Khafagy, F.A. Omara, An efficient load balancing technique for task scheduling in heterogeneous cloud environment, Cluster. Comput. 24 (4) (2021) 3405–3419.
[22] G. Rjoub, J. Bentahar, O. Abdel Wahab, A. Saleh Bataineh, Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems, Concurr. Comput.: Pract. Exp. 33 (23) (2021) e5919.
[23] L. Abualigah, M. Alkhrabsheh, Amended hybrid multi-verse optimizer with genetic algorithm for solving task scheduling problem in cloud computing, J. Supercomput. 78 (1) (2022) 740–765.

[24] M. Grzegorowski, E. Zdravevski, A. Janusz, P. Lameski, C. Apanowicz, D. Ślęzak, Cost optimization for big data workloads based on dynamic scheduling and cluster-size tuning, Big Data Res. 25 (2021) 100203.
[25] B. Wang, C. Wang, W. Huang, Y. Song, X. Qin, Security-aware task scheduling with deadline constraints on heterogeneous hybrid clouds, J. Parallel. Distrib. Comput. 153 (2021) 15–28.
[26] F. Ebadifard, S.M. Babamir, Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment, Cluster. Comput. 24 (2021) 1075–1101.
[27] R. Medara, R.S. Singh, Energy-aware workflow task scheduling in clouds with virtual machine consolidation using discrete water wave optimization, Simul. Model. Pract. Theory. 110 (2021) 102323.
[28] S. Hu, Y. Xiao, Design of cloud computing task offloading algorithm based on dynamic multi-objective evolution, Future Gener. Comput. Syst. 122 (2021) 144–148.
[29] Z. Tong, F. Ye, B. Liu, J. Cai, J. Mei, DDQN-TS: a novel bi-objective intelligent scheduling algorithm in the cloud environment, Neuro Computing 455 (2021) 419–430.
[30] J. Liu, Z. Wu, D. Feng, M. Zhang, X. Wu, X. Yao, D. Yu, Y. Ma, F. Zhao, D. Dou, Heterps: distributed deep learning with reinforcement learning based scheduling in heterogeneous environments, Future Gener. Comput. Syst. (2023).
[31] Y. Zhang, H. Zhu, D. Tang, T. Zhou, Y. Gui, Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems, Robot. Comput. Integr. Manuf. 78 (2022) 102412.
[32] S. Mangalampalli, S.K. Swain, V.K. Mangalampalli, Multi objective task scheduling in cloud computing using cat swarm optimization algorithm, Arab. J. Sci. Eng. 47 (2) (2022) 1821–1830.
[33] S. Nabi, M. Ahmed, PSO-RDAL: particle swarm optimization-based resource-and deadline-aware dynamic load balancer for deadline constrained cloud tasks, J. Supercomput. 78 (4) (2022) 4624–4654.
[34] A. Pradhan, S.K. Bisoy, A novel load balancing technique for cloud computing platform based on PSO, J. King Saud Univ.-Comput. Inf. Sci. 34 (7) (2022) 3988–3995.
[35] A. Choudhary, R. Rajak, A novel strategy for deterministic workflow scheduling with load balancing using modified min-min heuristic in cloud computing environment, Cluster. Comput. (2024) 1–22.
[36] A. Ullah, N.M. Nawi, An improved in tasks allocation system for virtual machines in cloud computing using HBAC algorithm, J. Ambient. Intell. Humaniz. Comput. 14 (4) (2023) 3713–3726.
[37] K. Siddesha, G.V. Jayaramaiah, C. Singh, A novel deep reinforcement learning scheme for task scheduling in cloud computing, Cluster. Comput. 25 (6) (2022) 4171–4188.
[38] H. Zhou, A novel approach to cloud resource management: hybrid machine learning and task scheduling, J. Grid. Comput. 21 (4) (2023) 68.
[39] Q. Su, L. Chen, A method for discovering clusters of e-commerce interest patterns using click-stream data, Electron. Commer. Res. Applic. 14 (1) (2015) 1–3.
[40] C. Xu, X. Wang, B. Hu, D. Zhou, Y. Dong, C. Huo, W. Ren, Graph attention networks for new product sales forecasting in e-commerce, in: Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, Springer International Publishing, 2021, pp. 553–565. April 11–14, 2021Proceedings, Part III 26 2021.
[41] Z. Huang, D. Zeng, H.A. Chen, comparative study of recommendation algorithms in e-commerce applications, IEEE Intell. Syst. 22 (5) (2007) 68–78.