Received: July 24, 2023. Revised: August 24, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Heuristically Modified LSTM-Based Reinforcement Learning for Task offloading in Industrial IoT Edge Computing

Udayakumar K[1] **Ramamoorthy S[1]***

[1]*Department of Computing Technologies,*
SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu-603 203, India
Corresponding author's Email: ramamoos@srmist.edu.in

Abstract: An intensive computation source has become increasingly important in recent years to meet the intensive resource and low-latency needs of industrial internet of things (IIoT) systems. Existing IIoT devices are built with limited computational resource, delivering results in a limited fashion when used in highly resource-intensive and delay-sensitive applications. It is difficult to process time-critical IIoT task due to varying demand like low latency, intensive computation and high data transmission. Offloading computing tasks to mobile edge computing (MEC) servers in the network's perimeter can effectively reduce delay. However, MEC server collected fewer resources than the resource cloud. To improve the resource utilization and minimize cost, this research develops an adaptive task offloading decision model through multi-constraint objective function. The goal is to minimize service delay, energy consumption, and maximize resource utilization through prediction based decision model. This study examines a non-orthogonal multiple access (NOMA) -based MEC for IIoT system, where edge nodes offload their tasks to nearby edge servers for execution. Heuristically modified long short-term memory (H-LSTM) employing hybrid cat and mouse dingo optimization (HCMDO)-based reinforcement learning is suggested to distribute tasks optimally. We formulate joint optimization by considering multiple parameters using HCMDO. Further, these optimal parameters are used in training H-LSTM along with benchmark dataset. The outcome of the H-LSTM network utilized in deep reinforcement learning (DRL) to improve convergence speed, accuracy and stability by predicting task and best server. Average energy consumption analysis performed in the developed model attained 19.8%, 15.1%, 16.9%, and 15.6% than conventional approaches. In addition, the experimental results shows developed model attain better outcome in terms of delay and resource utilization.

Keywords: Hybrid cat and mouse dingo optimization, Industrial IoT, Long short term memory, Edge computing, Reinforcement learning, Task offloading.

1. Introduction

Industry automation ensures efficient production with little need for labour. Automating design, analysis, assembly lines, warehouse management, and logistics depends on smart devices largely. Greater process flexibility, higher production quality, and more revenue are requirements of the Industry 4.0 revolution. Industry 4.0 encompasses many technologies like advanced robotics, the internet of things (IoT), machine learning (ML), augmented & virtual reality (AR & VR), big data analytics, cloud computing, and cyber security. Industry 4.0 includes security applications, augmented/virtual reality devices, real-time cyber-physical systems, and autonomous vehicles, which demand low latency with deadline and intensive computing resource, which we term time-critical applications. Even though cloud has intense resource, it delivers high latency. The delay in offloading and processing such applications leads to a severe loss of money and human lives in the industry. Offloading computing tasks to MEC servers in the network's perimeter can effectively reduce delay. In recent times, several enhancements have been performed in fifth-generation (5G) cellular technology and allow different applications like automatic driving, the IIoT,

and augmented reality (AR) [1]. Due to a lack of spectrum resources, the IIoT has been hampered in its development.

Recently, non-orthogonal multiple access (NOMA) has been envisioned as one of the enabling technologies for achieving ultrahigh throughput and high efficiency in cellular networks in order to enhance the sensing and transmission performance of the IIoT. Spectrum deficit is also a limitation that will impact each sensor's quality of service (QoS) due to a large number of connected industrial sensors [2]. The automated communication system needs controlling, validation, and minimal latency rate for developing many wireless communication devices like actuators and sensors in Industry 4.0. Generally, real-time validation tasks are performed intensely, and the wireless communication system is presented in minimal size; simultaneously, they have only limited memory source, communication, and computation [3]. Therefore, it is essential for improving the validation and latency rate minimization ability in 5G applications. Computational job offloading near the MEC server reduces remote cloud-user data exchange [4]. The edge network's MEC servers analyse, offload, cache, and process user data. A big data platform requires cache storage, and server analytics software for efficient analysis but MEC has fewer sources than the distant cloud [5].

Machine learning approaches have several features in resolving complex and non-convex problems [6]. In machine learning approaches, enhancing reliability in communication is considered an arduous task. A distributed approach is developed based on federated learning, and it allows resource allocation to perform effectively in ultra-reliable vehicular communication systems [7]. Finally, the professionals have explained the complexity attained in reliable communication of the 6G network. Further, knowledge in the communication area is improved effectively with machine learning networks and cross-layer optimization techniques [8].

The IIoT can utilize the 5G spectrum in the future to attain significant bandwidth. Allocating the same spectrum resource to numerous users can help with spectrum utilization and successfully address the lack of spectrum, making NOMA a suitable solution for 5G. Orthogonal multiple access (OMA) and NOMA were compared in terms of performance and it found that NOMA outperformed OMA [9]. By incorporating NOMA into the IIoT, the IIoT boosts the total transmission capacity by using the limited resources to link more IIoT devices. NOMA's reliability and load balance strategy are still in the initial investigation. The reliability of NOMA has been studied and improved through time and spatial diversity-based retransmission schemes for industrial automation applications [10]. A novel task offloading strategy in NOMA enabled edge-computing framework is essential to tackle the challenges attained in the existing models.

Contributions associated with the developed task offloading in NOMA-EDGE-enabled IIoT are elaborated as follows.

• Initially, we designed NOMA-MEC enabled IIoT scenario in which each IIoT device can partially offload task to edge servers (ES) through NOMA. To study this problem, we formulate a multi-constraint optimization problem for offloading decision to minimize service delay, balance load, energy consumption, and maximize resource utilization.

• We develop an efficient heuristic model named HCMDO for optimizing the decision variable in computational offloading.

• We develop a new task and load prediction model named H-LSTM to predict best server for each task with parameter optimization based on the developed HCMDO for maximizing accuracy.

• An adaptive task offloading decision model was built based on DRL and prediction using H-LSTM.

• Finally, analyse the efficacy rate of the suggested model for NOMA-MEC-enabled IIoT over conventional approaches and classifiers under different settings and datasets.

The reminder of this paper is organized as follows. The related work survey and problem statement are presented in section 2. System model in an edge computing system with NOMA is discussed in section 3. Task offloading decision model for time-critical application using DRL and H-LSTM approach elaborated in section 4. The experimented results as well as the conclusion part of the research are presented in section 5 and section 6, respectively.

2. Related works

In [11] have recommended a reinforcement model named Q-Learning to perform effective resource allocation by reducing the interference and neglecting the usage of the network. The developed approach was presented in the form of a decentralized manner. Here, the cognitive radio (CR) worked as a multi-agent and generated a dynamic team for attaining the effective optimal resource allocation model. In [12] have suggested a joint optimization technique for validating resource allocation in Edge Servers. The workload of the smart terminal (ST) offloaded and radio source allocations for non-orthogonal multiple access (NOMA) transmission effectively reduced the cost of the system. Still, the

non-convexity issue attained in the optimization issue was exploited, and an effective layered approach was developed to reach the optimal solution. The outcome showcased that the developed model achieved a better effectiveness rate and gained more efficacy.

In [13] have recommended radio access network (RAN) slicing-based two-level approach in the open-RAN (O-RAN) framework for assigning the validation as well as communication in the RAN sources. In every slicing phase of RAN, the resource slicing issue was developed with the help of the Markov decision approach and learning approach in resolving the problems. In [14] have proposed deep reinforcement learning-based collaborative computation offloading and resource allocation scheme. Unmanned aerial vehicles (UAVs) are used in emergency scenarios with network failure. To re-establish the network by serving as airborne base stations and computing nodes for the edge network.

In [15] have combined dynamic channel access and power control in a wireless interference network using multi-agent DRL. The multi-agent DRL algorithm with centralized training (DRLCT) solved the joint resource allocation problem. In this instance, training is carried out at the central unit, and following training, users decide independently on their transmission tactics using just local data. In [16] have assisted downlink of NOMA network through reconfigurable intelligent surface by proposing a capacity maximization strategy based on a double deep Q-Network (DDQN) under energy consumption constraints. The reconfigurable intelligent surface phase shift design and the UAV trajectory are collaboratively optimized using the DDQN approach. The simulations show that the recommended algorithm convergence and the chosen environment can help the neural network learn in the intended direction and behave better.

In [17] have presented a revolutionary reverse auction-based computation offloading and resource allocation mechanism (RACORAM) for mobile cloud-edge computing. The essential concept is to accept offloaded computation from nearby mobile devices, which are resource-constrained, the cloud service centre recruits edge server owners to replace them. The reverse auction-based compute offloading and resource allocation challenge aims to reduce the cost. The reverse auction encourages edge server owners to participate in the offloading process.

In [18] have modelled mobile-x architecture for task offloading in MEC. The dynamic nature of the network results in an inefficient allocation of the edge servers. As a result of processing delays and time constraints, activities are abandoned. Because of the ambiguous load dynamic state across the edge nodes, the researchers find it challenging and confusing to decide whether to unload. The decision to choose edge nodes for centralized edge offloading is what poses the problem. The offloading choice problem is then resolved by in-depth network task flow analysis and device feedback on edge services. This approach combines bi-directional LSTM and deep reinforcement learning to improve system cost in terms of time delay and energy consumption.

To solve inappropriate compute offloading and uneven resource allocation in MEC [19] proposed a deep learning-based task offloading and resource allocation technique. First, the multiuser multi-server MEC environment's calculation and communication models are fused to generate a new objective function. This objective function reduces terminal device energy utilization and maximizes computing task completion time. Deep reinforcement learning based on multi-agent reinforcement learning creates system benefits and resource consumption as rewards and losses. The dueling-DQN algorithm determines the optimum resource allocation approach for the system issue model.

In [20] have developed a novel approach to offload IoT tasks in an edge-cloud environment, which uses the fuzzy logic method for analysing application characteristics, resource utilization, and resource heterogeneity. Additionally, it can lower the total job failure rate due to problems with the network and processing resources. A set of fuzzy rules akin to human thinking make up a fuzzy rules basis. It is a straightforward if-then rule that addresses all scenarios related to application attributes and system circumstances.

2.1 Problem statement

Edge computing resources are limited compared to the cloud sector. However, IIoT system comprises low latency and intensive computing resource application, which we term time-critical applications. Even though cloud has intense resource, it delivers high latency. The delay in offloading and processing such applications leads to a severe loss of money and human lives in the industry. Offloading computing tasks to MEC servers in the network's perimeter can effectively reduce delay. Hence, the time-critical application must be allocated to ES, which expects execution within the deadline. The characteristics of IIoT, such as heterogeneity, wireless network, real-time, and high data generation, affect the performance if resources are not properly scheduled and utilized. These may cause delays, energy and bandwidth wastage, failure, and performance degradation. Multi-agent model-free reinforcement

learning schemes effectively converge and enhance the network capacity. However, performing with the channel imperfection effects among the cooperative CR networks is unsuitable. The user cooperation approach ensures the active computing of user nodes by sharing helper nodes' computation and communication resources. NOMA has minimized the system cost, including computation resource consumption and overload. But, it makes convergence speed slower and provides unstable performance in the convergence process. O-RAN solves the slicing problem and provides high robustness and efficiency in satisfying the requirements of services. However, it does not investigate the relationship between the duration of the reconfiguration interval and retraining frequency.

Collaborative computation offloading and resource allocation (CCPRA) scheme based on DRL handles emergencies where network failure exists in UAV-assisted IoT networks. It shows high energy costs and a lack of overall performance. DRLCT provided the solution for the joint resource allocation problem, multi-agent at centralized training increases the overhead. In RACORAM, the cloud service center chooses the edge servers for offloading its computation from nearby resource-constraint MDs. Since the reverse auction is followed, it is unsuitable for delay-constrained applications. The Mobile-X architecture model's reinforcement learning technique addresses offloading concerns and provides a better decision-making process independent of the system cost. It slows down convergence speed and results in unstable convergence.

The dueling-DQN algorithm determines the best resource allocation strategy in multiuser and multi-server MCE environments. However, not adhering to task deadlines which is crucial in time-critical applications like real-time video processing. The fuzzy rule-based offloading approach works efficiently regarding service time and resource utilization. It's not suitable for resource-intensive and latency-sensitive applications. Therefore, complexities attained in the traditional system inspire us to develop novel hybrid LSTM and reinforcement learning approaches with the optimization strategy to solve the task offloading and resource allocation problem in Industrial IoT system.

3. System model

The spectrum efficiency advantage of NOMA is utilized in this work to jointly optimize the computing and communication resource for edge-enabled IIoT systems. The IIoT devices that are edge node (e.g., smart camera) in our proposed NOMA enabled edge computing architecture as displayed in Fig. 1. The considered system model comprises a multi-task, multi-access node for the sake of simplicity. However, in practice there are many IIoT devices with tasks to be executed locally or edge. Multiple tasks under a partial offloading strategy are challenging and worth investigating from an industry 4.0 compliance perspective. Each node generates computation-intensive and delay-sensitive tasks (e.g., smart camera-based alert system through object detection, human motion tracking). The applications can delegate their computation-related duties to neighbour edge servers with computing capabilities, ensuring low latency service.

The computational delay is minimized so edge nodes effectively utilize NOMA to perform offloading in the validation workload presented in the ES group form $I=\{1,2,...,Y\}$ through multiple accesses MEC. The gain of channel power consumption (f_i) is given for ES in Eq. (1).

$$f_1 > f_2 > \cdots > f_y \tag{1}$$

The gain of channel power consumption from the edge node to ES is presented as y. In an upcoming equation, the transmit power of the task node to ES y is shown as T_y. The throughput S_y attained from the edge node to ES y is attained in Eq. (2) based on the Successive Interference Cancellation (SIC) principle.

$$S_y = vlog_2\left(1 + \frac{f_y T_y}{f_y \sum_{z-1}^{y-1} T_z + V m_o}\right) \forall\, y \in Y \tag{2}$$

Here, the channel bandwidth of the edge node is presented as v, and the background noise spectral power density is given as m_o. The task offloading decision model design includes task analysis and joint optimization models.

3.1 Task analysis model

Each edge node N in the IIoT system runs a group of task denoted as $I= \{1, 2, 3..., L\}$. Each task generated in different time slot $P= \{1, 2,..., P_l\}$. The newly generated task $Q_N(t)$ of edge node N at time slot $t \in P$ is denoted as given in Eq. (3)

$$Q_N(t) = (T_{size}, T_{type}, T_d, C, \lambda) \tag{3}$$

Here, T_{size} denotes task data size, T_{type} represents task type that is determined based on the task deadline T_d, C denotes the computation demand of the newly generated task, λ denotes the task arrival rate.

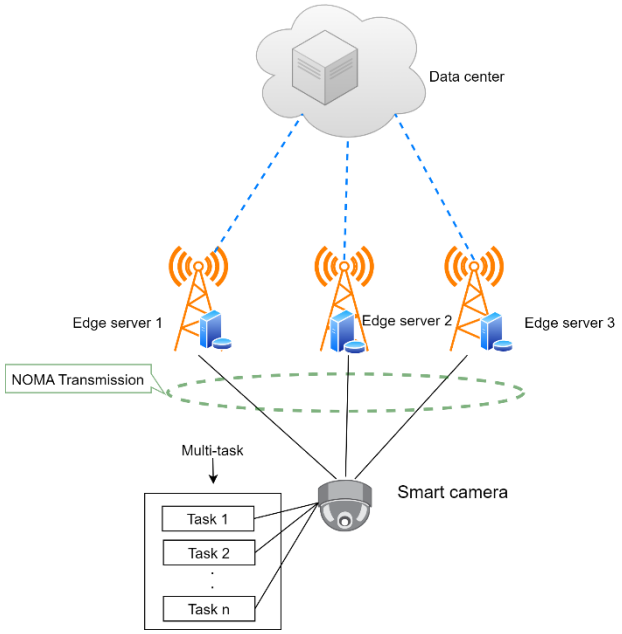

Figure. 1 NOMA-enabled Industrial edge computing

According to the application's tolerance for delays as determined by the task characteristics, the task is categorized as delay-constraint or delay-tolerant and resource sensitive.

The task categorization depends on the task's parameters and the user device's load profile. A task is categorized as delay-constraint if it requires immediate and precise results. The task is classified as delay-tolerant if not.

3.2 Joint optimization model

This model determines the target edge server for offloading tasks. Considering the local resource constraint, we use partial local execution in our work. We focus on the optimal execution of delay-constraint and computation-intensive tasks at edge servers. We formulate joint optimization of computation, communication, and cache resource in edge computing. The goal is to minimize delay, energy consumption and maximize resource utilization. This joint optimization problem results in improved quality of experience (QoE) as a whole.

Task nodes use bandwidth and cause transmission delays when they offload tasks to ES. Tasks in time slot t and bandwidth used during offloading must be less than the maximum bandwidth available to maintain a high transmission rate. In conclusion, the system's overall model trades off time delay and energy consumption during task computing to create a cost reduction problem. The solution aims to reduce the overall cost of the tasks produced by the system over time.

In the edge computation scenario, the task node sends the task across the shared wireless channel for ES to process. It incurs transmission delay and energy consumption depending on task size as specified in the communication cost. After receiving the computation tasks, the ES assigns a computing resource to each job. We refer to r_i, r, and R, respectively, as the computing resource that will be used to accomplish offloaded tasks, the computation resource allocation vector, and the total computation resource of the ES. As a result, the ES task calculation latency and power consumption are handled. We disregard the time and energy used by the ES in returning the calculation result to the task node in our model. The offloaded task at the edge server caches data from the cache storage.

4. Adaptive task offloading decision model

The proposed model is a hybrid of DRL and LSTM that has been heuristically adjusted. An H-LSTM network is introduced as the first layer to precisely capture the long-term historical relationships in the data. Our approach employs two layers of neural networks DRL and H-LSTM. The first H-LSTM network is used for task and load prediction at each time point t, and its outputs are all the ESs' anticipated states determined by historical data. An agent of DQN finds optimal action using the prediction result of H-LSTM. An agent then allocates the task to the ES. Following that, the chosen ES carry out the policy and provide their information to an agent and their present true states, which will be recorded in the historical data for future predictions. Each ES completes its execution and records the cost and resource utilization for further use in terms of rewards to an agent. We continue this procedure throughout subsequent intervals until the process converges and all jobs are allocated with the lowest possible cost. The workflow of the proposed work is as follows: 1) proposing HCMDO for dataset preparation by initializing and optimizing parameters, 2) developing H-LSTM for cost and load prediction, and 3) Deep reinforcement learning-based decision model.

4.1 Proposed HCMDO

A novel optimization approach named HCMDO is implemented for optimizing the decision user variable, decision server variable, decision communication and caching variable in the data augmentation phase, and bias, weights, hidden neuron count, and epochs count in LSTM to offer an effective prediction. Cat and mouse optimization

Algorithm 1: Developed HCMDO

Initiate the population for cat mouse and dingo	
Assign the parameter of both approaches	
For entire solution	
Validate the fitness of an entire solution	
Update the random number A by the newly developed concept provided in Eq. (4)	
	If ($A > 0.5$)
	Renew the solution using DOA
	Else
	Renew the solution using CMO
	End if
Find the optimal best solution	
End for	
Return the best solution	

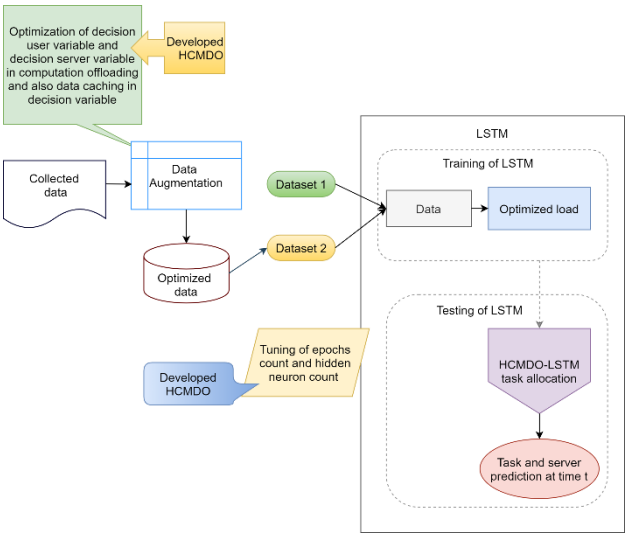

Figure. 2 Task and load prediction model

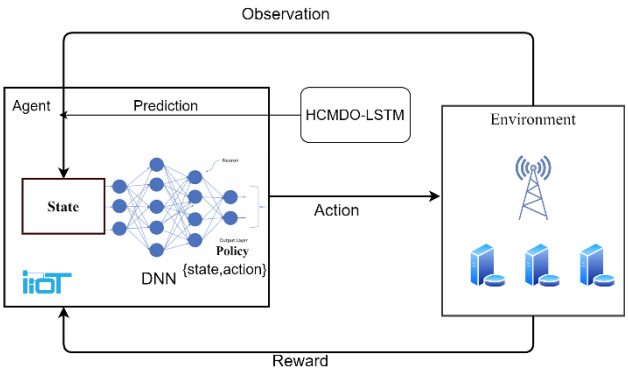

Figure. 3 Task offloading decision model

(CMO) is easy to implement and utilizes only a few variables to attain an effective outcome rate. Still, it mainly depends on the basic condition and consumes more time to implement. Therefore, the Dingo optimization algorithm (DOX) in the developed framework is needed, and this fused combination is termed HCMDO as presented in algorithm 1. DOX needs a minimum amount of mathematical effort and uses minimal validation time to attain a better optima value.

In this developed HCMDO, the random number A is updated based on a new concept conventionally used as the random parameter with the fixed range, presented in Eq. (4).

$$A = \left(\frac{wf-bf}{bf}\right) \tag{4}$$

Here, the term *wf* denotes the worst fit and *bf* represents the best fit.

4.3 Proposed H-LSTM prediction model

The architectural view of the developed task and load prediction is presented in Fig. 2. Initially, network parameters are attained from MEC sever. The task-related parameters are initialized and offered as the input to the dataset augmentation phase. Here, parameters in offered data like decision user variable, decision communication variable, and decision server variable in computation offloading and data caching in decision variable are tuned with the help of developed HCMDO and attained the outcome as optimal task data. The optimal task data is considered as dataset 2. Moreover, the data taken from online sources is presented as dataset 1, and further, both dataset 1 and dataset 2 are used in training the LSTM-based prediction phase. In the training phase of LSTM, targets are fixed as the optimal task to the MEC server from dataset 1 and dataset 2. From dataset 1, the task class is fixed as the target; from dataset 2, cost and load of MEC server is fixed as the target. The parameter of LSTM, like bias, weight, epoch count, and hidden neuron count, are tuned with the help of developed HCMDO for maximizing the accuracy rate to attain an effective cost and load prediction rate. In the testing phase, heuristically modified LSTM predicts task and load based on historical data.

4.4 Deep reinforcement decision model

A Deep reinforcement learning-based decision model is designed by integrating H-LSTM for attaining an effective task offloading in NOMA enabled edge computing under IIoT system. The architectural view of the proposed decision model is presented in Fig. 3.

LSTM output is given to the deep reinforcement learning based decision model for obtaining optimal task offloading policy. An agent in DRL interacts with the environment in our case ES by choosing action. The optimal action is picked with the help of ES state prediction based on cost and load. We considered simulated decision variables, benchmark data as features in training LSTM, the transmission and computation cost are taken in to account while predicting ES's state. The next task and its preference is predicted by H-LSTM based on the user history data. This justifies the reason for using two datasets for training LSTM in our proposed work. The prediction continues at every time point t until task queue of edge node becomes empty. An agent assigns task to optimally predicted ES for execution. The decision model is validated through reward function considering execution cost, resource utilization and service delay.

Deep reinforcement learning-based decision model describes a novel method for overcoming the problem of dynamically allocating the job and choosing optimal resource through efficient accuracy and convergence. Effective task allocation in IIoT increase productivity by ensuring effective decision-making, system efficiency, minimizing resource wastage, and lowering costs. The environmental information of DRL is explained below.

Agent: An agent is a software model runs in edge node. It is seen as a scheduler that chooses action in accordance with the state of the environment at the time and refines its decision-making by ongoing interaction with it. The agent's goal is to minimize overall system expenses by acting in the best way feasible in each circumstance.

Observation: The observation is the feedback given from the environment back to the agent. It helps an agent to decide what can be done in its next action. Most of all, the agent does not have a memory. So, its decision based on the observation of the current state. At every initial time, agent observes the state of an environment. A state $q \epsilon Q$ offers the status information that is displayed in Eq. (5).

$$Q = \{q \mid q = (R_c, O, I_\Psi, b_l, f_l, P_i) \quad (5)$$

In this, R_c describes task, O denotes the observation, I_ψ as the task information, b_l as tasks in the backlog, front log portions of the job queue as f_l, and P_i the prediction state information based on cost and load by H-LSTM.

Action: The first of the action's two goals is to specify how jobs should be scheduled. On the other hand, combining multiple functions into a single action might produce a large action space, making the problem excessively challenging. As a result, an innovative mechanism to reduce the action space is being created. The decision epoch and the actual timestamp are first distinguished. Each timestamp contains a number of decision epochs for carrying out actions. For every timestamp, the number of computing resources required is modified, and time is frozen to plan out each task in the backlog. An agent assigns every task to optimal predicted ES for execution through defined action space. Considering the state information, an agent in RL initially determines the task computation demand and deadline of the newly arrived task to decide whether task has to be executed locally or offload to ES. This decision denoted as offloading decision (O_d). Then, it chooses optimal ES called task allocation decision (E_s) based on predicted cost and load, finally it chooses NOMA transmission (N_t), as given in Eq. (6) as action

$$B = \{b \mid b = (O_d, E_s, N_t)\} \quad (6)$$

Reward function: the reward from an environment is a significant factor in validating developed framework. The policy network is updated such that an optimal choice made in the next time slot when the agent observes the state at time slot *t*, makes an action in accordance with the policy, and then receives a reward at time slot *t+1*. Each agent seeks to maximise its long-term discounted reward by enhancing the mapping from states to actions, which encourages the agent to consistently choose the best course of action in its ongoing interactions with the environment. Here reward function is defined as state and action at time *t* and reward *R* for the pair (q_t, b_t) given as below.

$$R = \sum_{t=0}^{T} \gamma^t r^t \quad (7)$$

Here γ^t represents discount rate and r^t denotes reward at time *t*. The objective function of the system model is achieved on the basis of reward function design. In common reward function defined in a way to minimize overall cost of processing delay and energy consumption by selecting the best offloading options. In our work, strict deadline is followed as we considered time critical IIoT task. Regardless of local or edge execution, energy consumption and delay defined as key attributes in reward function design. Finally, the offloading result (success/failure) is considered to evaluate the reliability of selected ES. We employ negative incentives in order to adhere to the goal of the paper's model. The DRL can receive

Algorithm 2: Developed reinforcement learning-based decision model

Input: task requests (TR)
Result: optimal task offloading policy
Initialize network *Q* with random weight w_i
Initialize replay memory *D* to the capacity *C*
for episode = 1 to R do
receive initial state observation q
for t = 1 to N do
explore an action b^t with probability 1-epsilon based on HCMDO-LSTM prediction
Or else exploit action b^t =argmaxQ(q^t,b^t,weight)
run action b^t
get reward r^t for chosen action b^t and next State q^{t+1}
cache (q^t, b^t, r^t, q^{t+1}) into D
Chose samples from D randomly
Update the weights of DNN for loss minimization through stochastic gradient descent
Policy update $\pi(q^t)$ after every steps
end for
end for

the highest reward while the system objective function is at the lowest possible level.

Policy: a policy *(π)* is mapping function that an agent uses to select action b at state q as $\pi: q \rightarrow b$. The policy map gives probability *(Pr)* as given in Eq. (8).

$$\pi(q, b) = Pr\ (q_t = q, b_t = b) \qquad (8)$$

The developed deep reinforcement learning-based decision model combines DQN and LSTM network to solve exploration and exploitation problem. The exploration phase selects actions based on predicted probability of LSTM network. In exploitation phase, actions are selected based on the derived policy. The trade-off between exploration and exploitation can be handled using LSTM network integrated in DQN as given in Algorithm 2.

In this paper, we evaluated the performance of proposed DRL-HLSTM by comparing four variants of similar learning methods such as RL [21], DRL-LSTM [22], DQN [23] and DDQN [24].

Reinforcement learning: Initially, we build basic RL using Q-learning algorithm. RL agent learns to act and adopts changes in an environment. With an objective of deriving optimal offloading policy, Q-learning takes decision and receives reward from an environment. It validates decision by reward analysis to update policy, which maximize the reward in future. RL agent develops policy π by learning state action pair at time t. immediate reward for action at time t is r. In our work, we receives tasks from edge node and RL agent map it to ES. The goal of the RL agent is to discover an offloading policy that minimises the cumulative reward value (i.e., cost) for all the considered ES and tasks. Each state and action pair is stored in Q-table. Nevertheless, when state and action space is large it is difficult to store and process (state, action) pairs in Q-table. Q-learning took more time to converge and sometimes not converges at all. It takes RL approach lower in all evaluation metrics.

Deep reinforcement learning: DRL combines RL and deep learning algorithm to maximize the total discounted reward. We use Deep quality network (DQN) as DRL, which optimize policy π^* by maximize the future reward in the long run, rather than the immediate next reward. Unlike RL, DQN estimate Q-value instead of computing Q-value for each state-action pair *(q,b).* It is essential for modelling large-scale scheduling scenarios with a large number of actions-state pairings. In our training procedure, the DRL agent selects a random scheduling action (i.e., assigning tasks to ES) with a high probability in order to investigate the influence of unknown scheduling alternatives and develop a more effective strategy. Using the Bellman equation, the agent increases the probability of selecting the action with the highest Q-value during training in order to minimise the expected cumulative reward (execution cost). Technically, the agent schedules one or more pending tasks at each time instant t based on the conditions specified. The optimal Q-value function indicates that, at time t, each policy chooses a valid ES to execute each task in order to minimise the total execution cost. We obtain the actual Q-value of action b by using the state q as input to the online network and q as input to the target network in order to determine the minimum Q-value of all actions in the target network.

DDQN: Double DQN is proposed to address the issue of overestimation. DQN takes the maximum value with max each time, and the difference between this maximum value and the weighted average value introduces an error, which leads to overestimation after a lengthy period of time accumulation. The Double DQN consists of two networks, A and B, and utilises these two networks to sequentially process the state evaluation and action output. Thus, one network is used to select the action, and the other network is used to alter the Q value based on the action selected.

DRL-LSTM: In this LSTM is integrated with DQN for model stability and fast convergence. However, the performance of classifier is highly

Table 1 Simulation parameter for training LSTM

Parameter	Range	Description	Unit
C	4-10	MEC count	-
K	100-400	number of task nodes	-
da	12777	mobile network operator	-
r	1000	cluster	-
PK	27	transmission power	watts
BM	[25,32]	channel bandwidth	MHz 5
Σ	[50,20]	transmission speed	Mbps
V	0.02-12	Transmission duration	sec
MME	100-300	computing memory	TB
$MCPU$	[2,2.5]	computation in CPU	GHz
$Rhok$	0.02-12	Processing time	sec
ZK	[452.5,732.5]	computational workload	cycles/bit
T_d	0.02-12	task deadline	Seconds
T_{size}	[2,7]	task size	MB

dependent on their hyper parameter values; therefore, it is essential to employ a method that ensures the optimal values. To address this problem, we propose heuristic hyper tuning approach through hybrid optimization algorithm termed HCMDO for LSTM. It has a faster convergence rate and can improve the prediction accuracy of the LSTM model effectively. Therefore, state prediction model is proposed based on HCMDO combined with long-short-term memory (LSTM) neural networks, for higher reliability. Due to the agent's initial random selection action, it takes longer to investigate and select the optimal result during training on the DRL task offloading decision model. In this paper, we propose predicting the cost and server burden based on the record history of the periphery server. Based on the results of the prediction, the optimal server is selected with a certain probability. This solution enables the agent to avoid selecting servers with high load and suggest low cost, thereby decreasing task processing latency and task abandoning rates.

5. Results and discussion

5.1 Experimental setup

To validate the proposed algorithm, we did simulation in the Python platform. The proposed work experiment carried in three stages. Namely dataset selection and preparation, H-LSTM training and DRL integration with H-LSTM. In this proposed approach, two different datasets are utilized. In dataset 1, the resource and iot data are collected from Google cluster trace. The data such as time, constraints, priority, instance event type, cluster, sample rate, memory access per information, end time, average usage, random sample usage, collection name, collection logical name, and assigned memory, vertical scaling, and so on are utilized for the analysis. In dataset 2, the simulated dataset is utilized to perform effective resource prediction analysis. Initially, the data are offered as the input to the augmented data phase, and the variables like decision user variable, decision server variable, communication variables, and data transmission decision variable are optimized by developed HCMDO. Then, optimal predicted data are attained and further offered to the prediction phase. The dataset utilized different parameters for the analysis, which are discussed in Table 1. Thus, the attained data from the dataset are offered as the input to the training phase of the LSTM model.

In existing studies, only google cluster dataset is considered for training, but it is insufficient to depict real scenarios. We initialized computation and communication parameters to improve prediction stability and accuracy to perform an effective offloading in an edge computing system. Those parameters are utilized as the input to the dataset augmentation phase. Optimal predicted task data are attained as the output. Further, they are offered as the input to train LSTM.

We used the simulated and online dataset to train LSTM network. The extracted features are normalized according to the proposed model. We employ the HCMDO algorithm to tune hidden neurons and epoch count of LSTM. The proposed H-LSTM prediction method undergone the following phases: data pre-processing, model training, and model prediction. In training LSTM, the error between output and real value is continuously reduced. The LSTM unit can store long-term information and is suitable for long-term training. The prediction outcomes of H-LSTM is given to the deep reinforcement learning-based decision model for obtaining optimal task offloading policy.

Finally, we integrate DRL and H-LSTM. Due to the agent's initial random selection action, it takes longer to investigate and select the superior result during training on the DRL decision model. We propose in this paper to forecast cost and server traffic based on peripheral server record history data. Based on the results of the prognosis, the server is

selected with a certain probability as the optimal resource for the subsequent instant. This solution permits the agent to effectively avoid selecting servers with high load and cost, thereby decreasing task processing latency and energy consumption. In this paper, the performance of the proposed work is compared with existing LSTM-DRL, RL, DQN, DDQN. We also compare the performance of proposed H-LSTM prediction accuracy with LSTM As we use high number of training instances (simulated and benchmark) H-LSTM & DRL able to make accurate predictions about the burden, cost and based on the prediction outcomes, the agent can make optimal use of the most thoroughly investigated strategy. By utilizing H-LSTM, we achieve promising result on average latency, energy consumption, and number of input task, overall cost. Through the decision model, newly generated tasks assigns with the optimal ES. The LSTM prediction model predicts the optimal server for each task in advance based on historical data and then feeds that prediction into the decision model to propose an offloading scheme. In the experiments, we used 400 nodes and 10 ES as maximum. In each iteration, task and ES are randomly selected between 100 to 400 and 4 to 10 respectively.

5.2 Performance comparison

The prediction accuracy and error analysis of the proposed H-LSTM is given in Table 2. The proposed H-LSTM is superior in terms both accuracy and error than the traditional LSTM. The training and testing accuracy of the proposed H-LSTM is depicted in fig. 4.

The Performance analysis of the developed adaptive task offloading decision model shows higher outcomes in terms of cost function, average latency, energy consumption, and number of input task as presented in Figs. 5-8. Average energy consumption analysis performed in the developed model attained 19.8%, 15.1%, 16.9%, and 15.6% better than conventional approaches like RL, LSTM-DRL, DQN, and Double-DQN respectively. Thus, the analysis in the suggested model achieved a better performance rate than the traditional approaches and offered a better task allocation rate in an industrial edge computing system.

6. Conclusion

In order to effectively handle different IIoT application requirement, heuristically modified LSTM based deep reinforcement approach was developed for task offloading in industrial edge computing systems. Data were initially provided to the dataset augmentation step. Here, the many data-presented factors are adjusted with the aid of a designed HCMDO to achieve ideally allocated jobs. Additionally, the obtained data and the dataset's data are provided to the LSTM prediction phase, which resulted in the identification of the best resource.

Table 2. Prediction and error analysis

No. of Nodes	Accuracy		Mean absolute percentage error	
	H-LSTM	LSTM	H-LSTM	LSTM
100	89.2	84.3	15.3652	19.3159
200	92.1	88.4	9.2985	15.3648
300	93.2	90.3	7.2468	11.6548
400	95.4	91.2	3.7892	8.3654

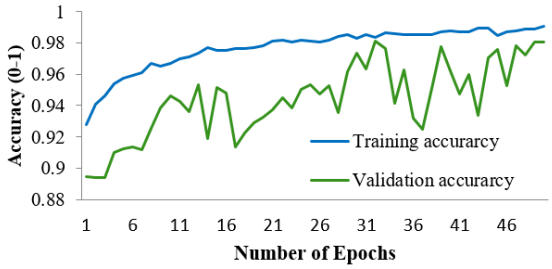

Figure. 4 Training and validation accuracy of H-LSTM

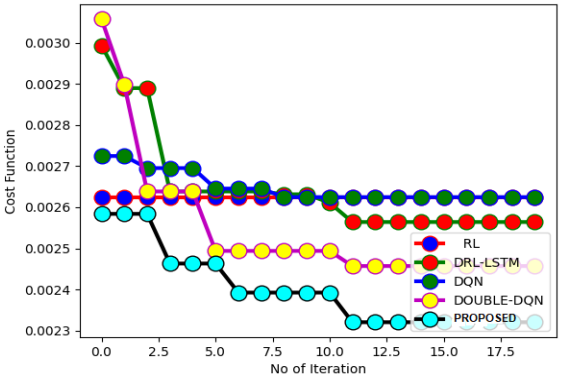

Figure. 5 Cost function

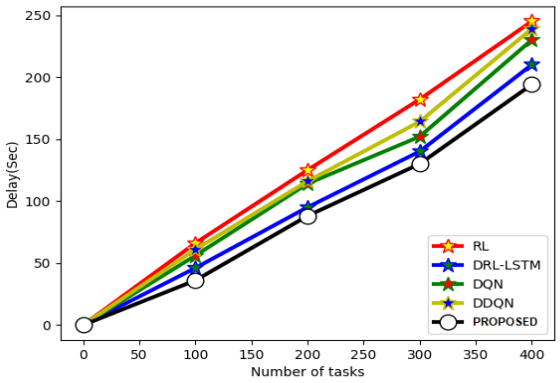

Figure. 6 Average delay

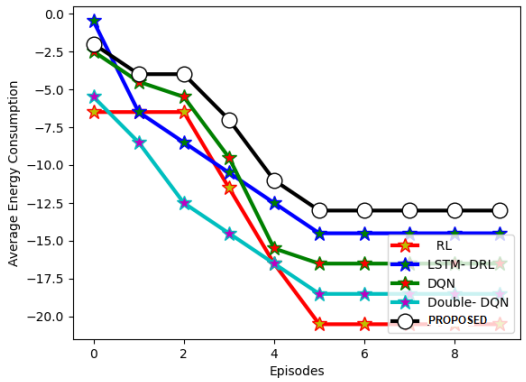

Figure. 7 Average energy consumption

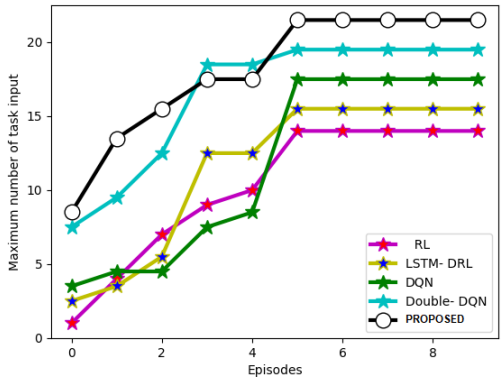

Figure. 8 The maximum number of task input

Here, the created HCMDO used to fine-tune the LSTM's parameters in order to enhance accuracy. The adaptive task offloading decision model based on deep reinforcement learning make use of the projected output for the task allocation. Comparing the recommended model to existing methods like RL, LSTM-DRL, DQN, and Double-DQN, the results are 19.8%, 15.1%, 16.9%, and 15.6% better. As a result, the suggested model's analysis outperformed more than traditional approaches in terms of performance and provided a superior result for systems utilizing industrial edge computing. Future study will take into account a variety of industrial IoT application scenarios in an effort to improve the performance of the suggested algorithm.

Conflict of interest:

The authors declare no conflict of interest.

Authorship contributions:

Conceptualization, K.Udayakumar and S.Ramamoorthy; methodology, formal analysis, Writing-original draft preparation, K.Udayakumar; writing-review and editing, supervision, S.Ramamoorthy.

References

[1] S. Nath and J. Wu, "Deep reinforcement learning for dynamic computation offloading and resource allocation in cache-assisted mobile edge computing systems", *Intelligent and Converged Networks*, Vol. 1, No. 2, pp. 181-198, 2020.

[2] L. P. Qian, B. Shi, Y. Wu, B. Sun, and D. H. K. Tsang, "NOMA-Enabled Mobile Edge Computing for Internet of Things via Joint Communication and Computation Resource Allocations", *IEEE Internet of Things Journal*, Vol. 7, No. 1, pp. 718-733, Jan. 2020.

[3] L. Wang, L. Jiao, J. Li, J. Gedeon, and M. Mühlhäuser, "MOERA: Mobility-Agnostic Online Resource Allocation for Edge Computing", *IEEE Transactions on Mobile Computing*, Vol. 18, No. 8, pp. 1843-1856, 2019.

[4] W. Na, S. Jang, Y. Lee, L. Park, N. N. Dao, and S. Cho, "Frequency Resource Allocation and Interference Management in Mobile Edge Computing for an Internet of Things System", *IEEE Internet of Things Journal*, Vol. 6, No. 3, pp. 4910-4920, 2019.

[5] E. Šlapak, J. Gazda, W. Guo, T. Maksymyuk, and M. Dohler, "Cost-Effective Resource Allocation for Multitier Mobile Edge Computing in 5G Mobile Networks", *IEEE Access*, Vol. 9, pp. 28658-28672, 2021.

[6] X. Chen, Z. Liu, Y. Chen, and Z. Li, "Mobile Edge Computing Based Task Offloading and Resource Allocation in 5G Ultra-Dense Networks", *IEEE Access*, Vol. 7, pp. 184172-184182, 2019.

[7] N. Kiran, C. Pan, S. Wang, and C. Yin, "Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks", *Journal of Communications and Networks*, Vol. 22, No. 1, pp. 1-11, Feb. 2020.

[8] M. Afrin, J. Jin, A. Rahman, A. Gasparri, Y. C. Tian, and A. Kulkarni, "Robotic Edge Resource Allocation for Agricultural Cyber-Physical System", *IEEE Transactions on Network Science and Engineering*, Vol. 9, No. 6, pp. 3979-3990, 2022.

[9] X. Liu and X. Zhang, "NOMA-Based Resource Allocation for Cluster-Based Cognitive Industrial Internet of Things", *IEEE Transactions on Industrial Informatics*, Vol. 16, No. 8, pp. 5379-5388, 2020.

[10] E. Iradier et al., "Analysis of NOMA-Based Retransmission Schemes for Factory

Automation Applications", *IEEE Access*, Vol. 9, pp. 29541-29554, 2021.

[11] A. Kaur and K. Kumar, "Energy-Efficient Resource Allocation in Cognitive Radio Networks Under Cooperative Multi-Agent Model-Free Reinforcement Learning Schemes", *IEEE Transactions on Network and Service Management*, Vol. 17, No. 3, pp. 1337-1348, 2020.

[12] L. Qian, Y. Wu, F. Jiang, N. Yu, W. Lu, and B. Lin, "NOMA Assisted Multi-Task Multi-Access Mobile Edge Computing via Deep Reinforcement Learning for Industrial Internet of Things", *IEEE Transactions on Industrial Informatics*, Vol. 17, No. 8, pp. 5688-5698, 2021.

[13] Filali, Abderrahime, B. Nour, S. Cherkaoui, and A. Kobbane, "Communication and computation O-RAN resource slicing for URLLC services using deep reinforcement learning", *IEEE Communications Standards*, Vol. 7, No. 1, pp. 66-73, 2023.

[14] A. M. Seid, G. O. Boateng, S. Anokye, T. Kwantwi, G. Sun, and G. Liu, "Collaborative Computation Offloading and Resource Allocation in Multi-UAV-Assisted IoT Networks: A Deep Reinforcement Learning Approach", *IEEE Internet of Things Journal*, Vol. 8, No. 15, pp. 12203-12218, 2021.

[15] Z. Lu, C. Zhong, and M. C. Gursoy, "Dynamic channel access and power control in wireless interference networks via multi-agent deep reinforcement learning", *IEEE Transactions on Vehicular Technology*, Vol.71, No. 2, pp. 1588-1601, 2021.

[16] H. Zhang, M. Huang, H. Zhou, X. Wang, N. Wang, and K. Long, "Capacity Maximization in RIS-UAV Networks: A DDQN-Based Trajectory and Phase Shift Optimization Approach", *IEEE Transactions on Wireless Communications*, Vol. 22, No. 4, pp. 2583-2591, 2023.

[17] H. Zhou, T. Wu, X. Chen, S. He, D. Guo, and J. Wu, "Reverse Auction-Based Computation Offloading and Resource Allocation in Mobile Cloud-Edge Computing", *IEEE Transactions on Mobile Computing*, 2022.

[18] G. Pandiyan and E. Sasikala, "Modelling mobile-x architecture for offloading in mobile edge computing", *Intelligent Automation & Soft Computing*, Vol. 36, No.1, pp. 617–632, 2023.

[19] Yu, Zijia, X. Xu, and W. Zhou, "Task Offloading and Resource Allocation Strategy Based on Deep Learning for Mobile Edge Computing", *Computational Intelligence and Neuroscience*, 2022.

[20] J. Almutairi and M. Aldossary, "A novel approach for IoT tasks offloading in edge-cloud environments", *Journal of Cloud Computing*, Vol. 10, No. 28, pp. 1-19, 2021.

[21] S. Xu et al., "RJCC: Reinforcement-Learning-Based Joint Communicational-and-Computational Resource Allocation Mechanism for Smart City IoT", *IEEE Internet of Things Journal*, Vol. 7, No. 9, pp. 8059-8076, 2020.

[22] Y. Tu, H. Chen, L. Yan, and X. Zhou, "Task Offloading Based on LSTM Prediction and Deep Reinforcement Learning for Efficient Edge Computing in IoT", *Future Internet*, Vol. 14, No. 2, p. 30, 2022.

[23] X. Deng, J. Yin, P. Guan, N. N. Xiong, L. Zhang, and S. Mumtaz, "Intelligent Delay-Aware Partial Computing Task Offloading for Multiuser Industrial Internet of Things Through Edge Computing", *IEEE Internet of Things Journal*, Vol. 10, No. 4, pp. 2954-2966, 15 Feb.15, 2023, doi: 10.1109/JIOT.2021.3123406.

[24] W. Cheng, X. Liu, X. Wang, and G. Nie, "Task Offloading and Resource Allocation for Industrial Internet of Things: A Double-Dueling Deep Q-Network Approach", *IEEE Access*, Vol. 10, pp. 103111-103120, 2022.