iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: https://doi.org/10.22214/ijraset.2026.79008

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

TOMEC-RAS-Net: A Lightweight Deep Reinforcement Learning Framework for Adaptive Task Offloading in Mobile Edge Computing

Durgesh Sharma[1], Sushil Kumar Sharma[2]
[1]*Research Scholar, Department of Computer Science and Engineering, Institute of Technology and Management Aligarh University: AKTU Lucknow*
[2]*Assistant Professor, Department of Computer Science and Engineering, Institute of Technology and Management Aligarh University: AKTU Lucknow*

Abstract: ***Mobile Edge Computing (MEC) enables resource-constrained devices to transfer computationally heavy tasks to nearby edge servers, reducing latency and energy consumption while maintaining quality of service. However, unstable network conditions, limited edge resources, and complex decision-making requirements often lead to task backlogs and degraded performance. This paper presents TOMEC-RAS-Net, a lightweight deep reinforcement learning (DRL) framework for real-time adaptive task offloading in dynamic MEC environments. The proposed framework integrates density-based spatial clustering (DBSCAN) for intelligent task grouping, an ensemble RAS-Net actor architecture combining ResNet, AlexNet, and ShuffleNet for robust feature extraction, and a priority-weighted resampling strategy to enhance learning efficiency under varying network conditions. A Markov Decision Process (MDP) formulation jointly optimises binary offloading decisions, CPU frequency allocation, and transmission power under queue stability and average power constraints. Experimental evaluation on an NVIDIA RTX 4090 platform demonstrates that TOMEC-RAS-Net achieves 98.74% accuracy, 98.92% precision, 98.65% sensitivity, 99.01% specificity, and a Matthews Correlation Coefficient (MCC) of 97.46% — outperforming DNN, CNN, ResNet, and LSTM baselines across all metrics. The model maintains low false positive (0.99%) and false negative (1.35%) rates and exhibits stable convergence over five-fold cross-validation, confirming its suitability for real-time MEC task offloading at the network edge.***

Keywords: Mobile Edge Computing, Task Offloading, Deep Reinforcement Learning, DBSCAN Clustering, Ensemble Learning, RAS-Net, Lightweight Model, Resource Allocation

I. INTRODUCTION

The rapid proliferation of mobile devices, Internet of Things (IoT) applications, and data-intensive services has dramatically increased computational demand at the network edge. Conventional centralised cloud computing architectures struggle with high latency, network congestion, and limited bandwidth — characteristics that are incompatible with latency-sensitive applications such as augmented reality, real-time video analytics, autonomous vehicles, and smart healthcare monitoring [1][2].

Mobile Edge Computing (MEC) addresses these limitations by deploying edge servers at base stations or access points near end users, creating a hierarchical architecture that minimises communication distance. In a standard MEC deployment, mobile devices offload computationally intensive tasks to edge servers via wireless channels, with optional cloud support for resource overflow [3][4]. Despite these advantages, MEC introduces new challenges: edge servers have limited CPU and memory relative to cloud data centres, network conditions fluctuate due to user mobility and channel fading, and real-time decisions must be made under strict latency and energy constraints.

Task offloading — the process of delegating computation from resource-constrained mobile devices to edge servers — is central to MEC efficiency. Offloading decisions must balance execution delay, energy consumption, and server load in real time [5]. Classical optimisation approaches (heuristics, convex optimisation, rule-based scheduling) rely on static environment assumptions and struggle in highly dynamic MEC networks [6]. Deep Reinforcement Learning (DRL) overcomes these limitations by learning optimal policies through environment interaction, but conventional DRL models are computationally expensive and memory-intensive — impractical for resource-constrained edge deployments [7].

This paper proposes TOMEC-RAS-Net, a lightweight ensemble DRL framework that jointly addresses task clustering, exploration efficiency, and convergence stability for MEC task offloading. The key contributions are: (i) a DBSCAN-based task clustering mechanism that reduces queue backlog and decision redundancy; (ii) an ensemble RAS-Net actor architecture combining complementary CNN backbones for robust and lightweight feature learning; (iii) a priority-weighted resampling strategy that improves adaptability under dynamic conditions; and (iv) a unified MDP-based optimisation formulation jointly handling queue stability, energy management, and offloading decisions.

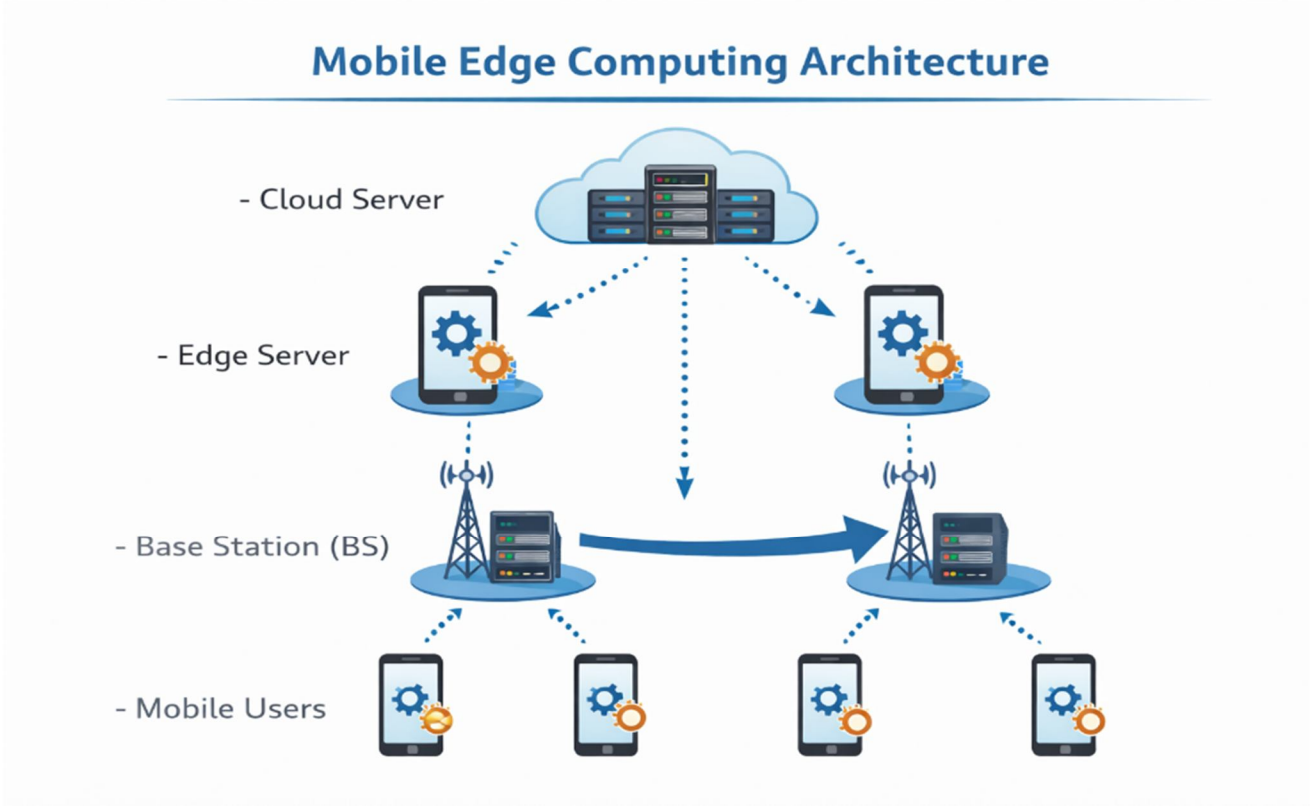


Figure 1. MEC Architecture: Cloud Server, Edge Server, Base Station, and Mobile Users

II. RELATED WORK

Task offloading and resource management in MEC have been extensively studied. We review four major research threads: MEC resource management, task offloading strategies, dynamic resource allocation, and energy-efficient offloading.

In resource management, Wang et al. [8] proposed a DRL-based adaptive system achieving high adaptability under varying workloads but requiring significant computational resources incompatible with edge deployment. Ma et al. [9] introduced a dynamic neural network framework for 6G MEC networks capable of handling complex demands but incurring high inference overhead. Duan et al. [10] focused on vehicular edge computing, delivering low-latency services but with limited generalisation to heterogeneous MEC environments.

For task offloading, Chen et al. [11] developed an energy-aware dynamic offloading framework using hybrid energy sources, though its complexity restricts real-time scalability. Yang et al. [12] proposed deep learning-based dynamic offloading with real-time decision capability but significant training overhead. Yuan et al. [13] extended MEC coverage via layered UAVs but introduced complex multi-layer coordination challenges.

Regarding dynamic resource allocation, Seid et al. [14] employed multi-agent DRL for UAV-enabled IoT edge networks, achieving adaptive coordination but facing scalability limits as device count increases. Zhao et al. [15] combined energy harvesting with dynamic scheduling to reduce task execution delay, though performance depends on accurate energy predictions.

For energy-efficient offloading, Qin et al. [16] proposed learning-based frameworks for vehicular networks, while Bolourian and Shah-Mansouri [17] introduced three-tier wireless-powered MEC architectures. Both approaches offer energy benefits but rely on predictable energy availability and are sensitive to system parameter tuning.

A common theme across all these studies is the trade-off between decision quality and computational overhead. None addresses the specific combination of unsymmetrical task clustering, ensemble lightweight actors, and priority resampling in a unified MDP formulation for heterogeneous, dynamic MEC environments — the gap addressed by TOMEC-RAS-Net.

Table 1. Summary of Related Work: Techniques, Advantages, and Limitations

Author & Year	Technique	Advantage	Limitation
Duan et al., 2021	Vehicular edge resource management	Low-latency for connected vehicles	Limited to vehicular scenarios
Wang et al., 2022	DRL-based adaptive resource mgmt.	Strong adaptability under varying workloads	High computational/training requirements
Chen et al., 2022	Dynamic task offloading with hybrid energy	Energy-aware for dynamic environments	Complex for large-scale real-time networks
Seid et al., 2021	Multi-agent DRL for UAV IoT edge	Adaptive UAV coordination, low latency	Scalability issues with increasing UAVs
Ma et al., 2023	Dynamic NN-based resource mgmt. for 6G	Handles fluctuating workloads/demands	High inference cost on edge devices
Qin et al., 2022	Learning-based energy-efficient offloading	Minimized energy; low-latency execution	Reduced efficacy under dynamic traffic

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. MEC Network Architecture

The system consists of *m* wireless devices (WDs) that transfer their computational workloads to an edge server (ES) over successive time intervals of equal duration, denoted by *S*, as illustrated in Fig. 1. In each interval, the queue associated with the *j*-th device receives incoming data represented by B_s^j, where the arrival process is characterized by a finite second-order moment. Specifically, B_s^j is assumed to follow an independent and identically distributed (i.i.d.) process with $F[(B_s^j)^2] = \eta_j < \infty$.

The channel gain between the j-th wireless device and the edge server in the s-th time slot is denoted as g_s^j. Under the block fading assumption, this gain remains unchanged within a given time slot, while varying independently across different time slots.

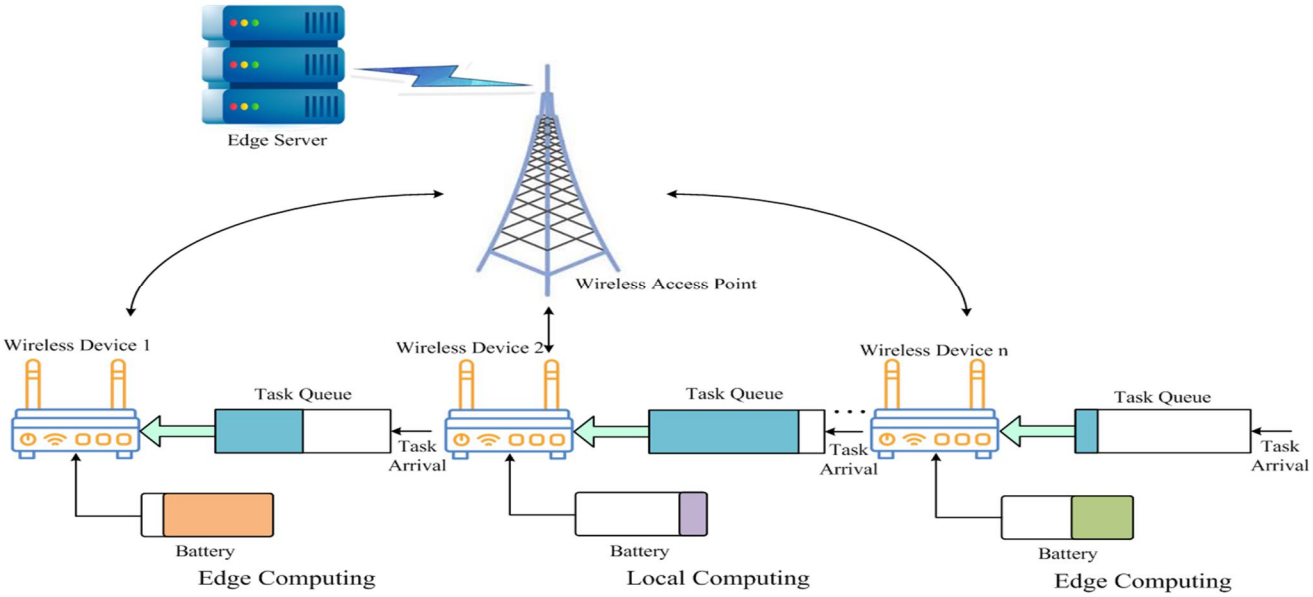


Figure 2. MEC Network Architecture: Multiple Wireless Devices Processing Tasks within a Time Frame

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

B. Computation and Energy Models

In the case of local processing, the amount of data (in bits) handled by the *j*-th wireless device during the *s*-th time slot is denoted as $C_s^{j,K}$, and the energy consumption $F_s^{j,K}$ are given in equation (3.1) and (3.2)

$$C_s^{j,K} = e_s^j S / \phi \tag{3.1}$$

$$F_s^{j,K} = l\left(e_s^j\right)^3 S \tag{3.2}$$

In this context, e_s^j denotes the CPU operating frequency of the *j*-th wireless device during local computation, constrained by an upper limit $e_{\max}^j$. The parameter $\phi > 0$ indicates the number of CPU cycles needed to process a single bit of data, while $l > 0$ represents the computational efficiency factor that captures the energy cost per CPU cycle. These expressions indicate that higher CPU frequencies increase both processing speed and energy consumption. Figure 3.1 illustrates the MEC network architecture, where multiple wireless devices (WDs) generate and process computational tasks.

C. Queue Dynamics and Stability

At the beginning of the *s*-th time interval, the data queue size of the *j*-th wireless device is denoted by $P_j(s)$, and its temporal behavior is governed by Equation (3.7).

$$P_j(s+1) = max\{P_j(s) - C_s^j + B_s^j, 0\}, \quad j = 1,2,\dots,M \tag{3.7}$$

In this context, B_s^j denotes the volume of newly arriving data (in bits) during the *s*-th time slot, while C_s^j corresponds to the amount of data processed, either through local computation or by offloading to the edge server (ES).

D. Optimisation Objective

The optimization aims to maximize the long-term average weighted computation rate across all wireless devices while maintaining queue stability and adhering to average power constraints. A key challenge is determining optimal task offloading and resource allocation decisions in each time slot without access to future channel conditions or incoming data information.

Let $Y = \{y_s\}_{s=1}^L$, $\tau = \{\tau_s\}_{s=1}^L, e = \{e_s\}_{s=1}^L, F_O = \left\{F_s^{j,O}\right\}_{s=1}^L$ represents the offloading time fractions local CPU frequencies and offloaded energy allocations over *L* time frames respectively. The optimization task can be formulated as a multistage stochastic mixed-integer nonlinear programming (MINLP) problem, as presented in Equation (3.11)

$$\max_{Y,\tau,e,F_O} \frac{1}{L}\sum_{s=1}^{L}\sum_{j=1}^{M} d_j F\left[r_s^j\right] \tag{3.11}.$$

Table 2. Problem Statement: Existing Issues, Impacts, and Proposed Enhancements

Aspect	Existing Issue	Impact	Proposed Enhancement
Task Offloading	Static/rule-based approaches	Suboptimal performance	Adaptive DRL-based model
Dynamic Network	Fluctuating bandwidth and latency	Increased delay	Real-time intelligent decision-making
Conventional DRL	High complexity and resource usage	Difficult edge deployment	Lightweight DRL framework
Energy Constraints	High computation energy consumption	Battery drain	Energy-efficient optimization
Multi-user Environment	Resource competition	Scalability issues	Efficient resource allocation strategy

IV. PROPOSED TOMEC-RAS-NET FRAMEWORK

A. Architecture Overview

TOMEC-RAS-Net integrates four cooperative modules: (1) DBSCAN-based task clustering for workload grouping, (2) ensemble RAS-Net actor for robust offloading decisions, (3) a critic module for utility-maximising action selection, and (4) a priority-weighted policy update module with experience replay. Figure 3 shows the RAS-Net architecture.

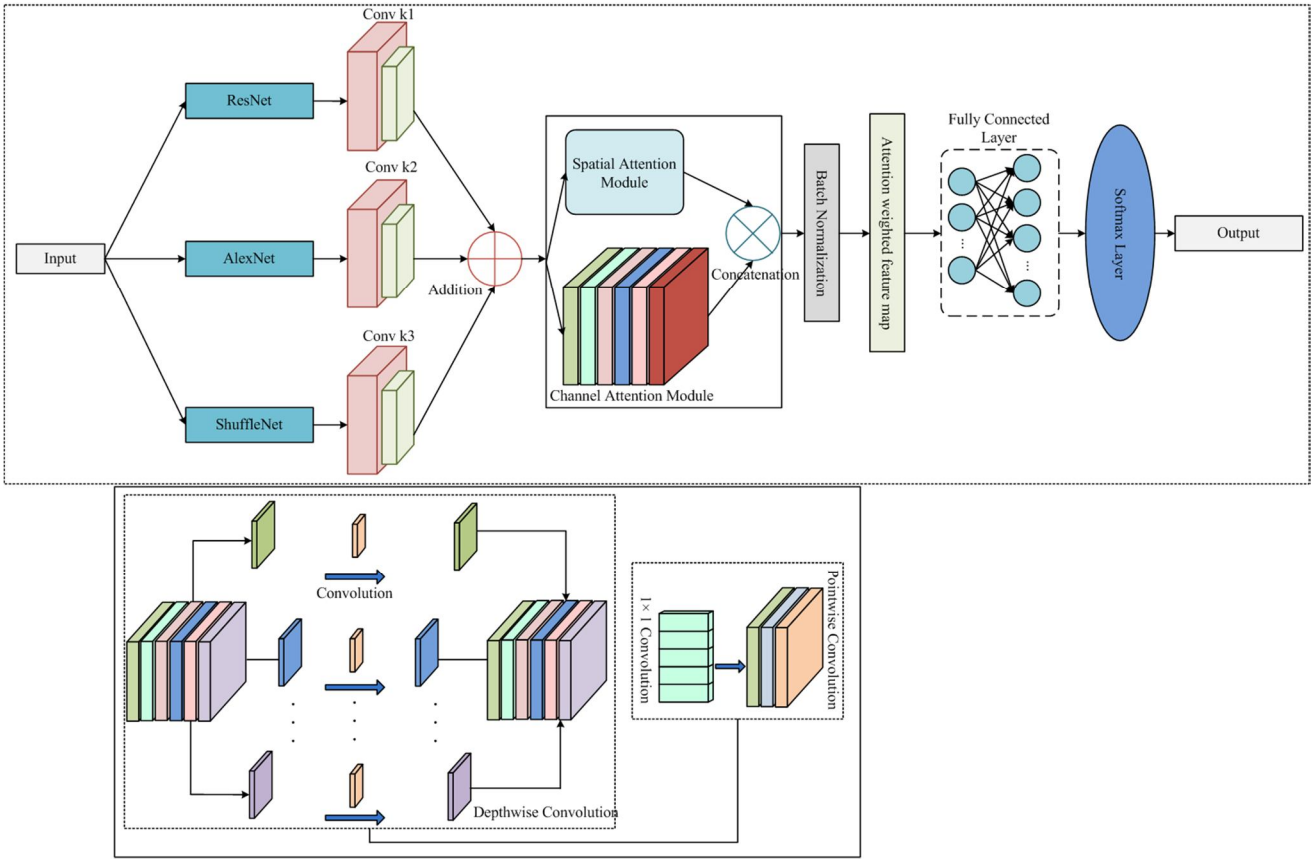


Figure 3. RAS-Net Architecture: Ensemble of ResNet, AlexNet, and ShuffleNet Actors for Lightweight Feature Extraction

B. DBSCAN-Based Task Clustering

Each task is represented as a feature vector $E_j = [B_s^j, y_s^j, g_s^j, W, P_j(s), X_j(s), K]$capturing data volume, offloading decision, channel condition, bandwidth, queue lengths, and Lyapunov parameter. Optimal DBSCAN parameters (ε, MinPts) are determined via grid search to minimise overall queue length. Tasks are classified as core, border, or noise points:
Cluster(j) = core point cluster if

$$Cluster(j) = \begin{cases} core\ point\ cluster, & if \left|\{i | c(E_j, E_i) \le \epsilon_{best}\}\right| \ge MinPts_{best} \\ border\ point\ cluster, & if \exists\ i\ s.t.\ d(E_j, E_i) \le \epsilon_{best}\ and\ \ i\ is\ a\ core\ point \\ noise, & otherwise \end{cases}$$

A readout function aggregates hidden states across clusters to compute Q-values: P(t,b,θ) = R(Σ_{l∈ε} aggr(g_l)), enabling consistent offloading decisions across grouped tasks.

C. Ensemble RAS-Net Actor Module

RAS-Net combines three complementary CNN architectures: ResNet uses residual connections (x = E(y,W) + y) for stable deep feature extraction and gradient flow. AlexNet provides global feature patterns and enhanced nonlinear representation. ShuffleNet applies grouped convolution with channel shuffling (x = Shuffle(y)) for computationally efficient lightweight processing suitable for edge deployment. The ensemble strategy — using multiple independent actor networks with weighted aggregation $y^s = \frac{1}{M_b}\sum_{j=1}^{M_b} \hat{y}_j^s$;— reduces Q-value estimation bias, improves exploration, and stabilises policy convergence compared to single-DNN DRL approaches.

D. Simulation Configuration

Table 3. Experimental Setup and Simulation Parameters

Parameter	Value / Description
Hardware	NVIDIA GeForce RTX 4090 (24 GB VRAM), Intel Core i9-13900K CPU, 64 GB DDR5 RAM
Software	Ubuntu 20.04 LTS, Python 3.9.16, PyTorch 2.0.1, CUDA 12.1
Baseline models	DNN, CNN, ResNet, LSTM
Evaluation metrics	Accuracy, Precision, F1-score, Sensitivity, Specificity, NPV, MCC, FPR, FNR
Validation strategy	5-fold cross-validation (K1–K5)
Offloading decision	Binary (local processing or full offloading to edge server)
Channel model	Block fading; i.i.d. across time slots; TDMA multiple access

V. RESULTS AND DISCUSSION

A. Overall Performance Comparison

Table 4 presents the performance of TOMEC-RAS-Net against DNN, CNN, ResNet, and LSTM baselines across all evaluation metrics. TOMEC-RAS-Net consistently outperforms all baselines, achieving 98.74% accuracy — surpassing LSTM (96.12%), ResNet (95.41%), CNN (94.26%), and DNN (92.84%). The proposed model also demonstrates superior precision (98.92%), sensitivity (98.65%), specificity (99.01%), F1-score (98.78%), and MCC (97.46% vs. LSTM's 92.41%).

Table 4. Performance Comparison: TOMEC-RAS-Net vs. Baseline Models (Best values highlighted in green)

Metric	DNN	CNN	ResNet	LSTM	TOMEC-RAS-Net
Accuracy (%)	92.84	94.26	95.41	96.12	98.74
Precision (%)	—	—	—	—	98.92
Sensitivity (%)	—	—	—	—	98.65
Specificity (%)	—	—	—	—	99.01
F1-Score (%)	—	—	—	—	98.78
MCC (%)	—	—	90.84	92.41	97.46
FPR (%)	6.24	4.82	4.24	3.48	0.99
FNR (%)	7.55	6.05	5.12	4.09	1.35

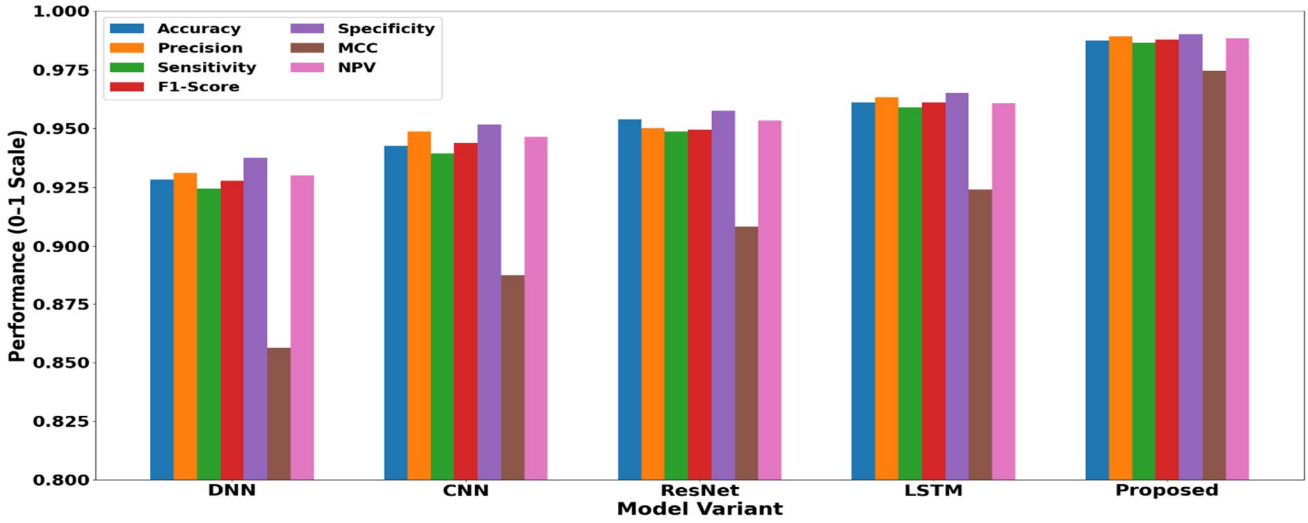


Figure 4. Comparative Analysis: Proposed TOMEC-RAS-Net vs. DNN, CNN, ResNet, and LSTM Across All Metrics

B. Error Rate Analysis

TOMEC-RAS-Net achieves the lowest error rates among all compared models with a False Positive Rate (FPR) of 0.99% and False Negative Rate (FNR) of 1.35%. In contrast, DNN exhibits 6.24% FPR and 7.55% FNR, CNN shows 4.82% FPR and 6.05% FNR, ResNet achieves 4.24% FPR and 5.12% FNR, and LSTM records 3.48% FPR and 4.09% FNR. The reduction in error rates confirms that TOMEC-RAS-Net significantly minimises incorrect offloading decisions — both false offloading triggers (FPR) and missed offloading opportunities (FNR) — which directly translates to improved MEC system reliability.

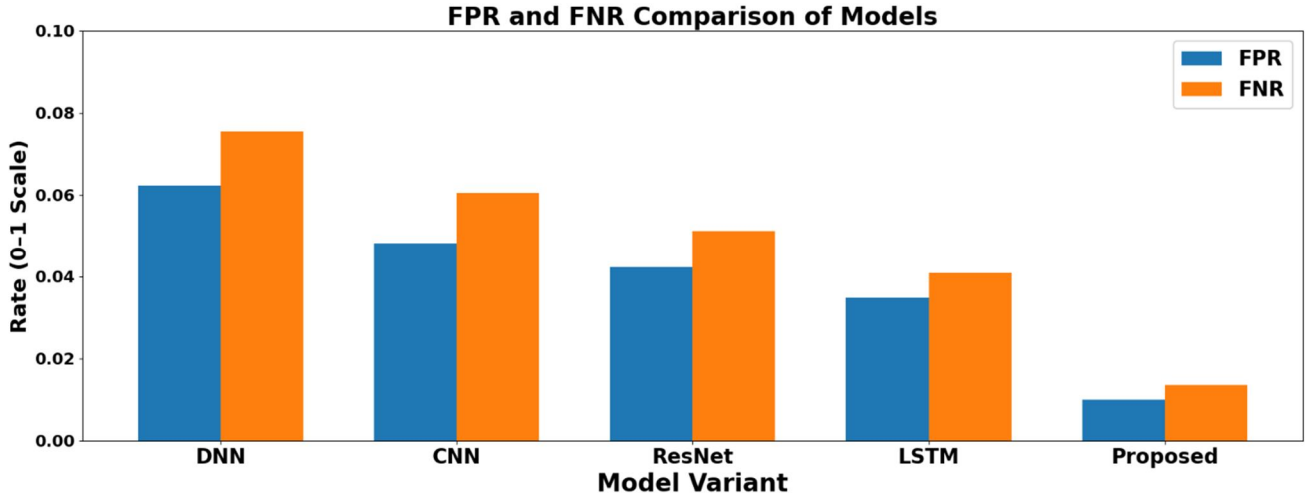


Figure 5. FPR and FNR Comparison: TOMEC-RAS-Net Achieves Lowest Error Rates Among All Models

C. Training Convergence

Figure 6 illustrates the training and validation accuracy and loss curves across training epochs. Training accuracy increases steadily from 0.30 at epoch 5 to 0.97 at epoch 95, while validation accuracy progresses from 0.35 to 0.99 over the same period. Both training and validation loss exhibit consistent parallel decay — training loss reduces from approximately 0.075 to 0.004, and validation loss from 0.060 to 0.003 — with no evidence of overfitting. This convergence behaviour confirms that the ensemble RAS-Net architecture, combined with priority-weighted resampling, achieves stable and efficient learning of optimal offloading policies without divergence or instability in dynamic MEC environments.

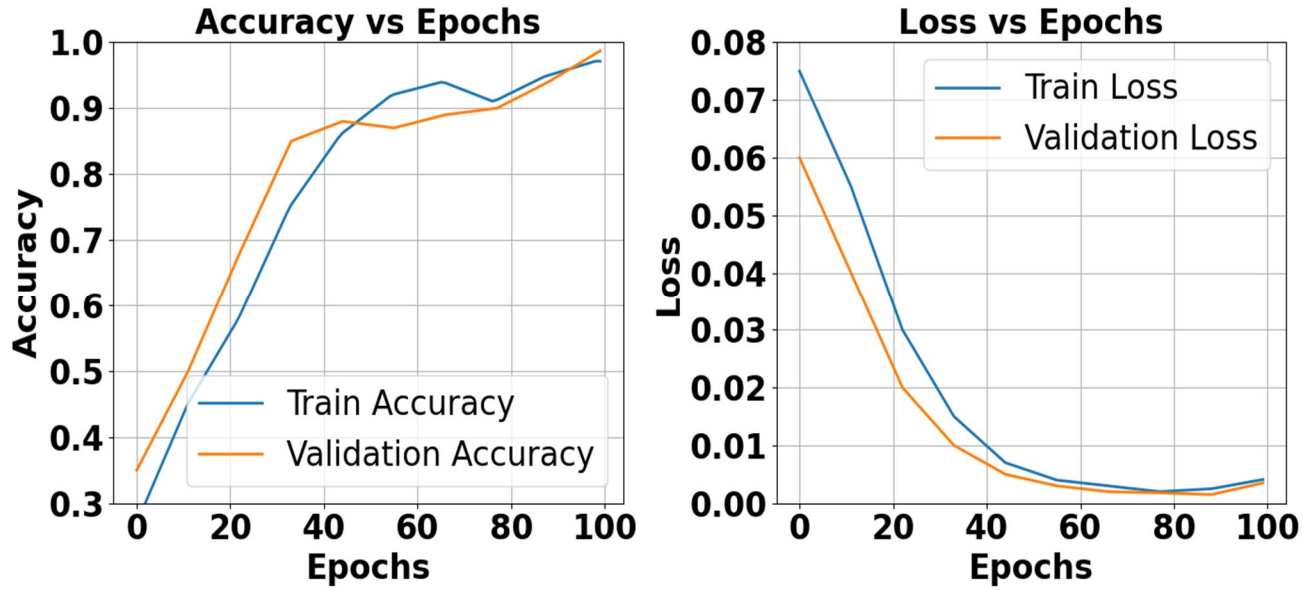


Figure 6. Training Behaviour: (a) Training and Validation Accuracy; (b) Training and Validation Loss Across Epochs

D. Cross-Validation Stability

Table 5 and Figure 7 present the five-fold cross-validation accuracy across K1–K5 folds. TOMEC-RAS-Net achieves fold accuracies of 98.51%, 98.92%, 98.67%, 99.02%, and 98.58%, yielding a mean of 98.74% with low variance — demonstrating consistent generalisation across different data splits. This is in stark contrast to DNN (mean 92.64%) and CNN (mean 94.26%), which exhibit greater fold-to-fold variability, indicating lower generalisation reliability.

Table 5. Five-Fold Cross-Validation Accuracy (%) — All Models (Best values highlighted in green)

Model	K1 (%)	K2 (%)	K3 (%)	K4 (%)	K5 (%)	Mean (%)
DNN	92.61	93.02	92.45	93.11	92.01	92.64
CNN	94.10	94.52	94.01	94.88	93.79	94.26
ResNet	—	—	—	—	—	95.41
LSTM	—	—	—	—	—	96.12
TOMEC-RAS-Net	98.51	98.92	98.67	99.02	98.58	98.74

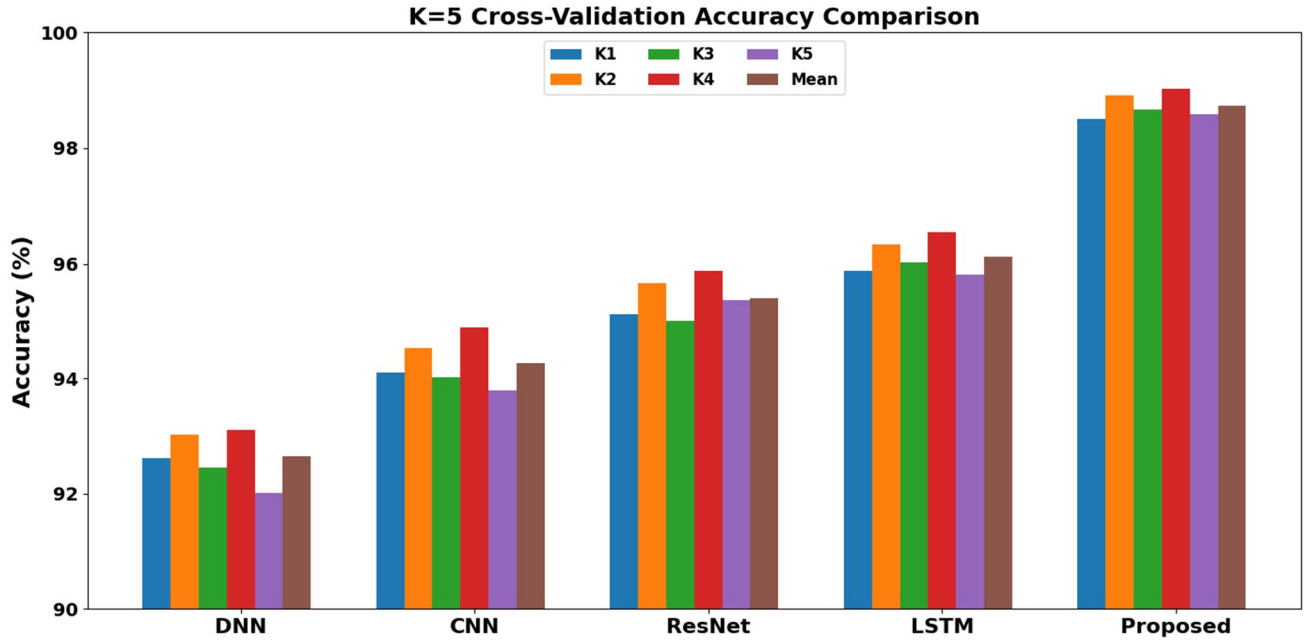


Figure 7. Five-Fold Cross-Validation Accuracy: TOMEC-RAS-Net Achieves Consistent High Performance Across All Folds

VI. CONCLUSION

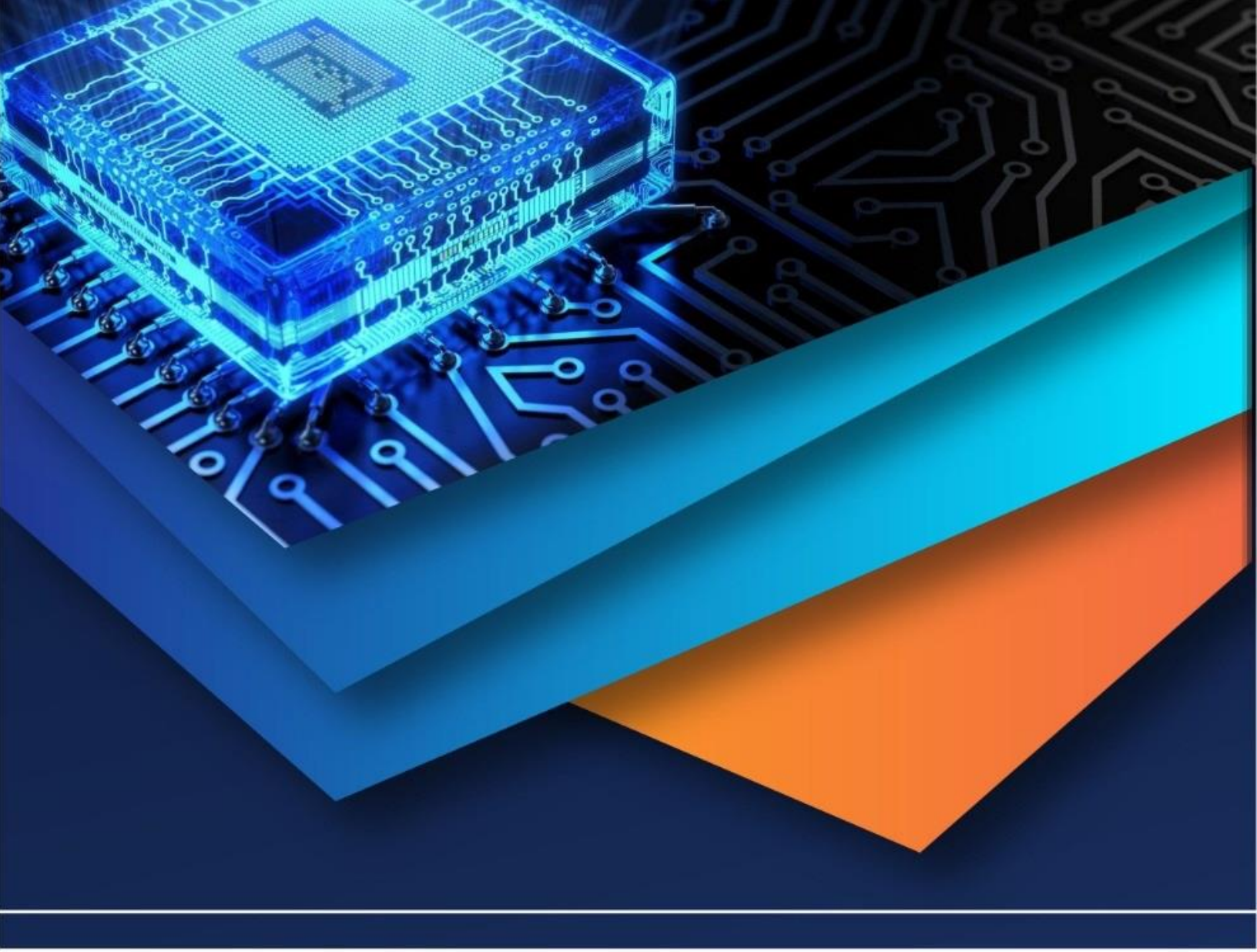
This paper presented TOMEC-RAS-Net, a lightweight ensemble deep reinforcement learning framework for adaptive task offloading in Mobile Edge Computing. The framework integrates three novel components: (i) DBSCAN-based density clustering for task grouping that reduces queue congestion and computational redundancy; (ii) ensemble RAS-Net actor architecture combining ResNet, AlexNet, and ShuffleNet for lightweight yet robust feature extraction with reduced estimation bias and improved convergence stability; and (iii) priority-weighted resampling for adaptive learning efficiency under non-stationary MEC conditions. The system jointly optimises binary offloading decisions, CPU frequency, and transmission power within a unified MDP formulation subject to queue stability and average power constraints.

Experimental results demonstrate that TOMEC-RAS-Net outperforms DNN, CNN, ResNet, and LSTM baselines across all evaluation metrics, achieving 98.74% accuracy, 98.92% precision, 99.01% specificity, MCC of 97.46%, FPR of 0.99%, and FNR of 1.35%. Stable convergence and consistent five-fold cross-validation performance (mean 98.74%) confirm generalisation reliability. These results establish TOMEC-RAS-Net as a viable, accurate, and efficient solution for real-time task offloading in resource-constrained MEC environments.

Future work will focus on deployment on real 5G/6G MEC infrastructure, extension to multi-agent federated DRL for scalable cooperative offloading, integration of energy harvesting and green computing strategies for battery-constrained IoT devices, and incorporation of blockchain-based security mechanisms for privacy-preserving task offloading. Adaptive lightweight neural architectures and dynamic clustering algorithms will also be explored to further reduce computational demands on severely resource-limited edge nodes.

REFERENCES

[1] Tang, M., & Wong, V. W. (2020). Deep reinforcement learning for task offloading in mobile edge computing systems. IEEE Trans. Mobile Computing, 21(6), 1985–1997.
[2] Lu, H., et al. (2020). Optimization of lightweight task offloading strategy for mobile edge computing based on deep reinforcement learning. Future Generation Computer Systems, 102, 847–861.
[3] Zhang, S., et al. (2022). DRL-based partial offloading for maximizing sum computation rate of wireless powered MEC network. IEEE Trans. Wireless Communications, 21(12), 10934–10948.
[4] Zheng, K., et al. (2023). DRL-based offloading for computation delay minimization in wireless-powered multi-access edge computing. IEEE Trans. Communications, 71(3), 1755–1770.
[5] Li, Y., et al. (2020). Distributed edge computing offloading algorithm based on deep reinforcement learning. IEEE Access, 8, 85204–85215.
[6] Lim, D., et al. (2022). DRL-OS: A deep reinforcement learning-based offloading scheduler in mobile edge computing. Sensors, 22(23), 9212.
[7] Nashaat, H., et al. (2024). DRL-based distributed task offloading framework in edge-cloud environment. IEEE Access, 12, 33580–33594.
[8] Wang, S., et al. (2022). Deep reinforcement learning-based adaptive resource management in MEC. IEEE Trans. Network and Service Management.
[9] Ma, G., et al. (2023). Dynamic neural network-based resource management for 6G MEC networks. IEEE Transactions on Communications.
[10] Duan, M., et al. (2021). Resource management for intelligent vehicular edge computing networks. IEEE Trans. Intelligent Vehicles.
[11] Chen, Y., et al. (2022). Dynamic task offloading for mobile edge computing with hybrid energy supply. Tsinghua Science and Technology, 28(3), 421–432.
[12] Yang, S., et al. (2022). Deep learning-based dynamic computation task offloading for MEC networks. Sensors, 22(11), 4088.
[13] Yuan, H., et al. (2024). Cost-efficient task offloading in MEC with layered UAVs. IEEE Internet of Things Journal, 11(19), 30496–30509.
[14] Seid, A. M., et al. (2021). Multi-agent DRL for task offloading and resource allocation in multi-UAV IoT edge network. IEEE Trans. Network and Service Management, 18(4), 4531–4547.
[15] Zhao, F., et al. (2021). Dynamic offloading and resource scheduling for MEC with energy harvesting devices. IEEE Trans. Network and Service Management, 18(2), 2154–2165.
[16] Qin, P., et al. (2022). Learning-based energy-efficient task offloading for vehicular collaborative edge computing. IEEE Trans. Vehicular Technology, 71(8), 8398–8413.
[17] Bolourian, M., & Shah-Mansouri, H. (2023). Energy-efficient task offloading for three-tier wireless-powered MEC. IEEE Internet of Things Journal, 10(12), 10400–10412.
[18] Qin, Y., et al. (2025). Task offloading optimization in MEC based on DRL using density clustering and ensemble learning. Scientific Reports, 15(1), 211.
[19] Bi, S., et al. (2020). Lyapunov-guided deep reinforcement learning for stable online computation offloading in MEC networks. arXiv preprint.
[20] Ullah, I., et al. (2023). Optimizing task offloading and resource allocation in edge-cloud networks: a DRL approach. Journal of Cloud Computing, 12(1), 112.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY