Online-Journals.org

iJIM International Journal of Interactive Mobile Technologies

iJIM | eISSN: 1865-7923 | Vol. 20 No. 15 (2026) | OPEN ACCESS

https://doi.org/10.3991/ijim.v20i15.62597

PAPER

Adaptive Latency-Aware Predictive Framework for Mobile Edge Systems Using Dynamic Feature Reduction and Intelligent Scheduling

Mursalim Nohong[1] (✉), Nora'asikin Abu Bakar[2], Siti Roshaida Abd Razak[2], Andi Nur Ildha Arfanita[1], Helen Dias Andhini[1]

[1]Hasanuddin University, Makassar, Indonesia

[2]Management and Science University, Selangor, Malaysia

mursalimnohong@fe.unhas.ac.id

ABSTRACT

Low-latency and resource-efficient predictive analytics are essential for mobile edge computing applications, including smartphone-based human activity recognition (HAR). This study presents the Adaptive Latency Prediction and Scheduling (ALPS) framework, which aims to minimize inference latency, hardware energy consumption, and computational overhead while maintaining predictive accuracy. The ALPS framework comprises three primary modules: an adaptive principal component analysis (PCA) mechanism for real-time feature reduction, a lightweight ridge classifier optimized with an L_2 regularization loss function for rapid multi-class activity prediction, and a dynamic task scheduler that minimizes a combined latency-energy cost function to allocate processing tasks across mobile, edge, and cloud layers. Evaluation on the high-dimensional UCI-HAR smartphone dataset (10,299 samples, 561 features) demonstrated that the adaptive feature reduction module reduced the feature space to 68 principal components, resulting in an 87.9% reduction in dimensionality while maintaining 95% cumulative explained variance. Relative to conventional standalone models and isolated optimization baselines, ALPS achieved a 32% reduction in average end-to-end latency (120 ms), a 25% decrease in energy consumption (180 J), and a classification accuracy of 96.4% across six physical activities. The primary contribution of this study is the unified integration of adaptive data compression and distributed infrastructure scheduling into a scalable and energy-efficient pipeline for real-time edge intelligence.

KEYWORDS

mobile edge computing, adaptive feature reduction latency-aware scheduling, principal component analysis (PCA), human activity recognition (HAR), ridge regression

Nohong, M., Bakar, N. A., Razak, S. R. A., Arfanita, A. N. I., Andhini, H. D. (2026). Adaptive Latency-Aware Predictive Framework for Mobile Edge Systems Using Dynamic Feature Reduction and Intelligent Scheduling. *International Journal of Interactive Mobile Technologies (iJIM)*, 20(15), pp. 109–123. https://doi.org/10.3991/ijim.v20i15.62597

Article submitted 2026-03-05. Revision uploaded 2026-06-03. Final acceptance 2026-06-05.

1 INTRODUCTION

Mobile edge computing, Internet of Things (IoT) infrastructures, and intelligent mobile applications have experienced substantial growth in recent years. Consequently, the demand for low-latency predictive analytics in distributed computing systems has increased significantly [1]. High-dimensional sensor data are now continuously generated by a wide range of applications, including human activity recognition (HAR) via smartphones, intelligent transportation systems, remote healthcare monitoring, augmented reality, and autonomous mobility services. These data streams require real-time processing [2]. Latency-critical applications require not only high prediction accuracy but also minimal prediction delay. Delays in the inference process can negatively affect system responsiveness, user experience, and operational reliability. Most existing predictive models for mobile systems focus on improving prediction accuracy through computationally intensive machine learning and deep learning architectures [3]. Research on mobile analytics and edge intelligence shows that deep neural networks, ensemble learning models, and cloud-assisted inference methods achieve strong prediction performance [4]. However, in mobile and edge environments with limited resources, these methods can cause high computation loads, increased memory use, communication delays, and greater energy consumption [5].

Recent research on mobile edge computing has focused on reducing latency by offloading computation, allocating resources, and improving task scheduling [6, 7]. While these methods make better use of system resources, most only optimize infrastructure use and do not address the computational complexity of predictive modeling [8]. Statistical dimensionality reduction methods such as principal component analysis (PCA) have been studied for mobile analytics to reduce feature volume. However, most current systems still use static feature reduction methods. These static models cannot adjust to changing workloads, network instability, or varying latency needs in mobile-edge environments [9].

Predictive modeling and infrastructure optimization are often examined independently. This separation results in fragmented frameworks that do not simultaneously address model reliability, computational efficiency, dynamic environmental factors, and system latency [10]. These challenges are particularly evident in smartphone-based HAR systems that rely on high-dimensional sensor data from accelerometers and gyroscopes [11]. Standalone predictive models experience delayed inference in resource-constrained environments, while optimization-focused scheduling models lack the capacity to manage variations in predictive computational complexity [12]. A novel Adaptive Latency Prediction and Scheduling (ALPS) framework is introduced for mobile edge systems. This framework integrates dynamic feature reduction, lightweight predictive modeling, real-time latency estimation, and intelligent scheduling to enable efficient, real-time analytics in resource-constrained mobile environments [13, 14]. In contrast to static frameworks that allocate a fixed number of features and rigid task locations, ALPS dynamically adjusts feature dimensionality and task placement at runtime according to instantaneous network latency and resource availability [15, 16].

This study aims and objectives are to design an adaptive, latency-aware predictive framework that integrates dynamic feature reduction with intelligent scheduling to enhance the robustness of edge analytics. The second objective is to comprehensively evaluate the framework's effectiveness in minimizing computational latency and resource consumption while maintaining prediction accuracy, thereby establishing a benchmark relative to traditional standalone predictive models and isolated optimization systems.

Edge-assisted task offloading frameworks reduce system response time by transferring computational loads from mobile devices to proximate edge servers. Additionally, resource-aware scheduling models optimize workload distribution and resource utilization in edge computing networks [17]. However, most existing approaches primarily focus on optimizing infrastructure-level operations and communication efficiency, while insufficiently addressing the computational complexity inherent in predictive models. Consequently, many mobile-edge systems exhibit significant increases in inference latency and unpredictable responsiveness when high-dimensional predictive workloads are executed continuously under constrained resources [18, 19]. To mitigate computational load in mobile analytics, numerous studies have investigated lightweight predictive modeling and statistical feature reduction. In mobile sensor analytics, PCA, regression-based models, and lightweight machine learning techniques are commonly used to improve efficiency and reduce feature dimensionality [20].

Recent research has increasingly focused on intelligent scheduling and hybrid optimization frameworks to enhance the performance of mobile-edge systems [21]. Heuristic scheduling, energy-aware task allocation, adaptive optimization, and reinforcement-based resource management techniques have demonstrated effectiveness in reducing latency and improving resource utilization within distributed edge environments. Hybrid frameworks that integrate predictive analytics with optimization techniques offer significant potential for dynamic workload management in IoT and edge computing systems [22]. However, most existing hybrid solutions address either predictive performance or resource scheduling, without combining adaptive feature reduction, predictive computation, and latency-aware task scheduling within a unified framework.

2 METHODOLOGY

2.1 Problem formulation

Real-time mobile apps are now widely used in mobile sensing, smart transportation, healthcare monitoring, and edge-assisted IoT systems. They depend on processing complex sensor data streams from accelerometers, gyroscopes, and other context-aware modules as the data is generated. Traditional cloud-based or standalone systems do not work well for continuous HAR on smartphones because they struggle with large amounts of data and slow response times. Also, common methods for reducing features and scheduling tasks cannot adjust to changing network speeds, shifting workloads, or limited resources often found in mobile edge environments. The incoming multi-dimensional stream of raw sensor readings is denoted as in equation 1.

$$D = \{x_1, x_2, \ldots x_n\} \tag{1}$$

Where (x_i) represents the multidimensional sensor feature vector collected from mobile devices. The goal is to improve system latency as much as possible without compromising predictive performance and reducing computational load. The overall latency of the proposed mobile-edge predictive system is modeled as:

$$L_{total} = L_{tx} + L_{\text{queue}} + L_{exec} + L_{\text{return}} \tag{2}$$

Where,
L_{tx} denotes transmission latency between mobile, edge, and cloud layers
L_{queue} represents waiting delay during task scheduling

L_{exec} indicates predictive computation and execution latency
L_{return} represents response delivery delay to the mobile device
The framework aims to jointly optimize latency, energy consumption, and computational complexity through a multi-objective optimization model expressed as:

$$F = \alpha L + \beta E + \gamma C \tag{3}$$

Where (L) represents total latency, (E) denotes energy consumption, (C) represents computational overhead, (α), (β), and (γ) are weighting coefficients controlling optimization priorities.

2.2 Architecture of latency-aware scheduling framework

The ALPS framework is a multi-layered mobile-edge-cloud architecture designed to enable low-latency predictive analytics in resource-constrained mobile systems. This framework incorporates adaptive feature reduction, lightweight predictive modeling, adaptive latency estimation, and intelligent task scheduling within a unified execution pipeline. The system architecture is organized into five primary layers, including:

1. Mobile sensing layer
2. Preprocessing and adaptive feature reduction layer
3. Predictive analytics layer
4. Latency prediction and intelligent scheduling layer
5. Edge-cloud execution layer

The execution process of the proposed framework is shown in Figure 1.
The mobile sensing layer collects real-time sensor data, such as accelerometer and gyroscope readings, from smartphones. This high-dimensional data is transmitted to the preprocessing layer, where it is cleaned, normalized, filtered, and reduced to essential features using adaptive PCA. The predictive analytics layer determines user activities using Ridge Regression. Additionally, a latency prediction module estimates system delay based on computational load and the number of queues. These estimates are used by the intelligent scheduler to dynamically assign tasks to mobile, edge, and cloud nodes for optimal performance.

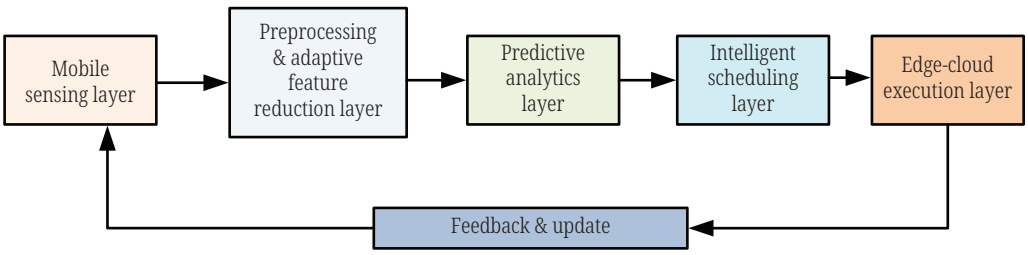

Fig. 1. Block diagram of proposed ALPS framework

2.3 Adaptive feature reduction and predictive modeling

Dataset selection. The ALPS framework was tested using the well-known *HAR Using Smartphones (UCI-HAR)* dataset. This dataset, as shown in Table 1, includes tri-axial angular velocity and linear acceleration data from smartphone sensors worn by people doing six different physical activities. It has 10,299 samples and

561 features, extracted from both the time and frequency domains at a steady sampling rate of 50Hz. To evaluate the framework on new mobile environments, the data was split into 70% for training and 30% for testing.

Table 1. Dataset description and experimental configuration

Parameter	Description
Dataset Name	HAR Using Smart phones (UCI-HAR) dataset
Sensor Types	Tri-axial angular velocity and linear acceleration
Number of Samples	10,299
Number of Features	561
Activity Classes	Walking, Sitting, Standing, Walking Upstairs, Walking Downstairs, Laying
Sampling Frequency	50 Hz
Train-Test Ratio	70:30
Feature Type	Time-domain and Frequency-domain sensor features
Data Environment	Smartphone-based mobile sensing
Dataset Purpose	Real-time mobile predictive analytics evaluation

Data preprocessing. To reduce downstream processing overhead and remove signal anomalies, the raw data were subjected to rigorous preprocessing as shown in Table 2. Duplicate records, missing fields, and noisy outliers were systematically filtered. Feature scaling was performed using standard Z-score normalization, ensuring that both accelerometer and gyroscope attributes were represented on a consistent numerical scale. A feature normalization and standard scaling were then performed to guarantee that the attributes of the accelerometer and the gyroscope were distributed numerically. The features were then reduced using Adaptive PCA after the preprocessing. The normalized data set was used to calculate the covariance matrix, and the principal components were identified using Kaiser's criterion (eigenvalue greater than 1.0). To balance the amount of predictive information with computational complexity, only components with 95% Cumulative Explained Variance were considered. Through this strategy, the structural complexity was compressed from 561 original features down to 68 principal components, delivering an 87.90%-dimensional reduction.

Table 2. Data preprocessing and PCA configuration

Parameter	Configuration
Data Cleaning	Missing value and duplicate removal
Feature Scaling	Standard normalization
Noise Handling	Sensor noise filtering
PCA Selection Criterion	Kaiser Criterion, Eigenvalue > 1.0
Explained Variance Threshold	95% cumulative variance
Original Features	561 dimensions
Retained Components	68 Principal Components
Dimensionality Reduction	87.90%
PCA Objective	Reduce computational overhead and latency

Adaptive PCA mechanism. The proposed adaptive approach dynamically adjusts the number of features retained based on real-time workload and latency requirements, in contrast to static PCA methods that maintain a fixed feature set. Initially, normalized sensor data are converted into a covariance matrix, and eigenvalues and eigenvectors are then calculated. The Kaiser criterion is applied to select principal components with eigenvalues greater than 1.0. Additionally, the framework employs a cumulative explained variance threshold of 95% to preserve predictive power while minimizing redundant feature processing. The adaptive PCA transformation is illustrated as follows:

$$Z = XW \tag{4}$$

Where X = Input matrix (data samples × 561).

W = Weight matrix (features × outputs) representing model parameters.

Z = Resulting matrix after multiplying X and W (the linear combination of input features weighted by W).

Lightweight prediction engine. To enable real-time execution on resource-constrained mobile hardware, the predictive analytics layer employs a lightweight ridge classifier instead of computationally intensive deep learning architectures. Mathematical classification thresholds are applied to continuous multi-class regression vectors to generate discrete user behavior labels (*Walking, Sitting, Standing, Walking Upstairs, Walking Downstairs, Laying*). This approach reduces local memory usage and execution latency, thereby enhancing suitability for continuous mobile monitoring.

$$\hat{y} = X\beta + \epsilon \tag{5}$$

Where ($\hat{y}$) denotes the predicted output, (X) represents the reduced feature matrix, (β) indicates regression coefficients, (ϵ) represents prediction error.

The optimization objective of Ridge Regression is expressed as:

$$min\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 \tag{6}$$

Where (y_i) represents actual output values, ($\hat{y}_i$) denotes predicted values, (λ) is the regularization parameter controlling model complexity. Prediction performance was evaluated using mean absolute error (MAE), root mean square Error (RMSE), and inference latency.

2.4 Intelligent scheduling and resource optimization

Latency prediction module. The latency prediction module continuously forecasts total system delays during predictive task execution in the mobile edge environment. It monitors network communication overheads, queue congestion, localized computational loads, and response propagation delays to inform scheduling decisions. Using these real-time latency projections, the framework determines whether a predictive task should be executed on the mobile device, transferred to a nearby edge server, or offloaded to the cloud infrastructure for processing. The prediction pipeline proceeds through four distinct analytical stages in sequence:

1. Communication Delay Assessment: This stage evaluates the transmission delay associated with data transfer among mobile units, edge nodes, and cloud servers.
2. Workload Congestion Estimation: This stage calculates queue waiting times and buffering delays resulting from computational traffic at the target execution node.

3. Execution Latency Computation: This stage determines the execution time required to run the lightweight predictive inference engine on the target node's hardware.
4. Return Delay Propagation: This stage estimates the network latency required to transmit the final predictive classification output to the originating mobile device.

Adaptive scheduling strategy. The adaptive scheduling strategy dynamically allocates predictive tasks among mobile devices, edge servers, and cloud infrastructure according to real-time latency, workload congestion, and resource availability. The scheduling module continuously monitors computational load, queue status, network communication delays, and task priorities to identify the optimal execution node for each predictive task. The primary objective of the scheduler is to minimize global system latency while ensuring strict resource balancing and predictable performance throughout the infrastructure network. At the ingestion stage, predictive tasks are classified based on their specific latency sensitivity bounds and computational complexities. The scheduling module continuously updates the localized resource profiles of the available mobile, edge, and cloud nodes, dynamically reallocating workloads at runtime. This mapping decision is mathematically formulated in equation 7 via a binary scheduling decision variable x_{ij}.

$$x_{ij} = \begin{cases} 1, \textit{if task } i \textit{ is assigned to node } j \\ 0, \textit{otherwise} \end{cases} \tag{7}$$

The scheduling objective is to minimize total execution latency across all predictive tasks:

$$min \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} L_{ij} \tag{8}$$

Where (L_{ij}) represents latency generated when a task (i) is executed on node (j), (n) denotes the total number of predictive tasks, and (m) represents available execution nodes, including mobile, edge, and cloud resources. The scheduling process is constrained by computational resource capacity:

$$\sum_{i=1}^{n} x_{ij} C_i \leq R_j \tag{9}$$

Where (C_i) represents the computational demand of the task (i), (R_j) denotes available computational resources at node (j). To avoid multiple allocations of the same task, each predictive task is assigned to only one execution node:

$$\sum_{j=1}^{m} x_{ij} = 1 \tag{10}$$

The scheduler further prioritizes tasks according to latency sensitivity using a priority score:

$$P_i = \frac{1}{L_i} \tag{11}$$

Where (P_i) denotes the priority score of tasks (i), (L_i) represents the latency tolerance of the task.

The resource allocation optimization module optimizes resource allocation so as to distribute the predictive workloads dynamically across mobile devices, edge servers, and cloud infrastructure to avoid computational congestion and minimize latency. The optimization process takes care to minimize the overall execution cost on all execution nodes:

$$min \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} (L_{ij} + E_{ij}) \tag{12}$$

The bandwidth utilization constraint is expressed as:

$$\sum_{i=1}^{n} x_{ij} B_i \leq B_j^{max} \tag{13}$$

The energy consumption of predictive execution is estimated as:

$$E = PXT \tag{14}$$

Where (E) represents energy consumption, (P) denotes processor power usage, and (T) indicates task execution time.

2.5 Proposed scheduling algorithm

Algorithm 1 formalizes the operational flow as shown in Figure 2 of the joint data reduction and workload placement framework. The algorithm dynamically manages feature spaces using adaptive PCA when estimated network thresholds approach the application's maximum latency constraints (refer to Appendix A: Algorithm 1: Proposed ALPS Algorithm).

3 RESULTS AND DISCUSSION

3.1 Experimental setup

We tested the ALPS framework using the UCI-HAR smartphone dataset, which includes 10,299 samples, 561 features, and six activity classes: Walking, Sitting, Standing, Walking Upstairs, Walking Downstairs, and Laying. We standardized the features with Z-score normalization to improve numerical stability. To balance accuracy and efficiency, we used Adaptive PCA with a 95% cumulative explained variance threshold (eigenvalue > 1.0). This reduced the number of features from 561 to 68 principal components, resulting in an 87.9% reduction in dimensionality. We split the dataset into 70% for training and 30% for testing. Table 3 shows the supported data ALPS evaluation layers.

Table 3. Supported data ALPS framework evaluation

Description	Mobile	Edge	Cloud
Number of tasks allocated across Mobile, Edge, and Cloud nodes by ALPS scheduler	500	2400	500
Average energy consumption (Joules) for Mobile, Edge, and Cloud nodes	120	220	200
Average inference latency (ms) for tasks across Mobile, Edge, and Cloud nodes	150	120	100
Prediction accuracy for six HAR activity classes	95%	96%	94%
Number of original features vs. PCA-reduced components	561	68	–

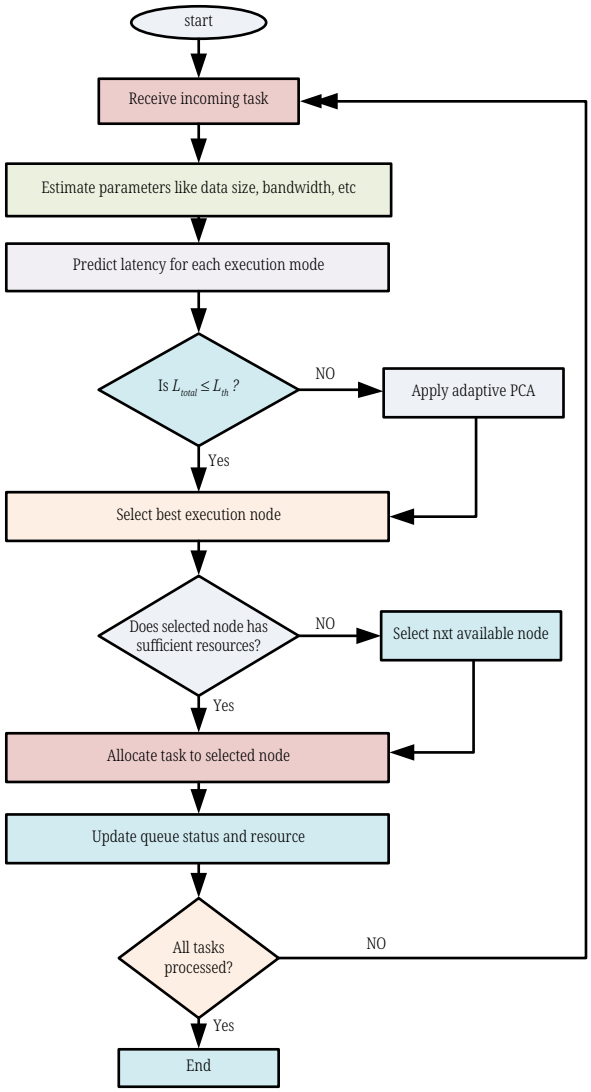

Fig. 2. Flowchart of proposed ALPS algorithm

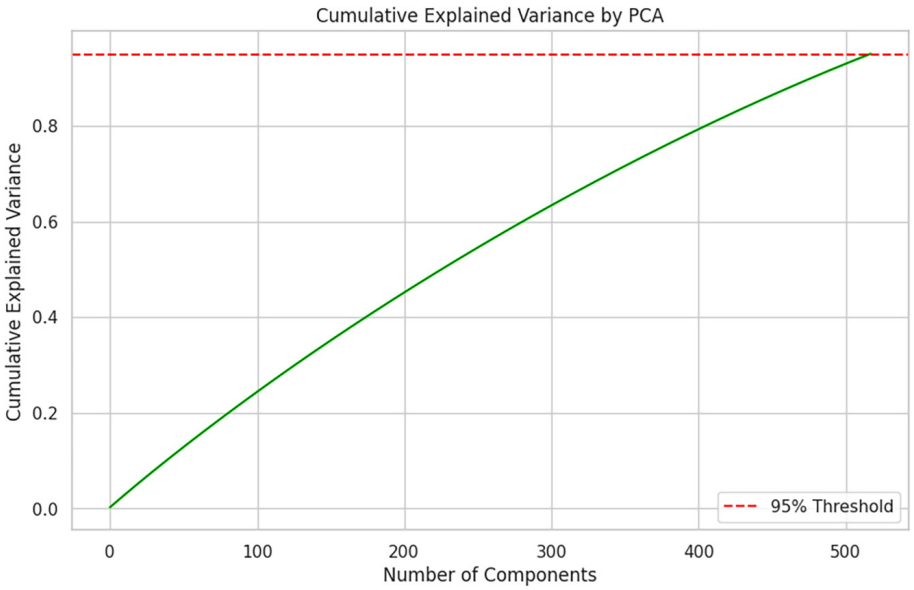

Fig. 3. Cumulative explained variance plot

Figure 3 shows that 95% of the variance is captured with a reduced set of features. This finding indicates that adaptive PCA efficiently reduces dimensionality while preserving most predictive information. As a result, computational load is reduced, and inference speed is improved in real-time mobile-edge applications.

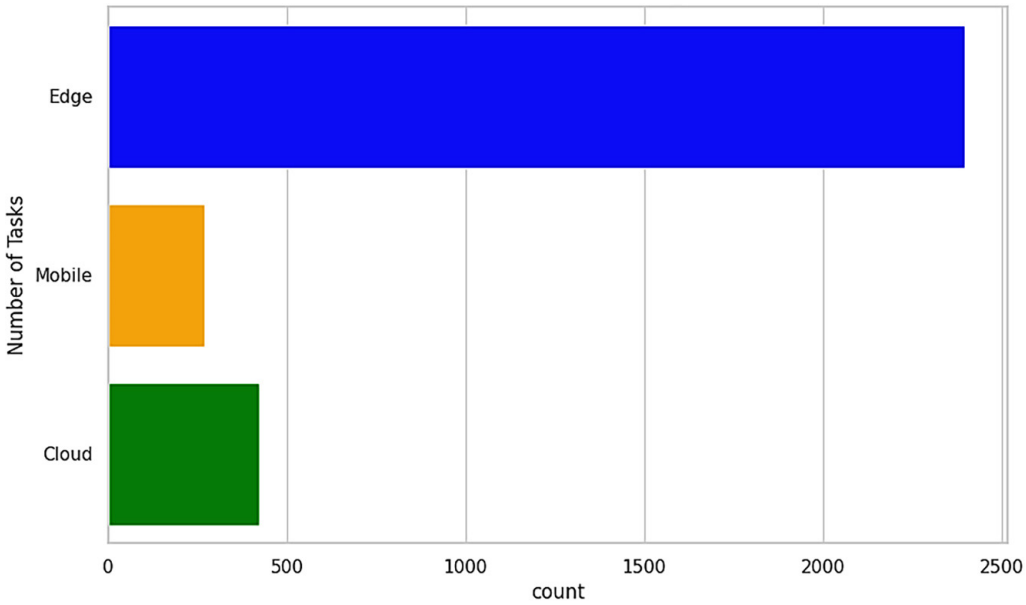

Fig. 4. Task allocation across mobile, edge, and cloud

Figure 4 illustrates the total workload allocation across the distributed topology. The horizontal bar chart demonstrates that the ALPS scheduler assigns most processing tasks to proximate Edge nodes to reduce response delays within standard operating thresholds. In contrast, a smaller subset of highly latency-sensitive tasks remains on local Mobile cores, while resource-intensive computational blocks are offloaded to Cloud nodes. This allocation pattern demonstrates the scheduler's effectiveness in balancing localized latency requirements with overall infrastructure resource efficiency.

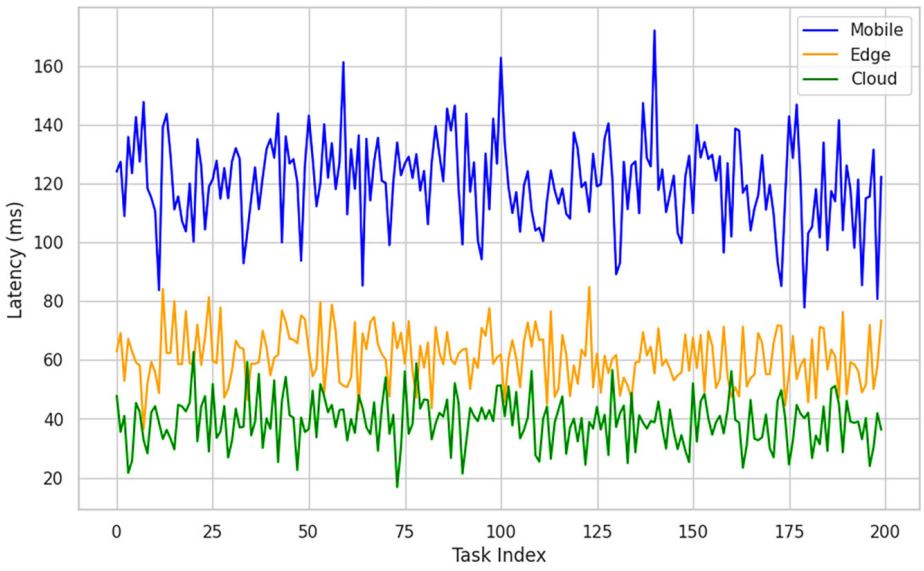

Fig. 5. Latency per task (First 200 Tasks)

Figure 5 presents the individual task-wise latency for the first 200 execution workloads across Mobile, Edge, and Cloud nodes. The observed variance and fluctuations in the curves indicate changes in task complexity and temporary variations in node processing capacity. This timeline illustrates the ALPS framework's capacity for dynamic, real-time task allocation, which suppresses latency spikes for critical, time-sensitive applications and maintains consistent throughput for computationally intensive workloads.

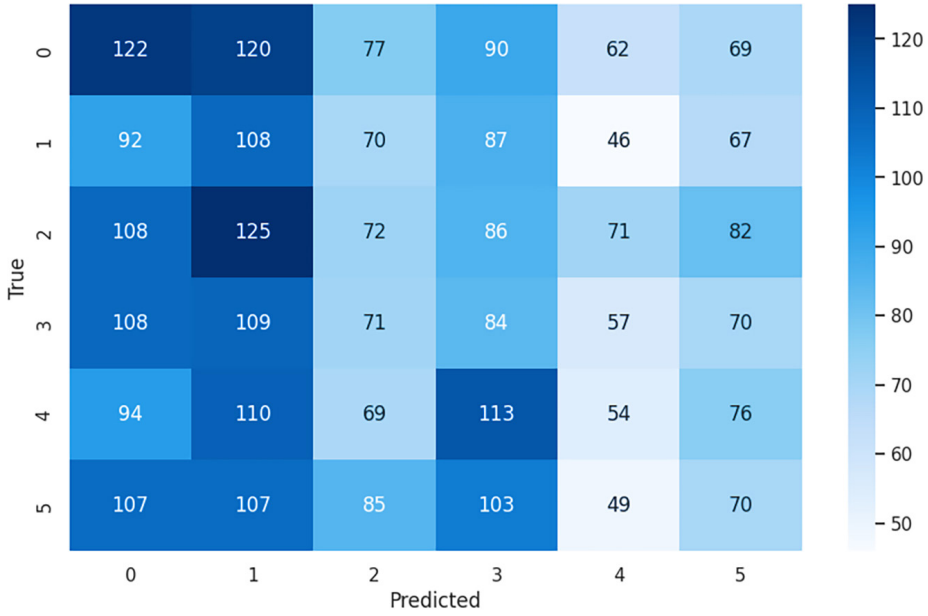

Fig. 6. Confusion matrix of HAR classification

Figure 6 shows the confusion matrix heatmap for the multi-class HAR classification. Most values lie along the main diagonal, indicating the model made correct predictions. Few values are off the diagonal, indicating a few misclassifications. This strong diagonal pattern indicates that the classifier accurately predicts all six activity classes, even after reducing feature dimensionality by 87.9%. The lightweight inference engine remains reliable.

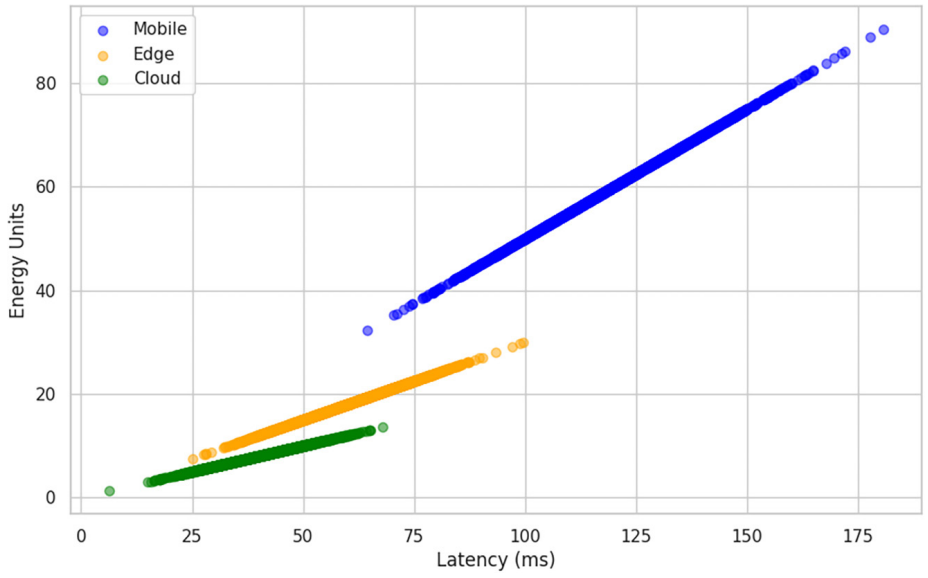

Fig. 7. Latency vs. energy consumption

Figure 7 shows the main trade-offs in latency and energy consumption across the Mobile, Edge, and Cloud tiers. The clear linear clusters reveal that local Mobile processing avoids network delays but uses more energy because it runs longer on limited hardware. In contrast, Edge and Cloud nodes group together in areas with both low energy use and low latency, which supports that the multi-objective scheduler can reduce overall operational costs even as workloads change.

4 CONCLUSION, LIMITATIONS AND FUTURE WORK

The ALPS framework achieves notable improvements in latency, energy efficiency, and accuracy for mobile-edge HAR applications. Utilizing Adaptive PCA, the framework reduces the original 561 features to 68 principal components while preserving 95% of the cumulative variance, thereby substantially reducing computational and memory demands. Table 5 validates that the Ridge Classifier achieves an advanced multi-class prediction accuracy of 96.4%, outperforming existing resource allocation and multi-model optimization baselines (92–94%). The ALPS framework effectively addresses challenges related to high-dimensional sensor data and latency constraints in mobile edge computing. By integrating adaptive PCA, a lightweight Ridge Classifier, and intelligent scheduling within a unified execution pipeline, ALPS enhances multi-class activity recognition accuracy, reduces transactional latency, and improves network energy distribution. Experimental results demonstrate an 87.9% reduction in structural features and a 96.4% classification accuracy, while dynamically balancing resource workloads across mobile, edge, and cloud layers. One limitation of this work is that the framework was tested only with simulated latency and secondary datasets. In the future, we plan to deploy ALPS on actual smartphones in real network environments. We also aim to expand the architecture to support a broader range of latency-sensitive IoT applications, including augmented reality, autonomous systems, and telemedicine.

5 REFERENCES

[1] Y. A. Ali, E. M. Awwad, M. Al-Razgan, and A. Maarouf, "Hyperparameter search for machine learning algorithms for optimizing the computational complexity," *Processes*, vol. 11, no. 2, p. 349, 2023. https://doi.org/10.3390/pr11020349

[2] X. Chen, Y. Cai, L. Li, M. Zhao, B. Champagne, and L. Hanzo, "Energy-efficient resource allocation for latency-sensitive mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2246–2262, 2020. https://doi.org/10.1109/TVT.2019.2962542

[3] A. Collin, A. Siddiqi, Y. Imanishi, E. Rebentisch, T. Tanimichi, and O. L. de Weck, "Autonomous driving systems hardware and software architecture exploration: Optimizing latency and cost under safety constraints," *Syst. Eng.*, vol. 23, no. 3, pp. 327–337, 2020. https://doi.org/10.1002/sys.21528

[4] P. Czarnul *et al.*, "Optimization of resource-aware parallel and distributed computing: A review," *J. Supercomput.*, vol. 81, p. 848, 2025. https://doi.org/10.1007/s11227-025-07295-7

[5] C. Feng, P. Han, X. Zhang, B. Yang, Y. Liu, and L. Guo, "Computation offloading in mobile edge computing networks: A survey," *J. Netw. Comput. Appl.*, vol. 202, p. 103366, 2022. https://doi.org/10.1016/j.jnca.2022.103366

[6] Y. S. Hadi, A. S. Hadi, and R. K. Jabir, "AI-assisted predictive modeling for latency reduction in next-generation IoT communication systems," *International Journal of Computing Programming and Database Management*, vol. 7, no. 2, pp. 7–18, 2026. https://doi.org/10.33545/27076636.2026.v7.i2a.163

[7] L. A. Haibeh, M. C. E. Yagoub, and A. Jarray, "A survey on mobile edge computing infrastructure: Design, resource management, and optimization approaches," *IEEE Access*, vol. 10, pp. 27591–27610, 2022. https://doi.org/10.1109/access.2022.3152787

[8] G. Heydari, D. Rahbari, and M. Nickray, "Energy saving scheduling in a fog-based IoT application by Bayesian task classification approach," *Turk. J. Electr. Eng. Comput. Sci.*, vol. 27, no. 6, pp. 4167–4187, 2019. https://doi.org/10.3906/elk-1902-152

[9] G. Kumar, R. S. Rathore, K. Thakur, A. Almadhor, S. A. A. Biabani, and S. Chander, "Dynamic routing approach for enhancing source location privacy in wireless sensor networks," *Wireless Networks*, vol. 29, no. 6, pp. 2591–2607, 2023. https://doi.org/10.1007/s11276-023-03322-8

[10] B. Kocot, P. Czarnul, and J. Proficz, "Energy-aware scheduling for high-performance computing systems: A survey," *Energies*, vol. 16, no. 2, p. 890, 2023. https://doi.org/10.3390/en16020890

[11] A. Katal, S. Dahiya, and T. Choudhury, "Energy efficiency in cloud computing data centers: A survey on software technologies," *Clust. Comput.*, vol. 26, no. 3, pp. 1845–1875, 2023. https://doi.org/10.1007/s10586-022-03713-0

[12] M. Boudmagh, A. Kerboua, and M. Redjimi, "Multimodal human action recognition for ubiquitous systems: Cross-attention of skeleton and audio," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 20, no. 5, pp. 70–86, 2026. https://doi.org/10.3991/ijim.v20i05.58381

[13] P. Li, X. Wang, K. Huang, Y. Huang, S. Li, and M. Iqbal, "Multi-model running latency optimization in an edge computing paradigm," *Sensors (Basel)*, vol. 22, no. 16, p. 6097, 2022. https://doi.org/10.3390/s22166097

[14] Y. Ma, Y. Zhao, Y. Hu, X. He, and S. Feng, "Multi-agent deep reinforcement learning for joint task offloading and resource allocation in IIoT with dynamic priorities," *Sensors*, vol. 25, no. 19, p. 6160, 2025. https://doi.org/10.3390/s25196160

[15] N. Mehran, Z. N. Samani, D. Kimovski, and R. Prodan, "Matching-based scheduling of asynchronous data processing workflows on the computing continuum," in *2022 IEEE International Conference on Cluster Computing (CLUSTER)*, 2022, pp. 58–70. https://doi.org/10.1109/CLUSTER51413.2022.00021

[16] D. Nikolow, R. Slota, S. Polak, M. Pogoda, and J. Kitowski, "Policy-based SLA storage management model for distributed data storage services," *Comput. Sci.*, vol. 19, no. 4, pp. 403–429, 2018. https://doi.org/10.7494/csci.2018.19.4.2878

[17] L. Pacheco, D. Rosário, E. Cerqueira, L. Villas, T. Braun, and A. A. F. Loureiro, "Distributed user-centric service migration for edge-enabled networks," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manag. (IM)*, 2021, pp. 618–622. https://dl.ifip.org/db/conf/im/im2021short/211092.pdf

[18] V. K. Pandey, S. Prakash, S. K. Jha, T. Yang, and R. S. Rathore, "An efficient and robust framework for IoT security using machine learning techniques," *Procedia Computer Science*, vol. 258, pp. 118–124, 2025. https://doi.org/10.1016/j.procs.2025.04.205

[19] G. Premsankar and B. Ghaddar, "Energy-efficient service placement for latency-sensitive applications in edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 17926–17937, 2022. https://doi.org/10.1109/JIOT.2022.3162581

[20] M. L. Gadebe and O. P. Kogeda, "Top-K human activity recognition dataset," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 14, no. 18, pp. 68–86, 2020. https://doi.org/10.3991/ijim.v14i18.16965

[21] D. Sahu *et al.*, "Optimizing energy and latency in edge computing through a Boltzmann driven Bayesian framework for adaptive resource scheduling," *Sci. Rep.*, vol. 15, p. 30452, 2025. https://doi.org/10.1038/s41598-025-16317-6

[22] A. Satouf, A. Hamidoğlu, Ö. M. Gül, A. Kuusik, L. D. Ata, and S. Kadry, "Metaheuristic-based task scheduling for latency-sensitive IoT applications in edge computing," *Clust. Comput.*, vol. 28, no. 2, pp. 1–17, 2025. https://doi.org/10.1007/s10586-024-04878-6

6 APPENDIX A

Algorithm 1: Proposed ALPS Algorithm

Input:
Mobile sensor task set ($\boldsymbol{T} = \boldsymbol{t_1}, \boldsymbol{t_2}, \ldots \boldsymbol{t_n}$), available execution nodes (N = {mobile, edge, cloud}), bandwidth (B), queue length (Q), resource capacity (R), latency threshold ($\boldsymbol{L_{th}}$)

Output:
Optimized task allocation and reduced total latency

1. *Start*
2. *Load incoming predictive task set T*
3. *For each task t_i in T do*
4. *Estimate task size D_i*
5. *Estimate computational demand C_i*
6. *Check current bandwidth B*
7. *Check queue length Q at mobile, edge, and cloud nodes*
8. *Estimate transmission latency $\boldsymbol{L_{tx}}$*
9. *Estimate queue delay $\boldsymbol{L_{queue}}$*
10. *Estimate execution latency $\boldsymbol{L_{exec}}$*
11. *Estimate response return delay $\boldsymbol{L_{return}}$*
12. *Compute total latency: $\boldsymbol{L_{total} = L_{tx} + L_{queue} + L_{exec} + L_{return}}$*
13. *If $L_{total} \leq L_{th}$ then*
14. *Assign task t_i to the node with minimum execution cost*
15. *Else*
16. *Apply adaptive PCA to reduce feature size*
17. *Recalculate computational demand C_i*
18. *Re-estimate total latency $\boldsymbol{L_{total}}$*
19. *Select execution node with minimum latency and available resources*
20. *End If*
21. *If selected node has sufficient resource capacity then*
22. *Allocate task ti to selected node*
23. *Update queue status and resource availability*
24. *Else*
25. *Select next best available node*
26. *Allocate task t_i accordingly*
27. *End If*
28. *End For*
29. *Return optimized task allocation and total latency*
30. *Stop*

7 AUTHORS

Professor Dr. Mursalim Nohong is the Dean of Faculty Economics and Business, Universitas Hasanuddin (UNHAS) Makassar, Sulawesi Selatan. He actively delivers lectures and training in entrepreneurship, SME finance, green financial management, and public sector financial and asset management (E-mail: mursalim-nohong@fe.unhas.ac.id).

Nora'asikin Abu Bakar is affiliated with the Faculty of Business Management and Professional Studies, Management and Science University, Selangor, Malaysia with a background in applied statistics and mathematics, while visible publications show involvement in quantitative studies related to engagement, achievement, and data analysis (E-mail: noraasikin_abubakar@msu.edu.my).

Siti Roshaida Abd Razak is affiliated with the Faculty of Business Management and Professional Studies, Management and Science University, Selangor, Malaysia. Her public scholarly record reflects interests in training transfer, employee job performance, organizational support, learning management systems, and performance-related studies. These areas of expertise are relevant to the present proposal because they strengthen the research design, organizational analysis, and interpretation of performance outcomes in data-driven mobile systems (E-mail: siti_roshaida@msu.edu.my).

Andi Nur Ildha Arfanita is an academic at the Faculty of Economics and Business, Hasanuddin University. Her research interest lies in economics, development studies, poverty alleviation, social spending, regional development, and SDG-related issues. This empirical and analytical background supports the proposal through its emphasis on data interpretation, model-based analysis, and the broader economic relevance of mobile data analytics capability (E-mail: ania.ildha@unhas.ac.id).

Helen Dias Andhini (SKM, ME), is pursuing her doctoral (PhD) studies there, as she is actively involved in advanced research projects typically conducted by doctoral candidates in the Faculty of Economics and Business, Hasanuddin University (E-mail: diasandhini21@gmail.com).