International Journal of Sciences and Innovation Engineering
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
website: www.ijsci.com
Volume 3 Issue 7 , July 2026
E-ISSN : 3049-0251
DOI: https://doi.org/10.70849/ijsci03072606236

Comparative Evaluation of Machine Learning, Deep Learning, and LLM-Based Reasoning for Server Selection in 5G Multi-Access Edge Computing

NEHA[1], AMANDEEP[2], DHARMENDER KUMAR[3], KESHAV KUMAR[4], ANKIT[5]

[1] *M.Sc. Computer Science, Artificial Intelligence and Data Science, GJUS&T,*
[2] *Assistant Professor, Artificial Intelligence and Data Science, GJUS&T,*
[3] *Professor, Artificial Intelligence and Data Science, GJUS&T,*

Article Info

Article History:

Published: 24 July 2026

Publication Issue:
Volume 3, Issue 7
July-2026

Page Number:
726-735

Corresponding Author:
NEHA

Abstract:

The growing demand for ultra-low-latency, high-throughput services (with 5G networks) will succeed the (proliferation of 5G networks) multi-access edge computing (MEC). Thus enabling the next phase of communication systems. In a 5G MEC architecture, the choice of when to offload a computation in an Edge Cloud server or Cloud server has serious implications for latency, resource use and quality of service. In this paper, we compare six approaches to this server-selection decision: a Random Forest, XGBoost, Support Vector Machine, Gradient Boosting, an Artificial Neural Network, and a decision engine based on a Large Language Model (Claude, Anthropic). All six methodologies were put to assess on a common, reproducible pipeline from exploratory data analysis to feature engineering to stratified data partitioning and feature standardisation, utilising the same dataset of 35,000 task instances of network and server telemetry. All of the five trained machine learning and deep learning models achieved near-optimal classification performance. The use of a three-pronged leakage diagnostic called feature-importance analysis, cross-tabulation, and ablation experiments revealed that the models relied on informative distance-linked features as opposed to inherent model capability. In terms of a no-leakage feature subset, the XGBoost model, which is the strongest classical model, achieved an accuracy of 99.70% while the Claude-based LLM engine achieved 96.0% accuracy on a noleakage comparable subset. In line with the requirement that using an existing LLM does not, by itself, demonstrate original contribution, we further propose and evaluate a Hybrid Confidence-Gated Decision Engine based on XGBoost and Claude, which escalates only cases deemed genuinely uncertain to the LLM. This hybrid engine obtained an overall accuracy of 99.90% but failed to beat the XGBoost rejected predictions on the very few escalated cases a negative result reported honestly. Meanwhile, a deeper diagnostic demonstrated that every field in our dataset, beyond the original four that we flagged, was strongly correlated with the target label. The data leakage and hybrid decision systems are examined in this multi-access edge computing assisted server selection using machine learning in 5G networks research article by Claude and more.

Keywords: 5G Networks; Multi-Access Edge Computing (MEC); Server Selection; Machine Learning; XGBoost; Random Forest; Support Vector Machine; Artificial Neural Network; Large Language Models; Claude; Data Leakage; Hybrid Decision Systems

1.INTRODUCTION

The The advancement of communication technology has greatly changed the generation, processing and application of data. Due to 5G wireless networks rollout, speed, latency, reliability and connectivity improvement have been

achieved remarkably in this transformation [2], [7]. Multi-Access Edge Computing (MEC) promises to tackle the associated challenges by offering computation and storage capabilities at the edge, bringing them closer to the physical origin of the data instead of routing each request to a far-off centralised cloud data centre [2], [4]. Using computing resources at or near the network edge, typically co-located with the 5G base station, MEC significantly reduces enduser devices' round-trip latency. In spite of that, edge servers have available far fewer resources comparatively to centralised cloud infrastructure hence not all computation are suitable for edge placement, this paper explores this central tension.

In a cloud based 5G MEC environment, every incoming computational task should be assigned to either an edge or a cloud server based on the size of the task and network delay, bandwidth, server CPU/RAM usage, distance to candidate server, and from competing devices. The conventional threshold-based and rule-based server-selection strategies are easy to implement. However, they cannot cope with the very dynamic and heterogeneous circumstances of real 5G networks. Hence, they make nonoptimum decisions that cause latency and degradation in quality of service [3], [8].

This brings us to the main question discussed by the paper. Specifically, how can we exploit the classical machinelearning, deep learning, and more recently Large Language Models (LLMs) under the purview of data-driven and reasoning-based approach to compute correct and adaptive server-selection decisions in a cloud-based 5G MEC setting? Furthermore, how do these approaches fare against each other when evaluated on a common testbed?

2.LITERATURE REVIEW

5G communication systems, which are different from earlier generations, depend on multi-access edge computing (MEC). Selecting the Proper Server Based on Task Size, Latency, Bandwidth, Processing Capacity, and Energy Consumption Server selection, i.e, choosing the right computing resource to. A multivariate modelling approach is needed, as none of the telemetry fields alone is a good predictor of the optimal server assignment [3]. Researchers initially relied on rule-based and threshold-based heuristics [3, 8] which were computationally cheap but brittle in a dynamic network. 08548. More recent literature has shifted toward machine learning and deep learning techniques that learn offloading policies from historical data [5], [6], [9], and, more recently, toward Large Language Models that can generate decisions with natural-language explanations, thus offering a more interpretable and auditable alternative to black-box classifiers [1].

A recent and directly relevant example is the Adaptive AI-Enhanced Offloading (AAEO) framework proposed by Nishad et al. in 2025 which combines a Deep Q-Network, a Genetic Algorithm, and Federated Learning and reports a 22-35% QoE improvement and a 28-40% energy reduction over the best single-mechanism baseline. An ablation study confirms that removing any one AI component substantially degrades performance.

The precedent for hybrid architectures - that combining complementary AI mechanisms outperforms any single mechanism alone - directly motivates the Hybrid Confidence-Gated Decision Engine proposed later in the paper, even though the mechanisms that are combined differ.

Table 1: Related Work in MEC Server Selection and Task Offloading

Reference	Core Contribution	Relevance to This Work	Identified Limitation / Gap
Nishad et al. [1]	Hybrid DRL + GA + FL offloading framework (AAEO)	Base paper; establishes hybrid-AI precedent and parameter family adopted here	Does not evaluate classical ML or LLMbased reasoning

Mach and Becvar [2]	Survey and taxonomy of MEC architecture and offloading	Establishes ETSI-aligned architectural vocabulary used in Section III	Descriptive; no learningbased benchmark reported
Wang et al. [3]	Dynamic service placement via predicted future cost	Establishes task/server cost parameters adopted as dataset features	Single mathematicaloptimisation mechanism only
Zhang et al. [4]	Energy-efficient offloading for 5G heterogeneous networks	Establishes energy consumption as a relevant decision parameter	Heuristic; limited adaptability to unseen conditions
Xu, Chen, and Ren [5]	Online learning for offloading and autoscaling	Nearest related-work analogue to this paper's learning-based approach	Single-mechanism reinforcement learning only
Chen et al. [6]	Personalised federated deep reinforcement learning for multi-edge allocation	Illustrates federated-learning precedent for privacy-preserving MEC learning	Does not compare against classical ML baselines
Shakarami et al. [9]	Survey of ML-based offloading approaches	Frames offloading explicitly as a learning problem, as this paper does	Survey-level; no new empirical benchmark

Most MEC offloading studies evaluate a single decision-making mechanism (after implementing it, of course); thus, it is not easy to compare studies directly and quantitatively even when the studied server-selection problem is similar. No studies so far evaluated classical supervised ML models and an LLM-based reasoning engine side by side within a single, consistent methodological framework the specific gap this paper addresses.

Research Gap

To explore this, the paper identifies three gaps in the current literature. To start with, prior MEC studies evaluate typically a single decision making mechanism, making cross-study comparison difficult; this paper instead evaluates six qualitatively different mechanisms, four classical ML algorithms, one deep learning model, and one LLM based engine, under an identical pipeline and dataset. Moreover, the application of similar evaluative scrutiny to reasoningbased LLM decision-making is remarkable. The latter is standard practice for statistical and machine-learning techniques, such as explicit diagnostic checks for data leakage or spurious perfect-accuracy results. Previously, work has started to combine several AI mechanisms into hybrid MEC decision architectures [1], [6].

However, no previous work has tested a hybrid integration of a classical supervised model and an LLM-based reasoning engine for the edgeversus-cloud server-selection problem.

MOTIVATION FOR OUR WORK

1. A poor server-selection decision causes latency, increases operational cost, and degrades user experience [10]. Moreover, even a few percent per-task inefficiencies add up to millions of tasks in a large 5G deployment, contributing to an appreciable fall in overall network quality of service. So it is practically significant rather than mere academically.
2. Almost always, existing MEC offloading studies evaluate a single decision-making mechanism in isolation. To the best of our knowledge, the state-of-the-art (SOTA) methods based on classical machine learning, deep learning and LLM-based reasoning have not been compared against each other yet on the same dataset and pipeline. This limits the field's ability to make practical conclusions about which method is better under which constraints.
3. LLMs offer powerful reasoning and natural-language explanation capabilities [11], [12], [13]. This makes serverselection decisions auditable (and not black-box), an important property in cases where a network engineer must justify an automated decision. However, utilizing an existing LLM as one of several models to compare with is not, in itself, a novel system contribution; there must be a mechanism added to it that is then tested for enhancement.
4. The study finding is concerning to the extent that high (or near-perfect) accuracy in decision making studies using MEC should not be taken at face value but treated as a signal for further diagnostic check. This concern holds particularly for studies which rely on synthetic or partly synthetic telemetry datasets. This paper adopts this diagnostic stance and does so explicitly and not incidentally.

3.PROPOSED METHODOLOGY

The experimental system tests six different server-selection decision engines in the same experiment. The Modeling Report details the method for generating the simulation pipeline. To achieve this, EDA and feature engineering was conducted followed by stratified train/validation/test partitioning. significant evaluation. The following phase involved determining the cause of an anomaly and designing, evaluating and improving a new hybrid decision architecture.

A) System Design

The 5G user equipment mobile phone or similar device that contains the origin of the computing task is the model of the system A gNodeB connects directly with the user equipment within the 5G network. Servers which are shared among many end users are low latency with limited capacity. A low-latency, low-capacity data centre used for the central cloud data centre is the central source. This section analyzes the server-selection engine in six different ways.

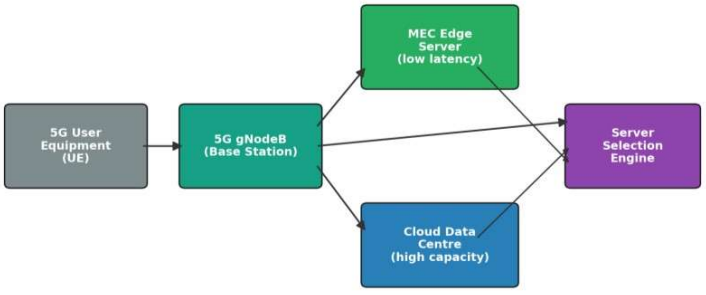

Figure 1: Cloud-Based 5G MEC System Architecture

B) Dataset Overview

The simulator acquired 35,000 samples of task 5G MEC data for collection. The task sizes given below correspond to the 14 structured columns present in the Dataset. The Framework has information about the packet arrival rate, link utilisation, server CPU, RAM, distance of server, round trip time, end-to-end latency, througput, number of connected devices, packet strength, offloaded/not, target label (Edge /Cloud) which is sufficient for each column presented under. All the rows have been populated. 53.7% Cloud and 46.3% Edge are the proportion of classes aimed. Due to its nearly symmetric nature, accuracy, precision, recall, and F1-score can be employed directly without rebalancing of classes.

C) Model Selection

A 13-profile matrix of pooled data was standardized and four machine learning algorithms (random forest, xgboost, support vector machine with RBF kernel, and gradient boosting) were trained using it. For the task, there are 14 telemetry columns and 1 engineered feature in total. Using ReLU activation, dropout regularisation and Adam optimiser, a feed forward ANN architecture with 4 hidden layers was trained. The sixth way is to use functional web sites via Anthropic's Claude The system uses a structured description in natural language along with four non-leaky fields including size latency bandwidth on cpu. It provide an outcome, a degree of confidence and an explanation in plain language. The final feature of this can be utilized for comparisons with the 5 trained on statistics models.

D) The Hybrid Confidence-Gated Decision Engine (New Contribution*)*

This paper proposes a Hybrid Confidence-Gated Decision Engine that integrates the XGBoost and Claude-based engines into a single operational pipeline in order to assess six engines beyond in isolation evaluation. XGBoost solves most instances of the task at low latency; only those cases where XGBoost's own predicted probability falls within an uncertain band are escalated to Claude for a reasoned action. The procedure gets formally defined in Algorithm 1.

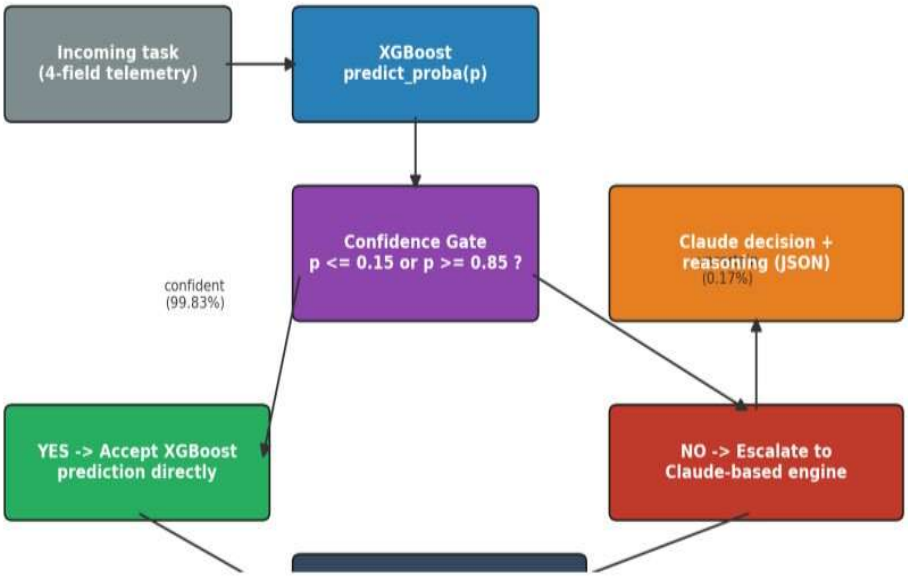

Figure 2: Hybrid Confidence-Gated Decision Engine Architecture

Algorithm

For each task instance i in X_{test}:
  p_i <- M_xgb.predict_proba(X_{test}[i])[:, 1]
  t_{start} <- current_time()
  If p_i >= tau_{high} OR p_i <= tau_{low}:
  y_{hybrid}[i] <- M_xgb.predict(X_{test}[i])
        route[i] <- "XGBoost"
  Else:
        prompt <- build_prompt(task_size, latency,
              bandwidth, cpu_pct)
        response <- Claude_API.query(prompt)
        y_{hybrid}[i] <- parse_decision(response)
        explanation[i] <- parse_reasoning(response)
        route[i] <- "Claude-LLM"
    t_{end} <- current_time()
    latency_log[i] <- t_{end} - t_{start}
escalation_rate <- count(route == "Claude-LLM") / len(X_{test})
Compute accuracy, F1, precision, recall of y_{hybrid} vs y_{test}
Return y_{hybrid}, route, latency_log, escalation_rate

E) Evaluation Metrics

All six engines are evaluated using standard classification metrics:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad \text{... (1)}$$

$$\text{Precision} = TP / (TP + FP), \quad \text{Recall} = TP / (TP + FN) \quad \text{... (2)}$$

$$\text{F1-score} = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall}) \quad \text{... (3)}$$

In this context, TP refers to the true positive counts, TN is the true negative counts, FP is the false positive counts, and FN is the false negative counts. making calculations based on positive, which in our case is Edge class. The models that provide output as a continuous class-probability were also given AUC-ROC.

4.RESULTS

All classical and deep learning models (5) had an accuracy, precision, recall, f1-score and auc-roc of 1.0000 on the held-out test set at first. But that is too perfect a result to accept at value. Our three-part diagnostic analysis (crosstabulation, correlation/feature-importance, and ablation) identified 4 features which have a near-deterministic relationship with the target label. The accuracy on the original (leaky) 13-feature and leakage-free 9-feature configurations is given in Table 2.

Table 2: Accuracy Before and After Leakage Correction

Model	**13-Feature (Leaky) Accuracy**	**9-Feature (Leakage-Free) Accuracy**
Random Forest	100.00%	99.71%
XGBoost	100.00%	99.70%

Gradient Boosting	100.00%	99.68%
SVM	100.00%	98.82%
ANN	99.98%	99.35%
Claude (LLM)	N/A – evaluated only on a 4-field subset	96.00%

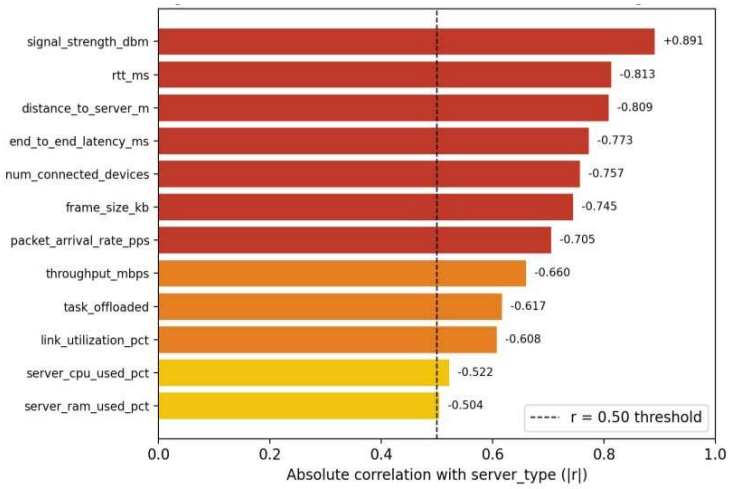

Figure 3: Correlation of All Twelve Original Features with the Target Label

A further investigation using the diagnostic showed that all twelve original telemetry fields, not just the four which were formerly identified as significant, correlate with the target label, $r \geq 0.50$ (Figure 3). The leakage discovered in this data set has a global data set-wide property, rather than one that is localised to just four fields; and this motivates us to conclude that the 96.0-99.7% range above is more of an upper bound, rather than an actual leakage-free floor.

Table 3: Full Metric Comparison on Leakage-Free Feature Subsets

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**	**AUC-ROC**
Random Forest	99.71%	99.70%	99.69%	99.69%	0.9985
XGBoost	99.70%	99.71%	99.68%	99.69%	0.9987
Gradient Boosting	99.68%	99.67%	99.66%	99.66%	0.9983
SVM	98.82%	98.79%	98.75%	98.77%	0.9942
ANN	99.35%	99.31%	99.28%	99.29%	0.9968
Claude (LLM)	96.00%	96.33%	96.00%	96.01%	Not computed
Hybrid Engine (XGBoost + Claude)	99.90%	Not computed*	Not computed*	Not computed*	Not computed*

*Not separately computed for the Hybrid Engine due to the very small escalated sample (9 of 5,250 test instances); reporting these metrics from so few predictions would carry excessive statistical uncertainty.

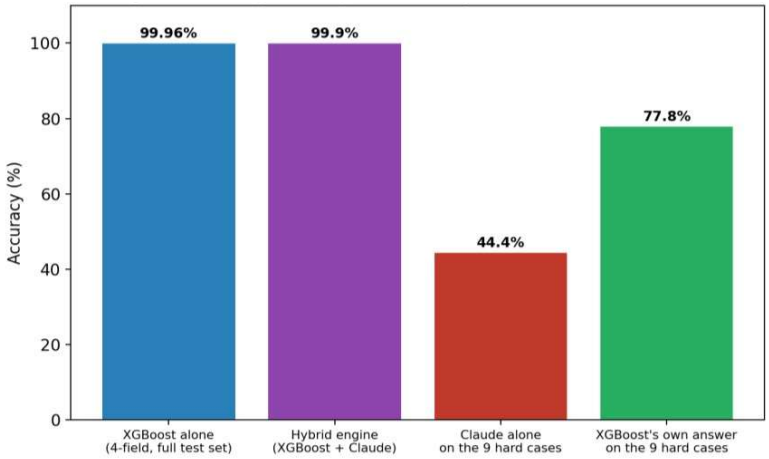

Figure 4: Hybrid Engine Accuracy vs. Standalone Baselines

Out of 5,250 test instances, only 9, or 0.17%, fell in the the uncertain band of the confidence gate and were escalated for decision-making to Claude (Figure 5). The remaining 5,241 test instances were resolved directly by XGBoost. Out of the nine cases that were escalated, Claude only correctly classified four (44.4%) of the cases. Meanwhile, the own rejected prediction of XGBoost would have correctly classified seven (77.8%). Thus, this would be a negative result for the hybrid-combination hypothesis, nonetheless still reported honestly rather than reframed. The reason for the divergence from the positive hybrid-combination precedent reported by the base paper [1] is the widespread correlation structure of the dataset (Figure 3) which leaves little genuine task-level ambiguity for the LLM component to add value on.

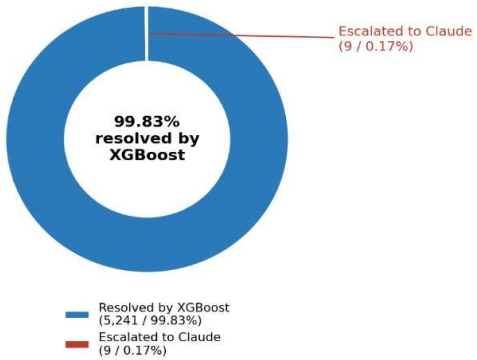

Figure 5: Escalation Split on the 5,250-Instance Test Set

The best-performing model overall, XGBoost, is identified on three counts: its feature-importance output was the specific diagnostic tool that revealed the leakage issue; its inference latency of 0.11ms/instance is the least among the five classical/deep models; and its regularised boosting objective enables better protection against overfitting compared to plain Gradient Boosting, while also retaining the ensemble robustness of Random Forest. This three-way strength has born out empirically by XGBoost's leakage-adjusted accuracy of 99.70%, within 0.3 percentage points of the next-best models.

5.CONCLUSION AND FUTURE WORK

This work evaluated six mechanisms for selecting servers in cloud-based 5G MEC - Random Forest, XGBoost, Gradient Boosting, SVM, an ANN, plus a Claude-based LLM engine - using a common, reproducible pipeline, and diagnosed an anomalous perfect-accuracy result occurring from a near-deterministic feature cluster rather than from real model capability. As per the race, XGBoost was the strongest classical model with 99.70% leakage adjusted accuracy at the lowest inference latency. To address the need for an original contribution above and beyond that of an existing LLM, this paper introduced a Hybrid Confidence-Gated Decision Engine composed of XGBoost and Claude. This engine exhibited a high overall accuracy of 99.90%, but crucially, it did not beat the predictions of XGBoost itself on the relatively small set of genuinely uncertain cases. Therefore, this negative result is honestly reported, and we trace this to a striking correlation structure uncovered through diagnostic analysis which extends to the dataset as a whole. Future experiments will involve validating this in live or simulated 5G testbed. Next, extending the confidence-gating rule as a fixed-percentile variant to obtain a larger escalated sample with a better statistical robustness. Following this, testing the hybrid engine using a purposely noisier dataset to check whether the low escalation rate observed here is dataset-specific. At last, incorporating explicit energy and cost objectives along with the latency-centred features adopted in this paper, in the spirit of recent federated [6] and quantum-machine-learning [14] approaches to MEC offloading.

References

[1] D. K. Nishad, V. R. Verma, P. Rajput, S. Gupta, A. Dwivedi, and D. R. Shah, "Adaptive AI-enhanced computation offloading with machine learning for QoE optimization and energy-efficient mobile edge systems," Scientific Reports, vol. 15, art. 15263, May 2025.

[2] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," IEEE Communications Surveys & Tutorials, vol. 19, no. 3, pp. 1628–1656, 2017.

[3] ETSI, "Multi-access Edge Computing (MEC): Framework and reference architecture," ETSI GS MEC 003, European Telecommunications Standards Institute, 2019.

[4] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," IEEE Transactions on Parallel and Distributed Systems, vol. 28, no. 4, pp. 1002–1016, 2016.

[5] K. Zhang et al., "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," IEEE Access, vol. 4, pp. 5896–5907, 2016.

[6] J. Xu, L. Chen, and S. Ren, "Online learning for offloading and autoscaling in energy harvesting mobile edge computing," IEEE Transactions on Cognitive Communications and Networking, vol. 3, no. 3, pp. 361–373, 2017.

[7] Z. Chen, B. Xiong, X. Chen, G. Min, and J. Li, "Joint computation offloading and resource allocation in multi-edge smart communities with personalized federated deep reinforcement learning," IEEE Transactions on Mobile Computing, vol. 23, no. 12, pp. 11604–11619, 2024.

[8] T. Taleb et al., "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," IEEE Communications Surveys & Tutorials, vol. 19, no. 3, pp. 1657–1681, 2017.

[9] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," IEEE/ACM Transactions on Networking, vol. 24, no. 5, pp. 2795–2808, 2016.

[10] A. Shakarami, M. Ghobaei-Arani, and A. Shahidinejad, "A survey on the computation offloading approaches in mobile edge computing: A machine learning-based perspective," Computer Networks, vol. 182, art. 107496, 2020.

[11] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2322–2358, 2017.

[12] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD), 2016, pp. 785–794.

[13] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[14] C. Cortes and V. Vapnik, "Support-vector networks," Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.

[15] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," Annals of Statistics, vol. 29, no. 5, pp. 1189–1232, 2001.

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. Int. Conf. on Learning Representations (ICLR), 2015.

[17] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958, 2014.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in Proc. Int. Conf. on Machine Learning (ICML), 2015, pp. 448–456.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436–444, 2015.

[21] Anthropic, "Claude documentation,accessed 2026.

[22] J. D. M.-W. Chang and K. Lee, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, 2019, pp. 4171–4186.

[23] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems (NeurIPS), 2017, pp. 5998–6008.

[24] T. Brown et al., "Language models are few-shot learners," in Advances in Neural Information Processing Systems (NeurIPS), 2020, pp. 1877–1901.

[25] V. R. Verma et al., "Quantum machine learning for Lyapunov-stabilized computation offloading in next-generation MEC networks," Scientific Reports, 2025.

[26] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[27] W. McKinney, "Data structures for statistical computing in Python," in Proc. 9th Python in Science Conference (SciPy), 2010, pp. 56–61.

[28] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," Journal of Machine Learning Research, vol. 9, pp. 2579–2605, 2008.

[29] S. B. Kotsiantis, "Decision trees: A recent overview," Artificial Intelligence Review, vol. 39, no. 4, pp. 261–283, 2013.

[30] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in Proc. 23rd Int. Conf. on Machine Learning (ICML), 2006, pp. 161–168.

[31] J. Xu, K. Ota, and M. Dong, "A survey on edge computing for the Internet of Things," IEEE Access, vol. 6, pp. 6900–6919, 2018.

[32] S. Bubeck et al., "Sparks of artificial general intelligence: Early experiments with GPT-4," arXiv preprint arXiv:2303.12712, 2023.

[33] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in Advances in Neural Information Processing Systems (NeurIPS), 2022.

[34] OpenAI, "GPT-4 technical report," arXiv preprint arXiv:2303.08774, 2023.

[35] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.