Workload Prediction for Efficient Node Management in Mobile Edge Computing

Efthymios Oikonomou*, Stefanos Plastras†, Dimitrios Tsoumatidis†, Dimitrios N. Skoutas†, and Angelos Rouskas*
*Department of Digital Systems, University of Piraeus, Piraeus, Greece
(oikonomouef, arouskas)@unipi.gr
†Department of Information and Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece
(s.plastras, d.tsoumatidis, d.skoutas)@aegean.gr

***Abstract*—The rapidly increasing number of mobile devices and resource-intensive applications poses substantial obstacles to traditional centralized mobile cloud computing, resulting in increased latency and decreased service quality. Edge computing, which places server capabilities at access nodes, provides a promising solution to the aforementioned issues. However, maintaining operational edge service nodes at each access node can be costly and inefficient. We propose and evaluate a scheme that combines a heuristic service node selection algorithm with machine learning based computational load prediction, with the goal of minimizing latency and balancing load among service nodes. Simulation experiments demonstrate that the proposed scheme substantially enhances system performance, paving the way for a more efficient and responsive edge network infrastructure, particularly in 5G and 6G mobile communication environments.**

***Index Terms*—Mobile Edge Computing, MEC, Machine Learning, Prediction, Workload Balance, 5G, 6G**

I. Introduction

The evolving landscape of telecommunication technologies and application domains is driving a shift in the resource access model for cloud computing services. To address the increasing need for low latency and enhanced quality of service (QoS), cloud resources are increasingly being distributed closer to edge users, resulting in faster response times and improved application performance. This trend has led to the emergence of the edge computing network architecture, as described in [1], [2].

The strategic deployment of edge servers poses a significant challenge in edge computing research [3]. Determining the optimal placement of these service nodes necessitates careful consideration of several factors, including network topology, operator policies, and user computational requirements. Furthermore, selecting the most suitable edge service nodes and assigning users to them demands consideration of resource availability, load balancing, service quality, and user preferences [4].

With the advent of 5G and Beyond 5G (B5G), user traffic and density are expected to surge. Predicting future workload becomes crucial to ensure efficient resource utilization and proactive management. Recent research highlights Artificial Intelligence (AI), and specifically Machine Learning (ML), as promising tools for predicting computing workloads in network clusters. By training models on available data, ML can provide valuable insights into future workload patterns [5], [6]. Predictive analytics architectures enable proactive resource allocation and management strategies, allowing the network to efficiently adapt to time-varying traffic demands. Precise workload projections allow operators to anticipate potential congestion and take preventive measures to maintain service quality. Supervised learning models, such as Support Vector Regression (SVR) [7], offer promising solutions for labeled predictions based on real-world network data.

Our research aims to tackle the challenge of optimizing latency and load distribution among service nodes in mobile edge computing, exploiting a computational load prediction procedure. More specifically, we propose a novel approach that integrates a service node management heuristic with a machine learning-based computational load prediction technique. The objective is to enhance the efficiency of edge computing systems by accurately predicting workload to assist the process of strategically placing service points to evenly distribute the workload across the edge network and realize low latency for user services. To the best of our knowledge, our contribution represents a significant advancement in this field, as we are not aware of any other research work that employs workload prediction for service node placement and workload balancing. Our results show that as the data set knowledge increases the prediction is more accurate and the decisions on placement of service points and workload distribution are more efficient.

The remainder of the paper is organized as follows. Section II provides an overview of the related work in the fields of workload balancing and workload prediction, in edge and cloud computing environments, respectively. Section III outlines the considered system model, while Sections IV and V present the two main components of the proposed edge management system: load balancing and workload prediction. In Section VI, we evaluate the proposed scheme under different simulation scenarios and discuss the results, while Section VII provides a summary of our findings and outlines our future research directions.

ISBN 978-3-903176-63-8 © 2024 IFIP

II. Related work

A. Balancing Workload and Reducing Latency

The primary challenges in edge computing are strategically placing or deploying nodes responsible for delivering edge services, as well as determining the optimal user distribution across edge servers. The authors in [8], suggested a method that places Cloudlets in a wireless metropolitan area. Additionally, they developed a user assignment strategy for these Cloudlets striving to reduce the total average response time in the system. The study presented in [9], addresses the challenge of placing edge servers aiming to achieve a balance in workload distribution among edge servers while minimizing the access delay for users interacting with these servers. This is accomplished through the integration of K-means and mixed-integer quadratic programming algorithms.

The authors in [10], proposed a rapid heuristic method in conjunction with a distributed genetic algorithm to address a QoS-oriented load balancing challenge among cloudlets within a Wireless Metropolitan Area Network (WMAN) in order to enhance the performance of mobile applications. In [11], the placement of cloudlets is established by the authors through a selection process from a set of potential locations, considering both the associated operational (OPEX) and capital (CAPEX) expenditures, along with the access delay incurred during task offloading. The work in [12] introduces the Avatar placement challenge within cloudlets and proposes a heuristic approach that considers both response time delay and energy consumption factors.

In [13], the authors presented the LAB load balancing strategy. Their approach involves allocating Internet of Things (IoT) devices to optimal fog nodes and base stations, aiming to reduce latency in the flow of IoT data. The goal is to reduce the latency of IoT data transmissions while considering both computational and communication delays. In research [14], the challenge of determining the optimal locations for edge servers is expressed as a multiobjective constraint optimization problem. This formulation aims to position edge servers in a manner that achieves a balance between their workloads and minimizes the access delay between industrial control centers/cellular base stations and the deployed edge servers.

B. Utilizing Machine Learning to Predict Workload

With the expected increase in user traffic and density, accurate workload prediction is crucial for proactive resource management and maximizing the efficiency of future networks. Recent research has highlighted ML methods as a promising prediction tool in modern data-driven networks [5], [6]. These approaches are capable of generating robust correlations between user behavior, node load, and time, thereby establishing intricate relationships among the diverse characteristics. Additionally, statistical methods have been employed as an alternative strategy for developing general load prediction models in cluster or cloud computing configurations utilizing time-series models [6], [15].

The study in [16] introduces a supervised learning scheme that utilizes an LSTM-based (long short-term memory) clustering-based workload prediction method to train separate models for each Virtual Machine (VM) cluster. By capturing the unique workload patterns of individual VMs, this method outperforms traditional approaches such as ARIMA and BRR and achieves up to 90% prediction accuracy for both Central Processing Unit (CPU) and memory workloads. In [17], the authors investigate the use of neural networks to predict future workload demands in cloud computing environments. They employ LSTM to predict the load over servers and evaluate its performance. The results demonstrate the effectiveness of LSTM in predicting workload patterns, enabling cloud providers to optimize resource allocation and mitigate adverse impacts such as service unavailability, energy consumption, and customer dissatisfaction.

The authors in [18] investigate the application of deep learning (DL) techniques for workload prediction in cloud computing environments. They emphasize the importance of data quality and uniformity in training DL models for accurate workload forecasts. They employ a comprehensive dataset to train and evaluate four DL models: recurrent neural networks (RNN), multilayer perceptrons (MLP), long short-term memory (LSTM), and convolutional neural networks (CNN). The results demonstrate that the LSTM model consistently outperforms the other models, exhibiting superior prediction accuracy regarding different error metrics.

Addressing the challenge of predicting host utilization in cloud data centers, the authors in [7] introduce a hybrid kernel-based SVR method for enhanced accuracy. The proposed method outperforms existing approaches in terms of error metrics, demonstrating its superior performance for VM migration and resource allocation. As research in this area continues to evolve, we can expect even more accurate and efficient prediction models to emerge, enabling significant advancements in resource management and network performance. Supervised learning models, such as SVR [7], offer robust solutions for labeled predictions based on real-world network data.

III. System Model

We consider a wireless cellular network architecture that employs edge computing principles to provide a distributed platform for executing various edge services closer to the user equipment (UE), enhancing network performance and reducing latency for the network operator's customers. This network topology features a deployment of M interconnected access nodes (ANs), represented as a connected graph G(V, E), with nodes $V = \{v_1, v_2, \ldots, v_M\}$ and network links E connecting the ANs. Nodes in the network can communicate with each other directly or indirectly through multi-hop paths.

To deliver edge services, the operator deploys service nodes (SNs) collocated with ANs. The actual number N of active SNs is proportional to computational demands from network users, allowing the operator to optimize energy consumption. The subset of operational SNs, is denoted as $S = \{s_1, s_2, \ldots, S_N\}$, where $S \subseteq V$, $|S| = N$ and $N \leq M$. When the number of available SNs is smaller than the total number of nodes in the network, it becomes essential to develop a

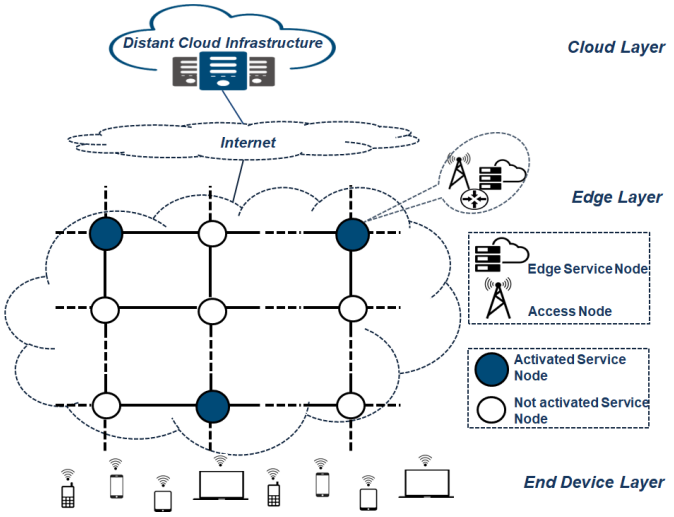

Fig. 1. Edge Computing network architecture: 9 Access Nodes with 3 operational Edge Service Nodes.

strategy for identifying which ANs will activate the hosted SN. Additionally, it is crucial to select the most appropriate SN for routing edge service requests from each remaining AN.

The edge user computational workload offered to AN v_j is denoted by the node weight w_j, representing the distributed number of users (also the computational needs) across the network and accessing edge resources through AN v_j. The cost of communicating or accessing edge services at an SN s_i hosted by AN v_k is represented by the weighted path communication cost w_j*d_{jk}, which includes latency and capacity costs of the intermediate links on the minimum delay path between v_j and v_k. In the context of our study, we assume that the required capacity to facilitate this communication is consistently available along the intermediate links of the specified path. Lastly, the computational capacity possessed by each SN is the same, under the assumption that all SNs share similar dimensions.

As our study does not focus on the management of the radio access network (RAN), we assume uniform wireless link characteristics between end-user devices and their corresponding ANs. Additionally, we consider all interconnecting transmission links (edges) within the graph to possess identical characteristics in terms of transmission capacity and latency. Therefore, the communication cost d_{jk} is solely determined by the number of connections (links) existing between nodes v_j and v_k within the shortest path. The system model, as depicted in Fig. 1, assumes a lower layer comprising end-user devices connected through wireless links to their nearest access nodes, through which they request edge services and resources. Out of the total nine (9) ANs, only three (3) currently have operational SNs.

A. Edge Management Module

The edge service node management module aims to optimize network performance and resource allocation at the network edge. Its primary functions are twofold: minimizing overall communication costs for users and ensuring fair and balanced workload distribution among service nodes.

In energy conservation scenarios, where the number of SNs is less than the number of ANs, the module strategically selects the optimal subset of ANs to host SNs. This selection is based on minimizing network response time for delivering edge services and achieving equitable workload distribution among active SNs.

B. Federated Learning: Minimizing Communication Overhead

To minimize communication overhead and enhance system efficiency, the management module leverages federated learning. Each node independently collects and analyzes its local traffic data, building an understanding of its own usage patterns. This information is then periodically aggregated at the management module without requiring constant communication from individual nodes.

This aggregated data allows the management module to accurately predict network-wide traffic patterns, dynamically select service nodes, and optimize network dimensioning. This approach aims to reduce both the system's response time when selecting SNs and the amount of data needed to be transferred for information gathering, leading to a more efficient and responsive network.

IV. Load Balancing Heuristics

Assuming a subset of N operational SNs to meet the computational demands given from all ANs, in this section, we present two heuristics, namely, the Forward Greedy and Reverse Greedy algorithms [19], [20] for the K-median clustering problem. These schemes are dedicated to identifying the optimal set of N SNs, aiming to minimize the overall communication cost incurred from each AN to its closest SN. Consequently, when a specific set of N SNs S, is selected, assigning ANs to their closest SN consistently results in the minimum possible communication cost. However, since these heuristics rely on distance, it becomes essential to incorporate load balancing into their decision-making process. If we define W_i as the sum of weights (w_j) of those ANs served by s_i that represent the computational workload on SN s_i, a straightforward measure for assessing the balance of the offered computational workload among the SNs is the sample variance of W_i's.

A. Load Balancing enhanced Forward Greedy heuristic

The Forward Greedy approach initiates with a single SN in set S that minimizes the sum of weighted path communication costs for all ANs in the edge network. Following, in each iteration, the number of SNs is gradually increased by one, with the selection of the SN v_k being the node that minimizes the potential increase in communication cost. This assumes that the all ANs will be allocated to their closest SNs within the set $S \cup \{v_k\}$. The process repeats until the total number of SNs reaches N. In cases where multiple choices of SNs exist, the algorithm employs the load balance criterion to guide its decisions. Specifically, it selects the SN v_k that, upon addition,

ensures that the remaining ANs are assigned to their closest SN in a manner that minimizes the sample variance of W_i's. We refer to this improved approach as the Load Balancing enhanced Forward Greedy heuristic (FGeLB).

B. Load Balancing enhanced Reverse Greedy heuristic

In contrast to the Forward Greedy scheme, Reverse Greedy initiates with M SNs by activating a single SN at each collocated AN. Then, in each iteration, the number of SNs is reduced by one, targeting the removal of the SN hosted at AN v_k that will have the least impact regarding the sum of weighted path communication costs towards the remaining SNs. Thus, the SN is removed, and this process continues until the number of SNs reaches N. When faced with multiple options of SN selection, the algorithm makes decisions according to the load balance criterion. Specifically, if more than one SN can be removed, the algorithm selects SN v_k in a way that reduces the sample variance of W_i's to a possible minimum. We call this heuristic Load Balancing enhanced Reverse Greedy (RGeLB).

V. Prediction of Service Nodes Workload

We propose using a Support Vector Regression model in conjunction with federated learning, as described at Section III, to optimize response time and minimize communication overhead in service node selection.

A. Rationale for SVR selection

SVR is an effective solution for predicting edge node workloads in real-time resource-constrained edge contexts due to its efficiency, accuracy, and data interpretability. Compared to complex models such as LSTMs, SVRs have a simpler structure that requires less training data and computational power, resulting in faster training and prediction times, which is critical for resource-constrained systems [21].

Furthermore, SVR performs well with smaller datasets, which are common in edge computing, by avoiding the overfitting problems that affect deep learning. This results in higher accuracy while maintaining a simpler model than standard deep learning approaches. Furthermore, SVR is more interpretable than complex neural networks, allowing for a better understanding of the factors that influence workload patterns. This allows for further model analysis and optimization to capture the unique features of edge node workload fluctuations [22]. SVR employs the following optimization function to balance model complexity and the fitting of the training data:

$$\min_{w,b,\zeta,\zeta^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*) \tag{1}$$

subject to the constraints:

$$y_i - \langle w, \phi(x_i)\rangle - b \leq \varepsilon + \zeta_i \tag{2}$$

$$\langle w, \phi(x_i)\rangle + b - y_i \leq \varepsilon + \zeta_i^* \tag{3}$$

$$\zeta_i, \zeta_i^* \geq 0 \tag{4}$$

where w is the weight vector, b is the bias term, $\phi(\cdot)$ is the feature mapping function, ζ_i and ζ_i^* are slack variables, C is the regularization parameter, and ε is the epsilon-insensitive loss parameter. To effectively train an SVR model for cluster workload prediction, the data undergoes a preprocessing pipeline. This pipeline involves scaling features to ensure consistency (data scaling), incorporating past values as additional features to capture temporal trends (timesteps), and selecting relevant features based on the prediction task.

Regarding model tuning, i.e., the process of fine-tuning the hyperparameters of a machine learning algorithm to enhance its performance for a specific task, SVR allows for the tuning of multiple parameters to improve model performance. The selection of kernel type (linear, polynomial, or radial basis function) significantly impacts the model's capacity to capture nonlinear interactions. The regularization parameter, C, balances the optimization of the margin and the reduction of training error, thus impacting the generalization of the model. The kernel coefficient, γ, dictates the influence of individual training examples on the decision boundary, hence affecting the model's flexibility. The ϵ parameter determines the acceptable error range, which affects the model's response to deviations from expected outcomes.

Furthermore, utilizing federated learning allows to reduce the communication overhead load and enhance efficiency. Each node independently analyzes its local traffic data to gain insights into its utilization. This information is then periodically transmitted to the management module, removing the need for continuous communication. By utilizing the collected data, the management module is able to predict network traffic patterns and efficiently select the service nodes.

VI. Evaluation Results and Discussion

In this section, we will evaluate the proposed scheme through two distinct scenarios. The first scenario focuses on the load balancing component and its associated heuristics, FGeLB and RGeLB. We will assess their effectiveness in distributing workload efficiently and ensuring system stability when the prediction model is properly trained and sufficiently accurate. The second scenario investigates the impact of user behavior shifts on traffic load forecasting accuracy. We will analyze how these shifts affect the scheme's performance and explore potential mitigation strategies.

A. Edge Network Topology and Workload Modeling

Building upon the system model section, we consider a specific edge network topology: a non-wrapped 10x10 symmetrical grid (Figure. 2). This grid comprises 100 ANs interconnected with eight neighboring ANs each, excluding those on the edges.

- *Sequence Numbers*: Numbers within the circles in Fig. 2 represent the sequence number of each AN.
- *Operational Service Nodes*: Represented by larger circles collocated with their corresponding ANs. The employed color scheme groups ANs associated with specific SNs for service access. For example, purple ANs 58, 67, 68,

69, 77, 78, 79, 87, 88, 89, 98, and 99 are served by the SN hosted at AN 88.

For the ML prediction process workload modeling, we leverage the GWA-T-12 dataset, sourced from Bitbrains. This dataset, recommended in recent studies [23]–[26], encompasses metrics from numerous virtual machines within a distributed computing environment. From this dataset, we selected data of 100 VMs to simulate the previously described network topology of 10x10 ANs. Bayesian optimization was employed to tune the SVR model, resulting in the following hyperparameters: regularization parameter (C=160), kernel coefficient (γ=15), epsilon-insensitive loss (ε=0.5), and an RBF kernel. Forecasting is conducted at specific time steps, each representing a five-minute period. Furthermore, by aligning with the practical goal of minimizing real-world operational costs, we focus on scenarios where only a limited number of SNs are active. Therefore, in the following scenarios, we consider a restriction where at most 15% of the ANs can host a SN.

B. Scenario 1: Evaluation of Load Balancing Heuristics

This scenario evaluates the effectiveness of the load balancing component and its heuristics, FGeLB and RGeLB, in a dynamic traffic environment. We examine their ability to distribute workload efficiently, achieve low latency for user requests and maintain system stability when the prediction model is adequately trained and accurate. However, as real-world traffic is constantly evolving, rendering historical data less reliable for prediction. While large datasets are valuable for training, their static nature doesn't reflect the dynamic traffic behavior we aim to explore. Therefore, we leverage the most recent 50% of the available data to train our prediction model, ensuring its relevance and accuracy in capturing dynamic trends.

Fig. 3 illustrates the total latency (total sum of weighted path communication costs) experienced by ANs towards their SNs for increasing number of service nodes for both FGeLB and RGeLB heuristics. For comparison, we also included the performance of the heuristics (FGeLB_Act, RGeLB_Act) when perfect knowledge of the computational load is available and therefore the actual weight values of the nodes are known. The latency results for FGeLB and RGeLB, both when prediction is employed and actual, showed a similar progressive decrease when more SNs become operational, with FGeLB scheme achieving better results especially in fewer number of SNs compared to RGeLB. Furthermore, the communication cost using predicted weight values closely matches the cost with actual values, suggesting a reliable prediction model.

Figure 4 depicts, for both schemes, the load of the most loaded SN compared to the mean load for increasing numbers of SNs. This serves as a metric for assessing load balancing effectiveness. Clearly, the FGeLB heuristic outperforms RGeLB, especially for smaller numbers of operating SNs. In Addition Fig. 5 illustrates the computational performance of both schemes. The FGeLB has a significantly lower runtime and outperforms the RGeLB, making it suitable for real-time applications where efficient load distribution and fast response times are critical.

C. Scenario 2: Changing Traffic patterns and Forecasting Accuracy

Building upon the findings of Scenario 1, which demonstrated the effectiveness of FGeLB in load distribution, Scenario 2 evaluates the latency performance of the same process in an environment of progressively changing traffic patterns, where the accuracy of traffic load prediction declines. In order replicate this challenge, we train our prediction model using 25% and 50% of the available data, respectively. For the purpose of comparison, we also assess the performance of FGeLB when the prediction is generated by utilizing the entire dataset (100%).

Table I clearly demonstrates the negative impact of decreasing training data on the prediction accuracy of the SVR model. The table shows a modest decline in performance metrics when reducing the training data from 100% to 50%. While all error measures (Mean Absolute Error - MAE, Root Mean Square Error - RMSE, and Mean Absolute Percentage Error - MAPE) increase slightly, the drop in R^2 is also small. This suggests that the model can still perform reasonably well with half the data, potentially offering a good balance between accuracy and resource efficiency. However, further reduction to 25% of the data leads to a significant drop in

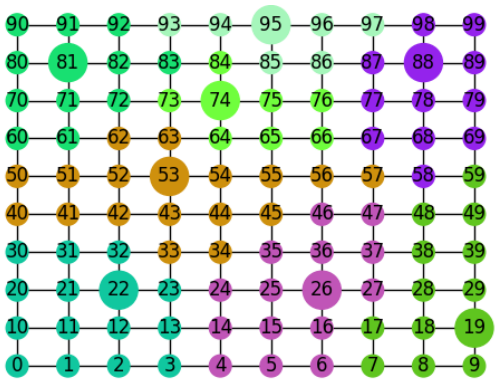

Fig. 2. Selection of service nodes and allocation of access nodes to service nodes.

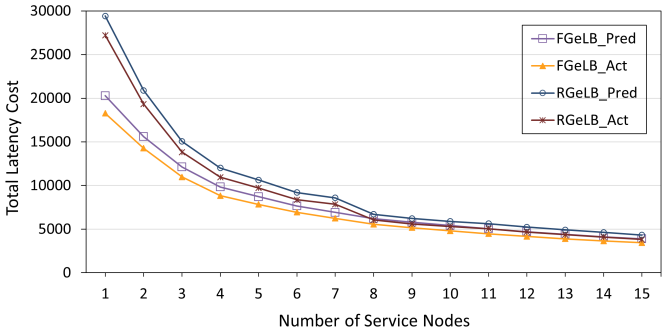

Fig. 3. Total Latency Cost (FGeLB, RGeLB).

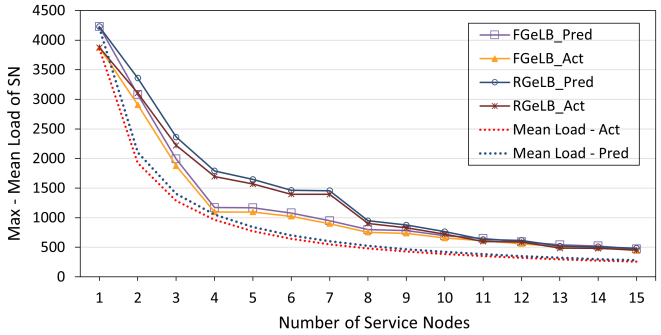

Fig. 4. Max - Mean Load of SN (FGeLB, RGeLB).

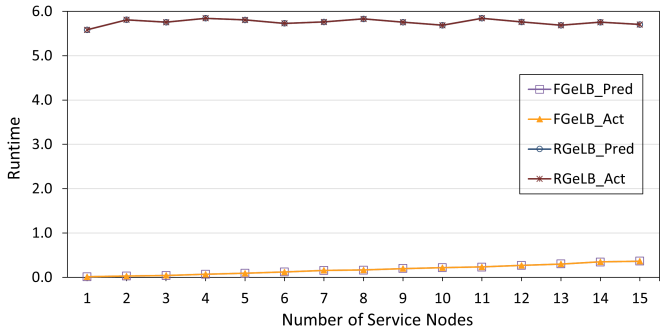

Fig. 5. Runtime (FGeLB, RGeLB).

performance, making predictions unreliable and implying a weaker relationship between the model and the true data.

TABLE I
EVALUATION OF PREDICTION ACCURACY
FOR PROGRESSIVELY CHANGING TRAFFIC PATTERNS

Dataset %	**MAE**	**RMSE**	**MAPE**	R^2
100%	1.56	1.87	1.84	0.97
50%	2.32	2.88	3.12	0.89
25%	8.16	6.32	10.00	0.47

Figure 6 depicts the dimensioning of the edge network, namely the selections of SNs and allocations of ANs produced by FGeLB, under different percentages of the dataset used. The numbers inside the circles indicate predicted (left column) and actual (right column) weights. As the percentage of the dataset used for prediction increases, the predicted weights better approximate the actual weights. Consequently, the network dimensioning based on predicted weights becomes closer to the dimensioning that would be obtained using the actual data weights. Notably, when the entire dataset is used, the dimensionings based on predicted and actual data are identical.

The decline of workload prediction accuracy negatively impacts the mobile edge network's performance, specifically with regard to latency. This effect is illustrated in Figure 7, which depicts the FGeLB's behavior in relation to network latency. By utilizing 25% of the dataset for the prediction process, the overall latency of a system that is dimensioned with predicted weights deviates by 24.55% to 28.70% in comparison to the

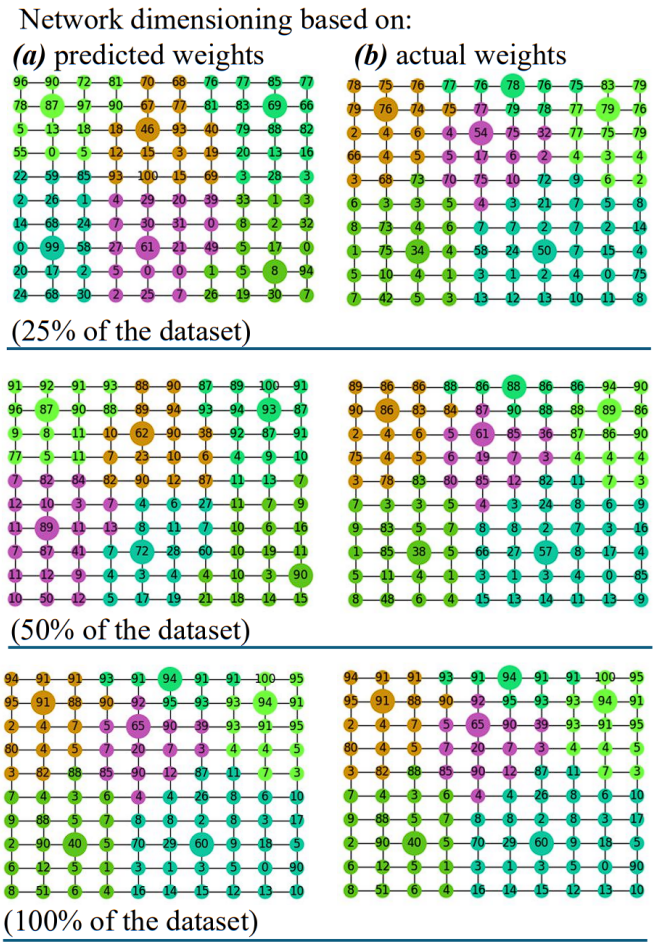

Fig. 6. Network Dimensioning for progressively changing traffic patterns (FGeLB).

system's total latency if the actual weights were utilized for dimensioning. The variance in total latency is reduced when 50% of the dataset is utilized in the prediction process to forecast workload weights and subsequently dimension the system. The observed values for total latency fall within the range of 9.96% to 11.66% compared to the latency if actual weights were used. Finally, when the workload of ANs is predicted using 100% of the actual data, there is minimal variation in the median values of the total latency differences, which range from 0.08% to 0.96%. Evidently, as the size of the dataset utilized in the prediction procedure expands, the system's dimensioning and behavior approach the anticipated values derived from perfected knowledge of workload data.

Finally, Figure 8 depicts the variance in the workload of the most loaded SN when the dimensioning of the edge network is based on predicted workload values compared to actual workload values. When 25% of the dataset is utilized, the algorithm shows varying levels of deviation, ranging from 9.85% to 22.28%, indicating a notable inconsistency. However, as the dataset size increases to 50%, there is a noticeable decrease in the variances, ranging from 5.24% to 8.36%. Lastly, with the

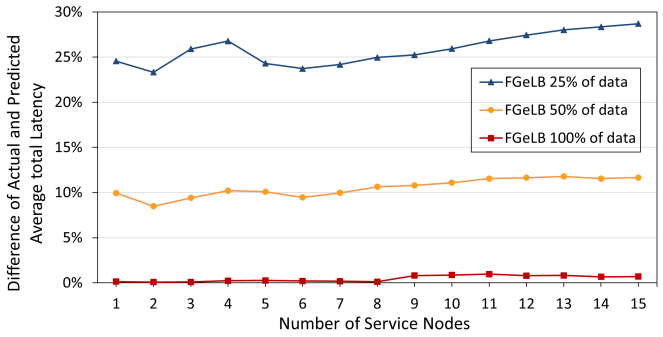

Fig. 7. Difference of Actual and Predicted Average total Latency (FGeLB).

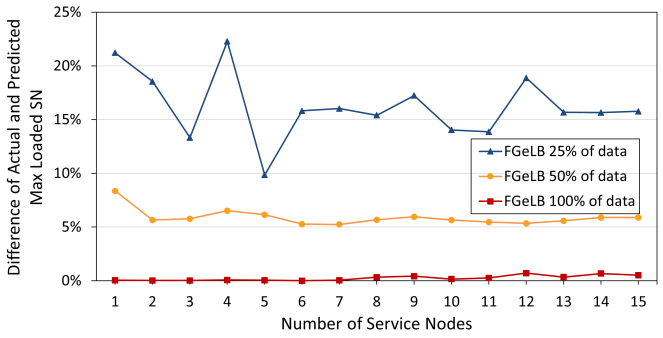

Fig. 8. Difference of Actual and Predicted Max Loaded SN (FGeLB).

full dataset (100%), the algorithm demonstrates remarkable accuracy, with variances approaching 0%.

VII. Conclusion

This paper explores the design and evaluation of an edge management module for a mobile edge network. The proposed approach leverages federated learning to minimize communication overhead and utilizes an effective load balancing technique (FGeLB) to ensure fair and efficient workload distribution among service nodes.

The study demonstrates that FGeLB outperforms RGeLB in terms of load balancing effectiveness and computational speed, making it suitable for real-time applications. Nevertheless, the system's performance is impacted by the precision of the computational load prediction model. As traffic patterns shift, the accuracy of predictions decreases, resulting in larger differences between predicted and actual network usage. Our upcoming research efforts will focus on determining the ideal balance between capturing current traffic patterns and obtaining enough data for accurate forecasts in dynamic environments.

References

[1] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.
[2] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.
[3] 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. 27 591–27 610, 2022.
[4] L. Kong, J. Tan, J. Huang, G. Chen, S. Wang, X. Jin, P. Zeng, M. Khan, and S. K. Das, "Edge-computing-driven internet of things: A survey," *ACM Comput. Surv.*, vol. 55, no. 8, dec 2022.
[5] M. Masdari and A. Khoshnevis, "A survey and classification of the workload forecasting methods in cloud computing," *Cluster Computing*, vol. 23, no. 4, pp. 2399–2424, Dec 2020.
[6] S. Kashyap and A. Singh, "Prediction-based scheduling techniques for cloud data center's workload: a systematic review," *Cluster Computing*, vol. 26, no. 5, pp. 3209–3235, Oct 2023.
[7] P. Nehra and A. Nagaraju, "Host utilization prediction using hybrid kernel based support vector regression in cloud data centers," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, Part B, pp. 6481–6490, 2022.
[8] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 725–737, 2017.
[9] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.
[10] M. Jia, W. Liang, Z. Xu, M. Huang, and Y. Ma, "Qos-aware cloudlet load balancing in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, vol. 8, no. 2, pp. 623–634, 2020.
[11] A. Ceselli, M. Premoli, and S. Secci, "Mobile edge cloud network design optimization," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1818–1831, 2017.
[12] X. Sun and N. Ansari, "Green cloudlet network: A sustainable platform for mobile cloud computing," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 180–192, 2020.
[13] Q. Fan and N. Ansari, "Towards workload balancing in fog computing empowered iot," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 253–262, 2020.
[14] S. K. Kasi, M. K. Kasi, K. Ali, M. Raza, H. Afzal, A. Lasebae, B. Naeem, S. u. Islam, and J. J. P. C. Rodrigues, "Heuristic edge server placement in industrial internet of things and cellular networks," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 308–10 317, 2021.
[15] A. Vashistha and P. Verma, "A literature review and taxonomy on workload prediction in cloud data center," in *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 2020, pp. 415–420.
[16] J. Gao, H. Wang, and H. Shen, "Machine learning based workload prediction in cloud computing," in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, 2020, pp. 1–9.
[17] M. P. Yadav, N. Pal, and D. K. Yadav, "Workload prediction over cloud server using time series data," in *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 2021, pp. 267–272.
[18] Z. Ahamed, M. Khemakhem, F. Eassa, F. Alsolami, and A. S. A.-M. Al-Ghamdi, "Technical study of deep learning in cloud computing for accurate workload prediction," *Electronics*, vol. 12, no. 3, 2023.
[19] D. Dohan, S. Karp, and B. Matejek, "K-median algorithms: theory in practice," *Princeton University*, 2015.
[20] O. Karaca, D. Tihanyi, and M. Kamgarpour, "Performance guarantees of forward and reverse greedy algorithms for minimizing nonsupermodular nonsubmodular functions on a matroid," *Operations Research Letters*, vol. 49, no. 6, pp. 855–861, 2021.
[21] V. Kramar and V. Alchakov, "Time-series forecasting of seasonal data using machine learning methods," *Algorithms*, vol. 16, no. 5, 2023.
[22] M. Tanveer, T. Rajani, R. Rastogi, Y. H. Shao, and M. A. Ganaie, "Comprehensive review on twin support vector machines," *Annals of Operations Research*, Mar 2022.
[23] Bitbrains, "GWT-12 Dataset," http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains, 2024, accessed: January 09, 2024.
[24] N.-M. Dang-Quang and M. Yoo, "Multivariate deep learning model for workload prediction in cloud computing," in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*, 2021, pp. 858–862.
[25] A. Iosup and D. Epema, "Grid computing workloads," *IEEE Internet Computing*, vol. 15, no. 2, pp. 19–26, 2011.
[26] S. Shen, V. Van Beek, and A. Iosup, "Statistical characterization of business-critical workloads hosted in cloud datacenters," in *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2015, pp. 465–474.