

Predictive Dynamic Virtual Machine Scaling for Federated Learning over Edge-Cloud Interworking

Sakshi Patni, *The Department of Computing, Gachon University, Seongnam, 13120, Rep. of Korea*

Sungpil Woo, *Electronics and Telecommunications Research Institute (ETRI), Daejeon, 34129, Rep. of Korea*

Joohyung Lee, *The Department of Computing, Gachon University, Seongnam, 13120, Rep. of Korea*

Abstract—With the expansion of data-generating devices, multi-access edge computing (MEC), has become more prominent in the Internet of Things (IoT) environment. The potential of MEC to enhance IoT devices with limited resources is examined in this research, with a focus on AI adoption scenarios. We propose a novel predictive virtual machine scaling method that dynamically balances local processing and cloud offloading for Federated Learning (FL) applications in MEC contexts. Our in-depth analysis of the VM scaling procedure highlights how it affects the computational burden and latency of FL services. Extensive tests on MNIST and CIFAR-10 datasets show the efficiency and flexibility of our method under different computing loads. Our technique performs better than benchmarks in terms of resource utilization, model convergence, and straggler mitigation. Our method maintains 95.54% resource utilisation efficiency across a range of workloads, while achieving a 20.73% decrease in total latency for FL jobs. Its adaptability in FL settings is shown by its strong performance on the more complicated CIFAR-10 dataset (91.73% accuracy) and the MNIST dataset (95.34% accuracy).

Index Terms: Federated learning, Edge-cloud computing systems, VM scaling, Scalability, Multi-access edge computing

Introduction

In the rapidly evolving Internet of Things (IoT) landscape, the convergence of massive data generation and artificial intelligence (AI) adoption presents unprecedented opportunities for innovation and knowledge creation. Distributed multi-access edge computing (MEC) emerges as a crucial support system for resource-constrained IoT devices, serving as an intermediary for efficient data processing and cloud connectivity.¹ The distributed architecture of MEC servers, managed by various service providers, creates an ideal environment for FL. This approach allows parallel training of machine learning models across multiple MEC servers while preserving data privacy by ex-

changing only model parameters through a centralized cloud server.² In the FL process, MEC servers perform local training on data from connected IoT devices, sending these local model updates to a cloud server for aggregation, thus completing the FL training cycle. This iterative process updates the FL model on each MEC server, which also handles inference tasks. The simultaneous execution of FL training and inference, collectively termed 'FL services,' imposes significant computational demands on MEC servers, which have limited resources compared to cloud servers. Recent MEC platform developments utilize containers (e.g., Kubernetes) or VMs for application deployment. Given the resource constraints of MEC servers, implementing advanced scaling algorithms, including both horizontal and vertical scaling, is critical. MEC can dynamically manage VM instances through node controllers (NC), adjusting CPU allocations to meet fluctuating

demands. As computational demands for FL services continue to rise, there's growing interest in AI-driven predictive auto-scaling of VMs on MEC servers.³ This approach aims to mitigate potential delays in VM scaling and ensure quality of service (QoS) for FL services. While previous studies in FL management typically assumed fixed computational resources and focused on hyperparameter optimization,⁴ this article advocates for tailored automatic VM scaling on MEC servers specifically for FL services. This approach recognizes and addresses the unique challenges posed by the evolving and critical domain of federated learning in edge computing environments.

These distinctions stem from several factors:

- **FL training latency:** In synchronized FL, which is the most commonly used approach, each local training session must meet the specified latency requirements for each round. If it does not, the local update should be ignored.
- **Straggler issue:** FL participants are required to wait until the slowest ("straggler") completes their local training. This waiting period can lead to idle time for other participants, resulting in the wastage of computing resources or slower convergence rates.
- **FL inference variability:** There is a wide range of inference services, each with its own latency requirements. For example, services such as augmented reality demand extremely stringent latency, sometimes as tight as 16.6 ms.⁵

While the potential benefits of this solution are significant, it is important to note the complexity involved in predicting future demands and implementing automatic VM scaling that meets the intricate requirements of simultaneous FL inference and training tasks. The challenge becomes even more pronounced when some MEC servers experience overload. To mitigate these issues, we also explore the possibility of jointly managing computation offloading for FL inference tasks and VM scaling, particularly in scenarios where MEC servers are overloaded, taking into account the interworking between MEC and cloud infrastructures. Such complexities and challenges drive us to summarize the key issues and potential solutions for developing innovative auto-scaling VMs on MEC servers in this article, with the aim of inspiring future research in this field. The contributions of this article can be summarized as follows:

- We provide a thorough examination of the automatic VM scaling process on MEC servers for FL services interworking with the cloud server.

It includes an in-depth analysis of how FL services impact VM scaling, specifically in terms of latency and computational load.

- Building upon this analysis, we present an innovative method for predictive VM scaling. This approach is designed to fulfill the latency demands of FL services while aiming to reduce over-provisioning to a minimum.
- Finally, we identify and discuss several potential avenues for future research in this area, highlighting the next steps in advancing this field.

Why VM scaling for FL services is important?

Background about VM scaling in MEC servers

In MEC-cloud interworking systems, FL services like training and inference demand substantial computational resources from MEC servers with strict latency needs. As FL demand grows from IoT devices, more sophisticated VM scaling on MEC is crucial. Existing work, as illustrated in Figure 1, focuses on VM scaling in clouds for general web traffic like HTTP requests, differing from computation offloading for FL. While web traffic is less computationally intensive, FL requires intensive computation and low latency. MEC servers placed closer to devices should play an active role to meet FL latency needs. Also, given MEC's limited capacity, cloud servers should assist by processing offloaded tasks, especially for FL inference.

Study on VM scaling A substantial amount of research has been dedicated to VM scaling. In the field, two primary VM scaling techniques are recognized: vertical scaling and horizontal scaling.

- **Vertical scaling:** It involves adjusting the CPU and memory resources allocated to a specific microservice. This modification allows for the enhancement or reduction of individual components within the microservice architecture, providing flexibility in adapting to varying workloads or performance requirements. It is particularly useful in scenarios where specific microservices experience fluctuations in resource needs, enabling a more efficient allocation of computing resources to meet dynamic requirements.
- **Horizontal scaling:** It adjusts the number of VM instances dedicated to a microservice. It enables organizations to efficiently handle changing workloads by increasing or decreasing VMs. This adaptability ensures optimal performance

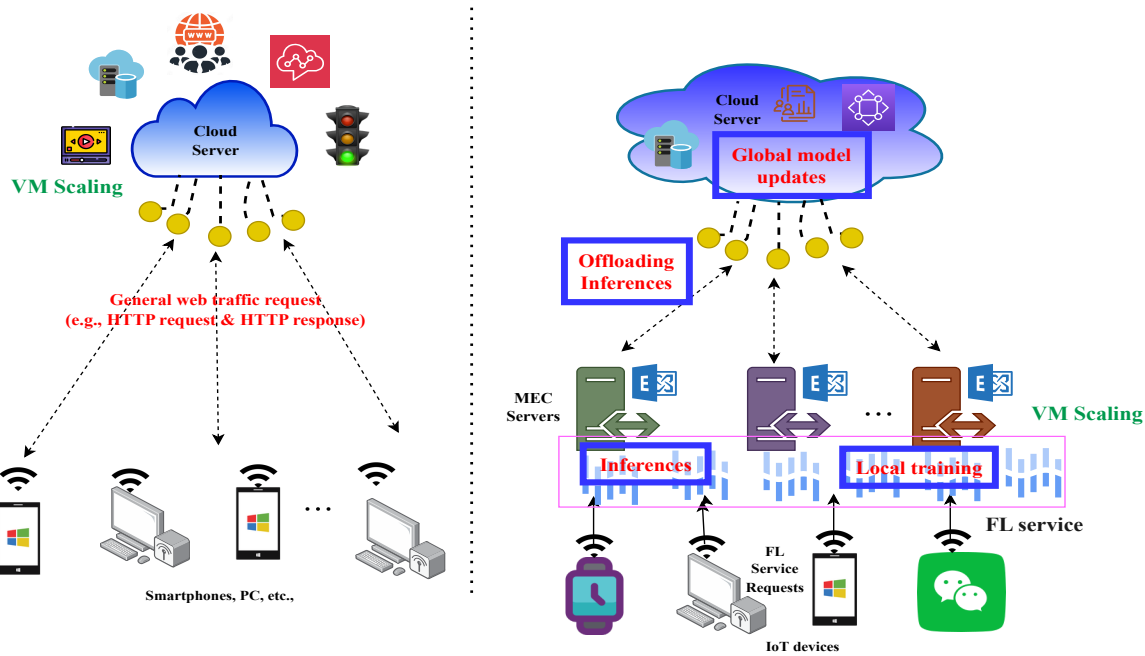


FIGURE 1: Traditional centralized cloud systems (left) contrast with edge-cloud solutions (right), including offloading during MEC server overload and dynamic VM scaling for FL

and resource efficiency in FL services during collaborative and decentralized ML tasks.

Different scaling mechanisms are investigated in recent FL research. By allowing for more parallelism, horizontal scaling may hasten the convergence of global models. Vertical scaling reduces stragglers and improves the performance of each individual client. Adaptive horizontal scaling based on convergence rates is suggested by Gao et al.⁶ Vertical scaling is introduced by Kai et al.⁷ in order to relieve bottlenecks. A hybrid strategy integrating both strategies is presented by Wan et al.⁸ in order to adjust to evolving FL workload characteristics. To satisfy particular FL needs, the best strategy often combines the two approaches. Several research on VM scalability and resource allocation in edge computing settings are compared in detail in the Table 1, with an emphasis on how well they apply to FL services. It draws attention to the distinctive features of our suggested methodology, which distinguishes it from other approaches in the area by explicitly addressing FL issues, using both horizontal and vertical scaling, utilizing deep learning for prediction, and including offloading tactics. Recent studies have explored various auto-scaling approaches for edge and cloud environments. Ray et al.¹¹ introduced a safe reinforcement learning-based policy for MEC to address latency in service deployment. Al-Dulaimy et

al.¹² proposed a multi-loop approach for cloud applications, dynamically adjusting VM resources based on SLA requirements. Heidari and Buyya [879] presented algorithms leveraging resource heterogeneity in cloud environments for efficient horizontal VM scaling. Phyu et al.¹⁵ focused on secure traffic forecasting in mobile networks for VM resource adjustment. The Snowball framework¹⁶ introduced incremental model update compression in FL, reducing energy consumption in uplink communication. Despite significant research in heuristic and predictive VM scaling, there remains a lack of explicit consideration for FL-specific requirements, particularly local training latency and straggler issues. This gap limits the applicability of existing approaches in FL scenarios, where these challenges are crucial for effective performance. Our work aims to address these FL-specific concerns in VM scaling.

Our approach uniquely incorporates both horizontal and vertical scaling for FL services, leveraging a predictive CNN model to anticipate resource demands. Based on anticipated FL workload characteristics, this hybrid approach dynamically modifies the number of VMs (horizontal scaling) and the resources allotted to each VM (vertical scaling). Combining these techniques allows us to effectively manage abrupt increases in client engagement as well as the fluctuating computing needs of different FL activities, all while

TABLE 1: Summary and Comparison of VM Scaling Studies for Edge Computing and FL Services

Study	Key Contributions	FL Consideration	FL-Specific Features	Scaling Type	Predictive Method	Offloading Consideration
9	Adjust scaling parameters to effectively manage the dynamic fluctuations incoming requests	No	N/A	Horizontal Scaling	ML Method	No
10	Forecasts the forthcoming count of Virtual Network Function instances by analyzing the anticipated traffic demand	Yes	Limited to VNF scaling	Both	Deep Learning	No
3	Comparative study analyzing the performance of diverse ML-driven models for predicting cloud workload	No	N/A	No	Workload Forecasting	No
11	Adaptive auto-scaling policy agent tailored to meet service-specific latency requirements amid changing traffic patterns	No	N/A	Horizontal Scaling	Reinforcement Learning	No
12	Optimally allocate resources to VMs in accordance with SLA requirements and real-time conditions	No	N/A	Horizontal scaling	Time-series forecasting	No
13	Automatically optimize the quantity and types of VM based on computation needs for convergent graph applications, enhancing performance and reducing costs	No	N/A	No	Graph and Map Reduce	No
14	Proposes a multi-dimensional contract-based incentive mechanism and Stackelberg game for resource allocation in edge-fog networks. Considers task offloading and resource sharing among edge nodes.	No	N/A	No	Contract Theory, Stackelberg Game	Yes
Proposed	Predict VM scaling and meet latency requirements while minimizing over-provisioning for FL services	Yes	FL training latency, Straggler mitigation, Privacy-aware	Both	Deep Learning	Yes

preserving stringent latency standards and optimum resource use.

Background of FL services In Federated Learning (FL), maintaining low latency is crucial, especially in synchronized FL where local training must meet round-specific requirements. FL training latency, representing the time for local training completion in each round, is critical for timely model updates across distributed MEC servers. The straggler effect, where slow participants hinder overall convergence, poses a significant challenge, necessitating strategic client selection, resource allocation, and dataset management. FL services also involve distributing the trained global model for inference at MEC servers, adding complexity due

to varied latency needs across applications. These range from augmented reality (≤ 20 ms) to healthcare (100-250 ms for remote robotic surgery) and intelligent connected vehicles (20-100 ms for 5G automotive vehicles). It's important to note that FL's training and inference stages are separate and sequential. During training, devices process data locally and transmit updated models to a central server. The inference stage occurs independently after model distribution. Our method doesn't alter the FL model structure but impacts resource availability for local computations through VM scaling. FL presents three distinct challenges for VM management: (a) Managing varied client capabilities (b) Synchronizing across dispersed clients (c) Adhering to strict privacy requirements limiting data

sharing Our VM architecture specifically addresses these FL-centric issues, ensuring efficient resource allocation while maintaining the integrity of the FL process.

Overall process of predictive VM scaling

Our scaling strategy optimizes resource allocation for FL local computations, dynamically adjusting VM resources to accommodate more devices or allocate increased computational power. We carefully consider the effects of resource scaling on model convergence and accuracy to align with FL principles. Figure 2 illustrates the predictive VM scaling process in MEC servers for FL services. IoT devices send datasets for training and request inference services, with MEC servers adjusting VM instances to meet latency demands. When necessary, tasks are offloaded to cloud servers, balancing resource constraints against transmission latency. Our approach optimizes resource allocation across this distributed system, considering both MEC capabilities and cloud offloading potential. It's tailored for FL by implementing secure aggregation, dynamically adjusting resources to mitigate stragglers, and adapting to heterogeneous client devices. Our method especially fits FL by: (1) using secure aggregation methods to protect privacy during virtual machine scaling, (2) dynamically changing VM resources to minimise stragglers and preserve synchronisation, and (3) adjusting to heterogeneous client devices by flexible resource allocation.

Consider a system with N different types of FL services. Each VM on the MEC server is dedicated to a specific task (e.g., local training or inference) within an FL service. The MEC server hosts multiple VMs simultaneously, enabling it to manage various FL services concurrently as requested by IoT devices. To enable predictive VM scaling on MEC servers, the detailed steps are outlined below:

- Step 1) Task Categorization: Every local training task and inference task have their own computing demand to meet specific latency requirements. This necessary computing demand should be categorized to aid in the prediction of future computing needs.
- Step 2) Development of the Prediction Model: Leveraging a deep learning model, this step involves training the prediction model using historical request patterns. The aim is to accurately forecast the future requests for each VM.
- Step 3) Predictive VM Scaling: In order to max-

imise VM scalability, this step entails modifying the computing resources for every VM in accordance with the projections generated by Step 2. It also entails outsourcing certain inference jobs to the cloud server, taking into consideration local resource limitations as well as network latency. To achieve this goal, it also includes offloading certain inference tasks to the cloud server, which helps mitigate issues caused by stragglers and prevents server overload at the MEC. It is important to note that to reduce idle times owing to straggler effects, the local training times across VMs for the same FL service should be closely aligned. Furthermore, the strategy aims to minimize the overprovisioning of VMs.

Novel predictive VM scaling for FL service

The process of FL services involves a diverse ecosystem comprising FL inferences, FL training requests originating from IoT devices. For developing the predictive model, the Convolutional Neural Network (CNN) is employed, renowned for its effectiveness in capturing spatio-temporal patterns.¹⁷ This CNN model processes historical and current resource usage data such as $(num_{clients} \times time_{steps} \times metric_{features})$ obtained from MEC servers. To dynamically optimize the network weights, the Auto Adaptive Differential Evolution (AADE) algorithm is used. This involves a series of operations such as initialization, evaluation, mutation, recombination, and selection. The fitness function Mean Squared Error (MSE) is applied to evaluate the networks and minimize the objective of accurate machine resource prediction for the upcoming requests. It is important to note that when dealing with MSE, accuracy is inversely related to the value of MSE. This allows accurate forecasting of future workload requests for each VM on MEC servers. The predictions drive proactive auto-scaling to adjust VM resources, meeting latency targets while minimizing over-provisioning. Offloading to cloud server handles excess requests, preventing overload. Overall, the novel approach include task categorization, develops a CNN model, and leverages forecasts to scale VMs predictively, aiming to optimize resource efficiency, service latency, and straggler mitigation for supporting FL services. Figure 3 shows the working for predictive VM scaling in a MEC system. The process begins by initializing the parameters, datasets, devices, and MEC models.

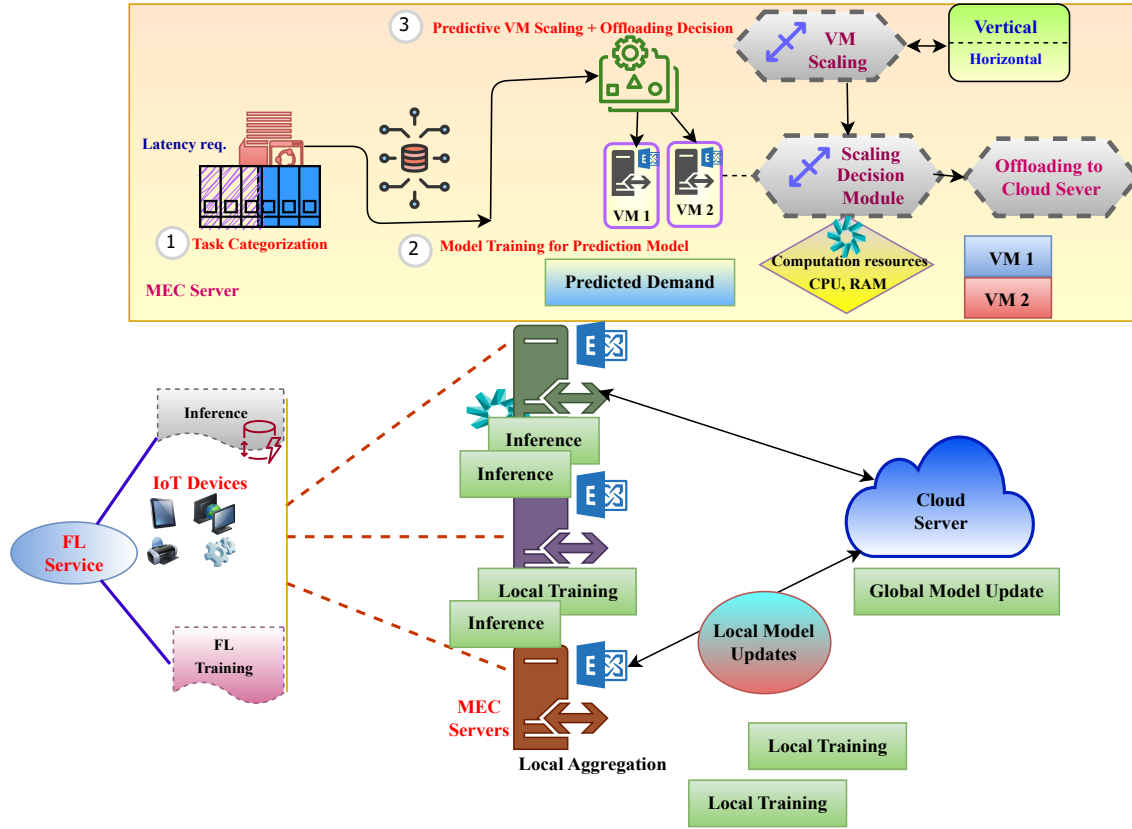


FIGURE 2: Edge computing distributes intelligence across devices, multi-access edge servers, and the cloud. Nearby edge servers aggregate updates for global model improvement, scale resources based on demand, and offload to the cloud when overloaded

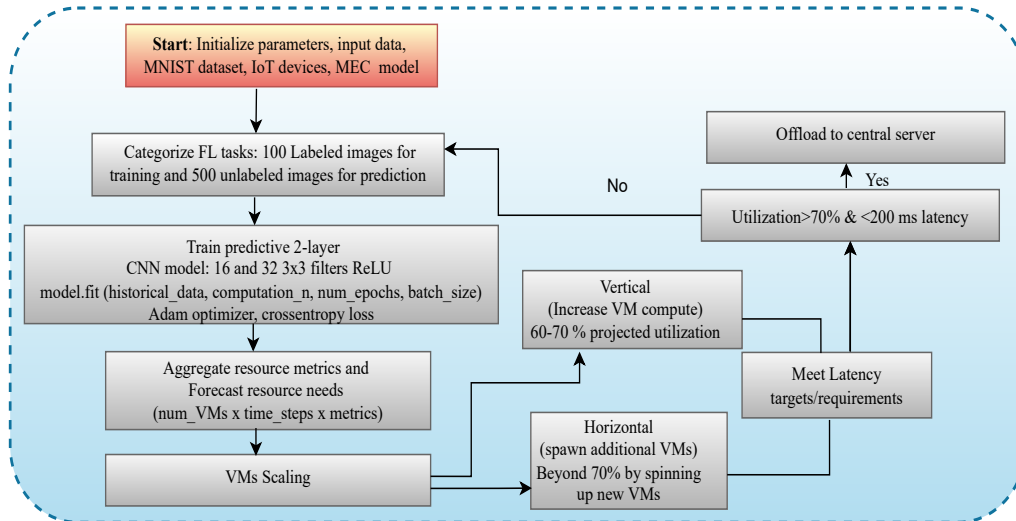


FIGURE 3: Working of Predictive VM scaling

CNN-based Predictive VM Scaling

A tensor of shape (numclients, timesteps, features) including information on CPU, memory, and network traffic is fed into our CNN model. A forecast of the resources required for the next time step is the output. Because CNN can effectively capture both spatial (across clients) and temporal patterns which is important for FL workloads - we choose it over LSTM. Our CNN design comprises of max pooling and a dense layer after two 1D convolutional layers (16 and 32 filters, 3×3 kernel). Our CNN was adjusted for certain deployments after being pre-trained on a broad dataset covering a range of FL situations. This method keeps the model's prediction accuracy high while allowing it to adjust to changing numbers, capacities, and tasks. The CNN model that we employ to estimate demand is an element of our VM scaling system, not the FL process itself. It uses non-sensitive, aggregated information about resource utilisation trends among FL clients to function.

Case Study: An Illustrative Example

In our FL setup, tasks are categorized based on FL services, with devices contributing to both inference and training requests. We use the MNIST dataset, which consists of standardised, flattened 28×28 pixel pictures with 784 characteristics. For local training, each client VM utilises 100 labelled pictures, and for future computing requirements, it requires 500 unlabelled images. Two convolutional layers (16 and 32 3×3 kernels) with 2×2 max pooling and ReLU activation are characteristics of our CNN model. A different CNN predictor, trained on aggregated resource measurements, makes predictions about communication and computation requirements. It has a regression output layer and its input tensor dimensions are $(num_{clients} \times time_{steps} \times metric_{features})$. The model uses 32-sized mini-batches with categorical cross-entropy loss to train across 50 epochs. It is constructed in TensorFlow using the Adam optimiser. Every week, it is retrained using new metrics from the most recent FL rounds. We evaluated generalisability while retaining 90% of the original prediction accuracy over a range of situations (10-1000 devices, varied complexity). Our approach optimises resource allocation for FL processes, while not being FL-based. We measured improvements in FL training time and model convergence rates as a consequence of resource allocation optimisation to assess its efficacy.

The proposed predictive scaling system utilizes both horizontal and vertical scaling to dynamically meet resource needs for FL workloads. In horizontal

scaling, the additional VMs are spun up to handle increased future demand predicted by the models. This allows faster training and inference by parallelizing across more machines. Horizontal scaling kicks in when the estimated resource utilization exceeds 70% of current capacity or if training time is projected to violate the latency target of 200 ms for local FL rounds. Adding nodes parallelizes work to meet the latency bounds.

In vertical scaling, for smaller surges, existing VMs scale up in compute power. For example, CPU cores or RAM are increased per VM. This occurs when predicted utilization is between 60-70% of current resources. Vertical scaling supports faster local FL without additional network overhead. If even after scaling, the local devices cannot meet the latency target, the model triggers offloading to the central servers for that round. Training latency across clients is monitored and modeled to predict stragglers. Additional replicated slots are launched to balance slower VMs. This redundancy brings their training time in sync with the median. The criteria above aim to meet FL latency goals while minimizing resource over-provisioning via smart horizontal/vertical decision making. The model outputs 168-hour forecasts. With a cluster size of 40 VM clients, sample dimensions would be (40 VMs, 168 hours per week, 2 metrics of computation and communication). This model facilitates appropriately provisioning capacity to meet needs within latency bounds.

Use cases and performance evaluation

In a MEC system with orchestrated FL services on diverse edge servers, a predictive mechanism forecasts task times. Server A has 180ms training, 22ms inference; B has 120ms training, 50ms inference; C has 150ms training, 18ms inference. The auto-scaler foresees delays on B. It reactively provisions VMs and vCPUs to mitigate delays. On B, it horizontally scales adding 2 VMs to address inference delays. On A, it vertically scales vCPUs from 4 to 8 to mitigate training stragglers. This dynamic scaling balances resources and optimizes FL in MEC. Experiments evaluate the predictive auto-scaler on MNIST for demand forecasting, server utilization, latency, and QoS probability in training. The experiments utilized a server equipped with an Intel Core i5-9980HK CPU operating at 2.40 GHz and 16 GB of RAM, scaling CPU by 100 millicore increments to a max 1000 millicores. This paper is compared with two approaches: Ensemble Empirical Mode Decomposition (EEMD),¹⁸ a resource demand prediction method using ensemble empirical

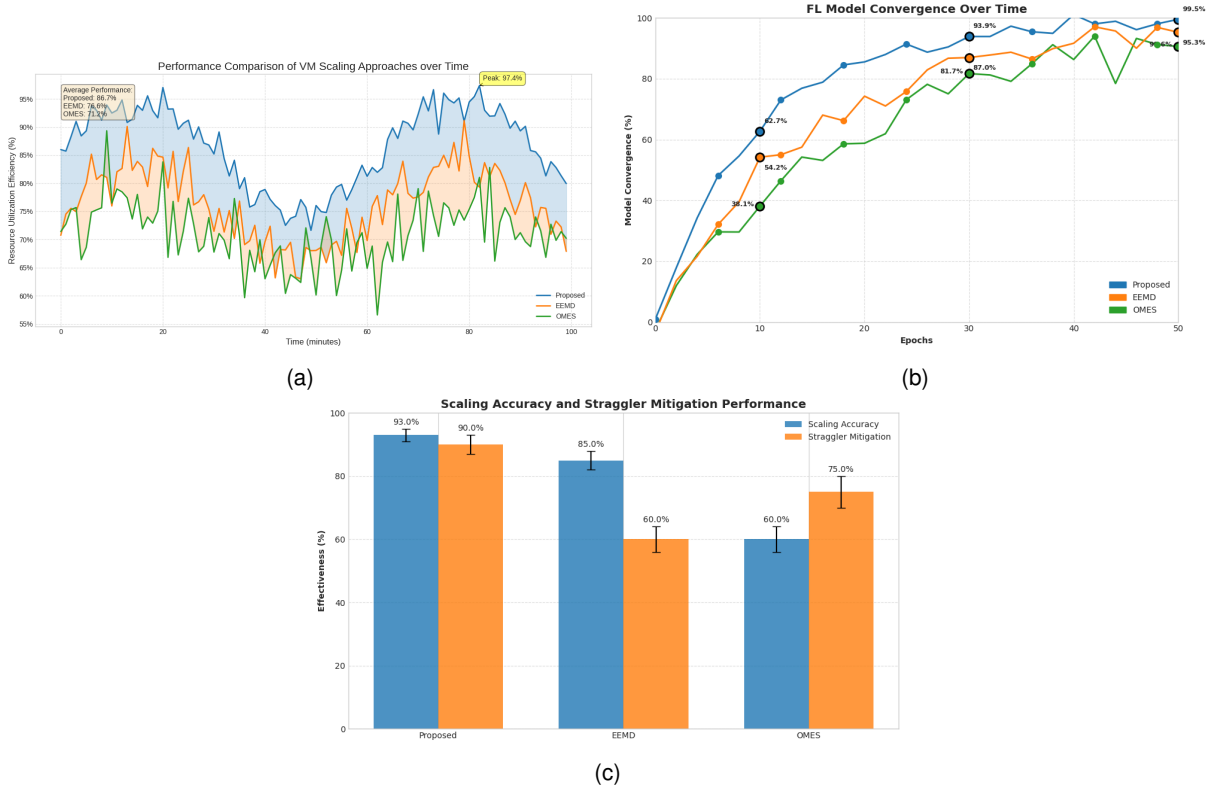


FIGURE 4: Performance comparison of the proposed approach with EEMD and OMES: (a) Resource Utilization Efficiency over time, (b) Model Convergence across training epochs, and (c) Scaling Accuracy and Straggler Mitigation effectiveness. These graphs demonstrate the proposed method's efficiency in resource management, learning speed, and system adaptability in FL environments.

mode decomposition for time series forecasting, and Optimal Heuristic Elastic Scheduling (OMES),¹⁹ which employs heuristic algorithms for optimal VM selection and allocation in cloud clusters. In Figure 4, three important FL performance indicators are compared between our suggested method and EEMD and OMES. The enhanced Resource Utilisation Efficiency in Fig. 4(a), which is attributed to our dynamic VM scaling technique that adjusts in real-time to shifting workloads. This is in contrast to the cluster-based allocation of OMES and the set prediction intervals of EEMD. For our solution, Fig. 4(b) shows quicker Model Convergence and greater final accuracy due to effective resource allocation to essential nodes, hence ensuring equal participation in the FL process and reducing stragglers. The better Scaling Accuracy and effective Straggler Mitigation of our technique, which combines predictive scaling and intelligent task offloading, are shown in Fig. 4(c). Compared to OMES's heuristic scheduling and EEMD's time series forecasting, this technique allows for better load balancing. These findings demonstrate the usefulness of our strategy in

tackling important FL concerns. Our method improves system performance and allocates resources optimally in FL settings by using both vertical and horizontal scaling strategies together with a deep learning prediction model. We used the MNIST and CIFAR-10 datasets to evaluate our approach in order to confirm its robustness. Our technique showed remarkable adaptability, reaching 95.34% and 91.73% prediction accuracy, respectively, with efficient resource utilisation (88% for MNIST, 85% for CIFAR-10), despite the increased complexity of CIFAR-10. Interestingly, as compared to static allocation approaches, our predictive system considerably reduces task delay and resource utilization with very little cost (2–3% CPU, 50–100MB memory). Also, Our CNN technique demonstrated 8.27% enhanced accuracy and 15.85% quicker prediction time in resource demand forecasting for FL workloads as compared to LSTM.

Open Research Challenges

Privacy-Preserving Scaling

FL requires privacy-preserving VM scaling, it is essential for sensitive edge device data. Future work should investigate safe multi-party computing protocols and sophisticated cryptographic approaches to guarantee secure scaling operations that adhere to stringent FL privacy criteria and preserve the privacy of local datasets.

Cost Optimization

To develop sophisticated cost-aware VM scaling strategies to dynamically allocate resources based on the most economical spot instances available, aiming to minimize scaling costs and enhance operational efficiency in FL systems. Consider factors such as spot instance pricing fluctuations, historical cost data, and workload predictions to make informed predictive scaling decisions, thereby achieving better economies of scale and cost-effectiveness.

Fault Tolerance and Resilience

Advanced fault tolerance strategies are needed to advance VM scalability in FL and safeguard the ML model and data integrity from failures and security threats. Future studies should concentrate on creating robust systems that can withstand connection problems and network oscillations, guaranteeing continuous functioning in vital applications such as autonomous cars and healthcare.

Conclusion

This paper presents a novel AI-driven predictive auto-scaling method specifically designed for FL services for virtual machines running on MEC servers. Our detailed research takes into account the synergy between MEC and cloud, addressing issues with MEC server overload and FL service complexity. In FL contexts, the research focuses on effective computational load allocation, straggler reduction, and latency control. Compared to current approaches, our solution shows significant gains, highlighting the promise of adaptive resource allocation in edge-cloud systems. This study establishes the groundwork for further investigations into complicated FL structures, improved privacy-preserving methods, and sophisticated resource optimisation. To fully realize the promise of AI-driven IoT applications and advance distributed machine learning, it will be essential to tackle these difficulties as edge computing and FL developments.

ACKNOWLEDGMENTS

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (2022-0-01032, Development of Collective Collaboration Intelligence Framework for Internet of Autonomous Things, 50%) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00423362, Collaborative Perception and Intelligence Framework for Hyper-connected Interaction among Human and Intelligent Things, 50%).

REFERENCES

1. A. Asheralieva, D. Niyato, and X. Wei, "Ultra-reliable low-latency slicing in space-air-ground multi-access edge computing networks for next-generation internet of things and mobile applications," *IEEE Internet of Things Journal*, 2023.
2. Z. Zhang, B. Guo, W. Sun, Y. Liu, and Z. Yu, "Cross-fcl: Toward a cross-edge federated continual learning framework in mobile edge computing systems," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 313–326, 2024.
3. D. Saxena, J. Kumar, A. K. Singh, and S. Schmid, "Performance analysis of machine learning centered workload prediction models for cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 4, pp. 1313–1330, 2023.
4. S. Trindade, L. F. Bittencourt, and N. L. da Fonseca, "Resource management at the network edge for federated learning," *Digital Communications and Networks*, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352864822002243>
5. D. G. Morín, P. Pérez, and A. G. Armada, "Toward the distributed implementation of immersive augmented reality architectures on 5g networks," *IEEE Communications Magazine*, vol. 60, no. 2, pp. 46–52, 2022.
6. Y. Gao, G. Lu, J. Gao, and J. Li, "A high-performance federated learning aggregation algorithm based on learning rate adjustment and client sampling," *Mathematics*, vol. 11, no. 20, p. 4344, 2023.
7. K. Fan, J. Hong, W. Li, X. Zhao, H. Li, and Y. Yang, "Flsg: A novel defense strategy against inference attacks in vertical federated learning," *IEEE Internet of Things Journal*, vol. 11, no. 2, pp. 1816–1826, Jan 2024.
8. S. Wan, J. Lu, P. Fan, Y. Shao, C. Peng, K. B. Letaief, and J. Chuai, "How global observation works in federated learning: Integrating vertical training into hor-

- horizontal federated learning," *IEEE Internet of Things Journal*, vol. 10, no. 11, pp. 9482–9497, June 2023.
9. L. Toka, G. Dobreff, B. Fodor, and B. Sonkoly, "Machine learning-based scaling management for kubernetes edge clusters," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 958–972, 2021.
10. T. Subramanya and R. Riggio, "Centralized and federated learning for predictive vnf autoscaling in multi-domain 5g networks and beyond," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 63–78, 2021.
11. K. Ray and A. Banerjee, "Horizontal auto-scaling for multi-access edge computing using safe reinforcement learning," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 6, pp. 1–33, 2021.
12. A. Al-Dulaimy, J. Taheri, A. Kassler, M. R. Hoseiny-Farahabady, S. Deng, and A. Zomaya, "Multiscaler: A multi-loop auto-scaling approach for cloud-based applications," *IEEE Transactions on Cloud Computing*, vol. 10, no. 4, pp. 2769–2786, 2020.
13. S. Heidari and R. Buyya, "A cost-efficient auto-scaling algorithm for large-scale graph processing in cloud environments with heterogeneous resources," *IEEE Transactions on Software Engineering*, vol. 47, no. 8, pp. 1729–1741, 2021.
14. M. Diamanti, P. Charatsaris, E. E. Tsiropoulou, and S. Papavassiliou, "Incentive mechanism and resource allocation for edge-fog networks driven by multi-dimensional contract and game theories," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 435–452, 2022.
15. H. P. Phyu, R. Stanica, and D. Naboulsi, "Multi-slice privacy-aware traffic forecasting at ran level: A scalable federated-learning approach," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2023.
16. P. Li, G. Cheng, X. Huang, J. Kang, R. Yu, Y. Wu, M. Pan, and D. Niyato, "Snowball: Energy efficient and accurate federated learning with coarse-to-fine compression over heterogeneous wireless edge devices," *IEEE Transactions on Wireless Communications*, vol. 22, no. 10, pp. 6778–6792, 2023.
17. D. Bega, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "Deepcog: Cognitive network management in sliced 5g networks with deep learning," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 280–288.
18. J. Chen and Y. Wang, "A resource demand prediction method based on eemd in cloud computing," *Procedia Computer Science*, vol. 131, pp. 116–123, 2018.
19. K. Tuli and M. Malhotra, "Optimal meta-heuristic elas-

tic scheduling (omes) for vm selection and migration in cloud computing," *Multimedia Tools and Applications*, vol. 83, no. 12, pp. 34 601–34 627, 2024.

Sakshi Patni is currently working as Research Professor in the Department of Computing, Gachon University, South Korea. Before joining Gachon University, she worked as an Assistant Professor in the Engineering Institutes. Her main research interests are Cloud Computing, Resource management, Federated Learning and Information Security. She has recieved best paper Award in IEEE International Conference (ICECCS) organized in Malaysia. Contact her at sakshichhabra555@gmail.com.

Sungpil Woo is working as researcher in Autonomous IoT Research Section from Electronics and Telecommunications Research Institute (ETRI), South Korea. His research interests include Deep Multimodal Representation Learning and Reinforcement Learning for Healthcare and Internet of Things (IoT). Contact him at woosungpil@etri.re.kr.

Joohyung Lee is currently an Associate Professor with the School of Computing, Gachon University and also a Visiting Fellow with the Department of Electrical and Computer Engineering, Princeton University. Before joining Gachon University, he was a Senior Engineer with Samsung Electronics From 2014 to 2017. His current research interests include federated learning, 6G networks, cloud/edge computing, smart grids, augmented reality, virtual reality, and network economics. He is the corresponding author of this article. Contact him at j17.lee@gachon.ac.kr.