FedPC: Cloud CPU Forecasting using Federated Learning and Periodicity-based Clustering

DOI:https://doi.org/10.54654/isj.v2i25.1115

Nguyen Quoc Khanh, Tran Quang Duc*, Nguyen Van Toan, Tong Van Van

Abstract— **Accurate CPU workload forecasting is vital for efficient resource management and system availability in cloud computing, yet faces challenges in data privacy, security, and the "cold start"problem for new Virtual Machines (VMs). Traditional methods risk privacy and struggle with limited data. We propose FedPC, a novel framework leveraging privacy-preserving Federated Learning (FL) with Periodicity-based Clustering. FedPC enables collaborative training without exposing local data, crucially supporting effective forecasting for new VMs with minimal historical information. It clusters VMs by workload periodicity, training tailored LSTMs within each cluster to handle heterogeneity securely. Evaluated on Azure Public Dataset V1, FedPC surpasses FedAvg in privacy and matches state-of-the-art methods in accuracy. This demonstrates FedPC's efficacy in securely, adaptively managing resources, thereby enhancing system availability, especially in dynamic cloud environments with frequent VM creation and scarce initial data.**

Tóm tắt— **Dự báo mức độ sử dụng CPU đóng vai trò quan trọng trong việc quản lý tài nguyên và đảm bảo tính sẵn sàng của hệ thống dịch vụ điện toán đám mây. Tuy nhiên, bài toán này gặp phải hai thách thức liên quan đến tính riêng tư của dữ liệu và vấn đề "khởi động nguội" (cold-start) ở máy ảo (Virtual Machine - VM) mới, khi dữ liệu giám sát của VM còn hạn chế. Trong nghiên cứu này, nhóm tác giả đề xuất FedPC, một phương pháp kết hợp học Liên hợp (FL) với phân cụm theo chu kỳ. Cụ thể, kiến trúc đề xuất FedPC phân cụm VM dựa trên chu kỳ hoạt động, cho phép huấn luyện mô hình chung mà không lộ dữ liệu cục bộ, từ đó cải thiện độ chính xác trong kết quả dự báo cho những VM mới khởi tạo. Kết quả thử nghiệm trên bộ dữ liệu Azure Public Dataset V1 cho thấy FedPC vượt trội so với các phương pháp hiện tại về độ chính xác, chứng minh khả năng quản lý tài nguyên an toàn, linh hoạt, nâng cao tính sẵn sàng trong môi trường đám mây.**

Keywords— Federated Learning; Privacy-Preserving; Data Security; Workload Forecasting; Cloud Computing; LSTM; Secure Resource Management.

Từ khóa— Học liên hợp; Bảo vệ Quyền riêng tư; An toàn Dữ liệu; Dự báo tải; Điện toán đám mây; LSTM; Quản lý tài nguyên an toàn.

I. INTRODUCTION

In recent years, the rapid development of artificial intelligence (AI) and large language models (LLMs) such as ChatGPT, GPT-4, or BERT has been associated with an explosion in cloud computing demand. An increasing number of organizations and individuals are deploying services and applications on cloud platforms like Microsoft Azure, Amazon AWS, and Google Cloud, with many advantages such as not requiring direct configuration, being able to adjust resources flexibly, and the ability to scale when necessary. Especially for problems requiring significant resources in the short term, like AI/LLMs, businesses can save on deployment, operation, and maintenance budgets. However, with the ability to change computing resources, forecasting server load in cloud computing environments is essential to optimize and save computational resources to avoid redundancy and ensure sufficient capacity in situations of abnormal computing load increases. Load prediction systems help optimize resources, reduce operating costs, and ensure stability in services provided to customers. However, prior methods such as those in [1 - 4] are often limited by their input vector sequences, allowing them to capture only a small subset of data features. Furthermore, they typically fail to

This manuscript was received on May 19, 2025. It was reviewed on June 17, 2025, revised on July 19, 2025 and accepted on August 05, 2025.

* Corresponding author.

exploit the long-term periodicity or recurring patterns within the data.

In cloud virtual machines (VMs), CPU is an important resource parameter for evaluating the computational load of services and applications. The problem of predicting CPU usage has received increasing attention from the research community as well as cloud service providers. Accurately predicting future usage loads can minimize energy consumption and computational resources while providing better service to customers. CPU consumption prediction falls under the category of time series forecasting, with many challenges such as high volatility, being influenced by user demand factors, latency when processing large-scale data, data security, and large computational requirements. Currently, existing solutions are mainly divided into two main groups: statistical methods and machine learning methods [5]. The first method characterizes the sequence of CPU measurements as a white noise source controlling a filter and uses historical data to find filter parameters. With the latter method, a function is estimated by fitting a curve to the set of CPU samples.

In this paper, we aim to optimize the accuracy of load forecasting in cloud computing in the context of limited input (training) data. This is consistent with the characteristics of cloud systems in which VMs are continuously created and deleted. For newly initialized VMs, there will not be enough data to train models in the initial phase. The federated learning problem combined with clustering will solve this problem by sharing a global model, calculated from the weights of models from other VMs, for use by a newly joined virtual machine. Periodicity-based clustering helps group machines with the same characteristics to better share data features. To achieve this, we have made the following contributions:

- We introduce FedPC, a federated learning architecture combined with periodicity-based clustering (Periodicity-Aware Clustering). It has been shown that CPU data exhibits long-term dependencies and periodicity. This periodicity is evident because tasks are typically sent to cloud services periodically. By exploiting and leveraging periodicity in prediction, we demonstrate that the proposed algorithm yields better results than other solutions with limited input data.
- We have conducted a comprehensive evaluation of the solution using CPU data from 160 Microsoft Azure VMs. FedPC has been proven to be superior to other advanced techniques, including ARIMA, Bayesian Neural Networks (HBNN), Probabilistic LSTM (LSTMD), and conventional federated learning models.

The remainder of this paper is organized as follows: Section II reviews related work on CPU usage prediction methods. Section III presents our proposed FedPC framework. Section IV describes the experimental setup, including dataset specifications and performance metrics, and presents comparative results of FedPC and other state-of-the-art workload prediction techniques. Finally, Section V concludes the paper with a summary of our contributions and directions for future work.

II. Related Work

CPU usage prediction methods can be broadly categorized into two main groups: statistical and machine learning. The statistical group includes commonly used methods such as Autoregressive Integrated Moving Average (ARIMA) [6], and Seasonal ARIMA (SARIMA) [7]. Both ARIMA and SARIMA, however, require parameter tuning, which can be done manually or automated using techniques like grid search or AIC-based selection, and fail to capture nonlinearity. Khan et al. [8] grouped VMs with similar load patterns using co-clustering, then applied HMM for prediction, reducing error by 20% compared to ARIMA. Gong et al. [9] combined HMM with signal analysis in the PRESS system, predicting short-term load on the Xen platform. HMM performs well with multi-state data but incurs high computational costs as the number of states increases, though techniques like pruning or approximate inference can mitigate this overhead.

The machine learning group prominently features the LSTM model. LSTM, a recurrent neural network, handles long-term dependencies through its gating mechanism. Patel et al. [10] combined 1D CNN (for local feature extraction) with LSTM to predict CPU load, reducing MSE

by 15% compared to ARIMA. Variants like BiGRU with wavelet integration (Dogani et al. [11]) decompose load signals into frequency components, predicting via an attention mechanism and improving accuracy by 56% on Alibaba data. LSTM excels with noisy data but requires substantial training data, which can be addressed using techniques like transfer learning or data augmentation. Traditional neural networks such as ECNN (Error Correction Neural Network) and BPNN (Backpropagation Neural Network) focus on learning nonlinearity. Islam et al. [12] used ECNN with linear regression to predict multi-resource loads, reducing SLA violations by 25%. Lu et al. [13] proposed RVLBPNN (Random Variable Learning BPNN), achieving higher accuracy than HMM on Google Cloud data. These models adapt quickly to short-term fluctuations but are prone to getting trapped in local minima, though modern optimization techniques like Adam or RMSprop help mitigate this issue. It has been demonstrated that LSTM and other machine learning models generally outperform statistical methods like ARIMA and SARIMA in cloud load prediction due to their ability to learn long-term dependencies and model nonlinear patterns. However, ARIMA and SARIMA may be more suitable for short-term predictions and scenarios where data characteristics shift. The choice of an appropriate method depends on the data's properties and the application's specific requirements.

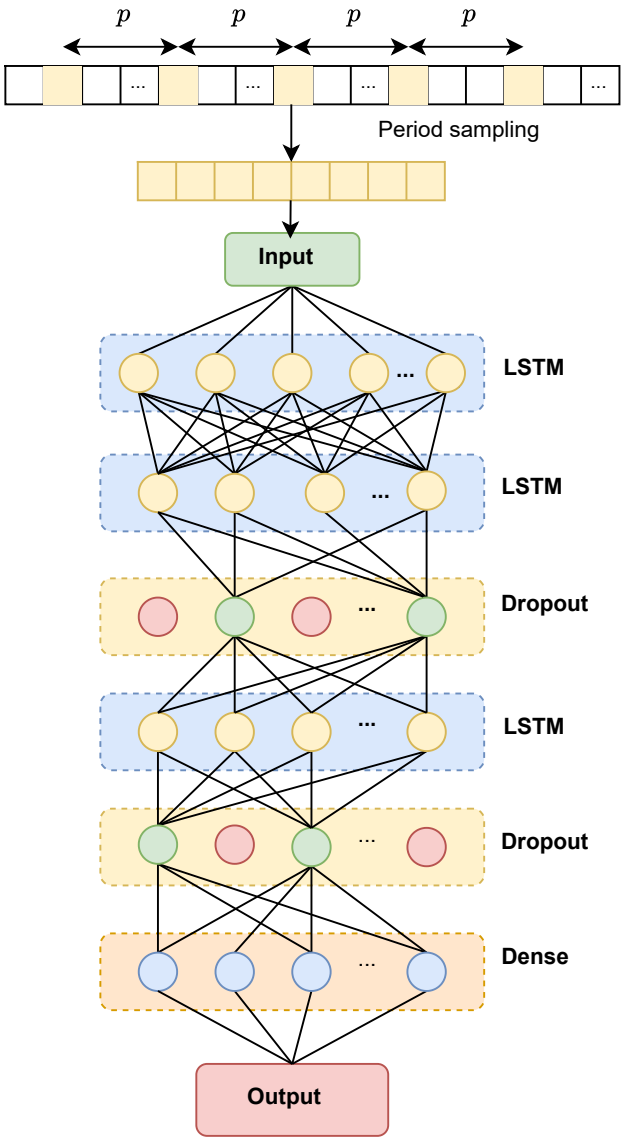

Figure 1. The architecture of PA-LSTM

III. Methodology

This section presents the methodology of our proposed FedPC framework. Our approach integrates a Periodicity-Aware LSTM (PA-LSTM) network as the core predictive engine within a novel federated learning scheme. The discussion covers the entire process, from the initial clustering of clients based on workload periodicity, through the collaborative training protocol within each cluster, to the final strategy for model aggregation.

A. Periodicity-based LSTM

Cloud computing workloads, particularly CPU utilization, frequently exhibit periodic patterns corresponding to daily, weekly, or other cyclical user behaviors [14], [15]. While standard Long Short-Term Memory (LSTM) networks excel at capturing long-term dependencies, their performance can be further enhanced by explicitly incorporating knowledge of these recurring patterns. Recognizing this, and inspired by recent work on Periodicity-Aware LSTM (PA-LSTM) [5], we employ a PA-LSTM architecture specifically adapted for our federated learning (FL) framework. This approach leverages the dominant period identified within a VM's workload history to structure its input features, proving effective within the distributed learning context.

The core distinction of PA-LSTM lies in its input construction strategy. Instead of feeding consecutive past measurements into the network, PA-LSTM employs periodic sampling based on

the determined period p. Specifically, to predict the workload at a future time step implied by horizon h, the input feature vector $\mathbf{z}$ at time t is constructed using historical data points sampled at intervals of p, as defined in:

$$\begin{aligned}\mathbf{z} = [&x(t - \Delta - (\textbf{\textit{len}} - 1) \times p),\\ &x(t - \Delta - (\textbf{\textit{len}} - 2) \times p),\\ &\dots,\\ &x(t - \Delta)]\end{aligned}$$

where ***len*** represents the number of past periodic samples considered, and $\Delta = \lceil h/p \rceil \times p$ is an offset ensuring that the input window aligns appropriately with the prediction horizon h. If $p = 1$, this formulation naturally reduces to the standard LSTM input sequence.

This periodically sampled input vector $\mathbf{z}$ is then processed by the PA-LSTM. In FedPC, the PA-LSTM network architecture, illustrated in Figure 1, comprises six layers: three stacked LSTM layers designed for hierarchical temporal feature extraction (capturing short-, medium-, and long-term patterns inherent in periodic data), followed by two dropout layers included for regularization. These dropout layers are particularly vital within our FedPC framework to mitigate overfitting and enhance model generalization across potentially heterogeneous client data distributions. The architecture concludes with a dense layer employing linear activation, which maps the learned temporal representations to the final continuous CPU utilization prediction. Critical hyperparameters, including dropout rate, batch size, and input sequence length (*len*), were carefully tuned to optimize predictive performance.

B. Federated Periodicity-based Clustering - FedPC

Federated Learning (FL) [16] offers inherent privacy guarantees and naturally leverages distributed data, making it highly suitable for cloud workload forecasting where data originates from numerous distinct VMs and cannot be centrally aggregated. However, the standard Federated Averaging (FedAvg) algorithm [16], while foundational, often struggles with the statistical heterogeneity commonly observed across different VMs' workload patterns. This heterogeneity can significantly impede the convergence and final performance of a single global model trained across all clients. To address this critical challenge of data heterogeneity within the FL context, we propose Periodicity-based Clustering (FedPC) framework employing a clustering strategy. Specifically, we group clients (VMs) based on the similarity of their dominant workload periodicity, a key characteristic identified in cloud data [15, 17]. This allows for the collaborative training of specialized forecasting models within more homogeneous clusters of clients. By tailoring models to specific periodic behaviors, FedPC aims to significantly improve predictive accuracy compared to training a single, monolithic global model attempting to capture diverse patterns simultaneously.

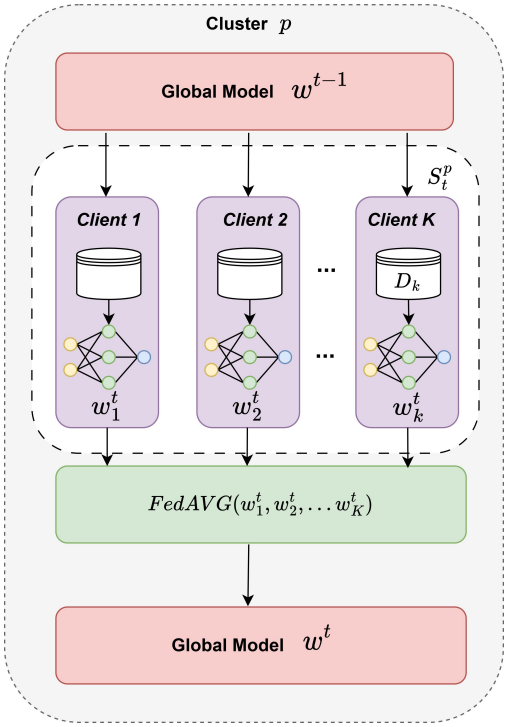

Figure 2. The architecture of FedPC

The proposed FedPC, detailed in Algorithm 1 and Figure 2, operates through distinct initialization and iterative training phases to collaboratively learn cluster-specific workload forecasting models. During initialization, each client k first analyzes its local dataset D_k using the Periodogram method to identify its dominant workload period p_k. The local data is then processed via Period Sampling based on p_k to yield a potentially transformed dataset D_k', optimized for period-aware modeling. Clients are subsequently assigned to initial clusters $S_0^{p_k}$

Algorithm 1: FedPC

Input:

T: number of FL rounds
K: total number of clients, indexed by k
D_k: local dataset of client k
C: total number of clusters, indexed by p
M_p: pre-trained PA-LSTM model with period p
τ: threshold for client filtering
t_{filter}: minimum round number to start client filtering

Initialize:

for *each client* k *in* K **do**
  $p_k \leftarrow$ **Periodogram**(D_k)
  $D'_k \leftarrow$ **Period Sampling**(D_k, p_k)
  $S_0^{p_k} \leftarrow S_0^{p_k} \cup \{k\}$
end
for *each cluster* C_p *in* C **do**
  On the server: setup global model parameters
  $\boldsymbol{w}^0 \leftarrow M_p$
  for *each round* t **do**
    Send $\boldsymbol{w}^{t-1}$ to the clients that are elements of S_t^p
    for *each client* $k \in S_t^p$ ***in parallel*** **do**
      if $t \geq t_{filter}$ **then**
        Split D'_k into training set D_k^{train} and validation set D_k^{val}
        if $\tau <$ ***Loss***$(\boldsymbol{w}^0, D_k^{val})/$***Loss***$(\boldsymbol{w}_k^t, D_k^{val})$ **then**
          $S_t^p \leftarrow S_t^p \cup \{k\}$
        end
      end
      $\boldsymbol{w}_k^t \leftarrow$ **PA-LSTM**$(\boldsymbol{w}_k^{t-1}, D'_k)$
    end
    Determine the global model updates, i.e.,
    $\boldsymbol{w}^t \leftarrow \frac{1}{|S_t^p|} \sum_{k \in S_t^p} \boldsymbol{w}_k^t$
    Update clients in cluster in next round
    $S_{t+1}^p \leftarrow S_t^p$
  end
end

Output: The global model $\boldsymbol{w}^T$

according to their identified periods. Concurrently, the central server initializes a separate global model w^0 for each cluster p, utilizing a pre-trained PA-LSTM model M_p specifically configured for the cluster's characteristic period.

Following initialization, the framework proceeds through T rounds of federated training. In each round t, the server disseminates the current global model weights w^{t-1} associated with cluster p to all clients actively participating in the set S_t^p. Each client k then performs local training, using its local data D'_k and the PA-LSTM architecture to produce a new local model w_k^t. A crucial component of FedPC is a client filtering mechanism, activated for rounds $t \geq t_{filter}$. Within this mechanism, each client assesses the performance improvement achieved by its local training. Specifically, it compares the loss of its newly trained local model w_k^t against the loss of the cluster's initial pre-trained model w^0 on a held-out portion of its local data (D_k^{val}). If the performance of the new local model w_k^t does not meet the required threshold τ when compared to the pre-trained model w^0 (or is worse), the client is marked as unsuitable for contributing in this round, and its weights are excluded from the subsequent aggregation step. The server subsequently computes the updated global model w^t for the cluster by averaging the weights w_k^t received exclusively from the contributing clients (those passing the filter). The set of active clients S_{t+1}^p is prepared for the subsequent round, and this iterative process continues until round T. The final outcome of the FedPC process is a set of optimized, cluster-specific global models w^T.

IV. Experiments

A. Dataset specification

In this paper, we examined our solution and other algorithms using Azure Public Dataset V1 [**?**] . Azure Public Dataset V1 contains CPU traces for 2,013,767 VMs running over a three-month period from November 2016 to February 2017. Each VM has at least 28 consecutive days of CPU load, which is recorded every 5 minutes.

We performed out-of-sample predictions for three different horizons, i.e., 10, 30, and 60 minutes ahead. These horizons correspond to 2, 6, and 12 steps ahead for the Azure Public Dataset V1. A regressor that works well for one horizon may not capture patterns for another. We used various horizons to demonstrate the ability

of the proposed method to cope with CPU loads that fluctuate over time. Experiments were run on a system with an Nvidia RTX 3080 GPU and an Intel Core i7 CPU with 64 GB of RAM.

B. Performance measures

To measure the performance of this solution, we employ both Mean Squared Error (MSE), Mean Absolute Error (MAE) which are defined as follows.

$$\text{MSE} = \frac{\sum_{t=1}^{T}(x(t) - \hat{x}(t))^2}{T} \quad (1)$$

$$\text{MAE} = \frac{\sum_{t=1}^{T}|x(t) - \hat{x}(t))|}{T} \quad (2)$$

C. Comparison with FedAvg and PA-LSTM

In this section, the study compares the proposed FedPC solution with a non-federated PA-LSTM [5] and other federated learning (FL) architectures to evaluate the effectiveness of periodicity-based clustered federated learning. FedAvg serves as the basic FL baseline, simply averaging weights from all clients. *FedAvg with PA-LSTM* represents a naive integration where clients use PA-LSTM locally, but aggregation still occurs globally without clustering. The standalone, non-federated PA-LSTM serves as a benchmark. This baseline uses the same PA-LSTM architecture described in Section III but is trained individually on each client's data (simulating ideal performance without federated learning constraints or communication overhead).

TABLE I. MSE AND MAE OF FEDPC AND OTHER FL METHODS ON AZURE PUBLIC DATASET V1

Horizon	Method	MSE	MAE
10 minutes	FedAvg	23.804	1.8962
	FedAvg with PA-LSTM	17.439	1.6115
	PA-LSTM	6.554	0.8816
	FedPC	**5.811**	**0.8761**
30 minutes	FedAvg	24.985	1.9339
	FedAvg with PA-LSTM	19.147	1.7018
	PA-LSTM	7.688	0.9431
	FedPC	**7.066**	**0.9488**
60 minutes	FedAvg	24.742	2.0274
	FedAvg with PA-LSTM	20.347	1.7797
	PA-LSTM	8.647	1.0229
	FedPC	**7.717**	**1.0126**

Table I presents a comparative evaluation of our proposed FedPC framework against baseline federated learning approaches and the original PA-LSTM model [5] across varying prediction horizons. The results clearly underscore the limitations of standard Federated Averaging (FedAvg), particularly in scenarios involving a large number of clients (160 VMs in this case). FedAvg consistently exhibits the highest Mean Squared Error (MSE) and Mean Absolute Error (MAE), for instance, reaching an MSE of 23.804 and MAE of 1.8962 for the 10-minute horizon. This degraded performance is attributable to the challenges of averaging models trained on potentially highly heterogeneous client data, which hinders convergence towards an effective global model. Integrating the periodicity-aware model directly within the FedAvg framework (FedAvg with PA-LSTM) leads to a noticeable reduction in error (e.g., MSE drops to 17.439 at 10 minutes). This indicates that leveraging periodicity information via PA-LSTM inherently benefits the forecasting task. However, this naive combination still significantly underperforms compared to the original, non-federated PA-LSTM model (MSE of 6.554). This suggests that the simple averaging process in FedAvg dilutes the specialized features learned by PA-LSTM when aggregating updates from clients possibly belonging to different underlying periodic patterns.

The proposed FedPC framework consistently achieves the lowest or near-lowest error rates across all horizons, closely matching or even slightly outperforming the original PA-LSTM. This superior performance highlights the effectiveness of FedPC's core strategy: combining the privacy-preserving benefits of FL with intelligent, periodicity-based clustering. By grouping clients with similar temporal patterns, FedPC allows the specialized PA-LSTM architecture to be trained more effectively within each cluster, mitigating the negative impact of heterogeneity. This clustering approach ensures that model aggregation occurs among more compatible clients, leading to robust and accurate cluster-specific models. Impressively, FedPC achieves these state-of-the-art results within the FL paradigm, demonstrating its ability to effectively leverage distributed data through collaborative learning within well-defined clusters.

D. Comparison with other methods

This section compares the FedPC framework with other recent methodologies, including ARIMA [1], HBNN [2], LSTMD [2], and esDNN [4], each offering unique approaches to time series and workload forecasting. ARIMA is a statistical model combining autoregressive (AR) and moving average (MA) elements. LSTMD leverages a probabilistic Long Short-Term Memory (LSTM) framework incorporating uncertainty. HBNN employs a Bayesian Last Layer network blending deep learning with probabilistic inference. EsDNN is a supervised Deep Neural Network tailored for short-term CPU workload forecasting, capitalizing on deep learning to detect complex patterns.

TABLE II. MSE AND MAE OF FEDPC AND OTHER STATE-OF-THE-ART METHODS ON AZURE PUBLIC DATASET V1

Horizon	Method	MSE	MAE
10 minutes	LSTMD	8.030	1.1588
	ARIMA	13.343	1.4321
	HBNN	40.212	3.1045
	esDNN	6.170	0.9520
	FedPC	**5.811**	**0.8761**
30 minutes	LSTMD	14.759	1.4361
	ARIMA	13.393	1.4349
	HBNN	63.057	3.8413
	esDNN	7.129	0.9927
	FedPC	**7.066**	**0.9488**
60 minutes	LSTMD	10.727	1.3436
	ARIMA	13.447	1.4392
	HBNN	69.715	4.3192
	esDNN	8.085	1.0469
	FedPC	**7.717**	**1.0126**

Table II provides a comparative analysis of Mean Squared Error (MSE) and Mean Absolute Error (MAE) for FedPC against various state-of-the-art forecasting methods on the Azure Public Dataset V1 across three prediction horizons. The evaluated methods include LSTMD, ARIMA, HBNN, and esDNN. FedPC demonstrates superior performance, consistently achieving the lowest MSE and MAE across all prediction horizons compared to these other state-of-the-art techniques. For example, at the 10-minute horizon, FedPC achieves an MSE of 5.811 and an MAE of 0.8761, surpassing the next-best method, esDNN (MSE 6.170, MAE 0.9520). These results strongly underscore its efficiency and potential for deployment even when individual clients might initially have limited historical data, a common scenario in dynamic cloud environments (the "cold start"problem).

In comparison, traditional methods like ARIMA consistently show higher errors, while advanced deep learning models like LSTMD and esDNN perform reasonably well but are outperformed by FedPC. The HBNN model exhibits significantly higher errors across all horizons.

As the prediction horizon increases from 10 to 60 minutes, error metrics generally rise for all methods, reflecting the inherent difficulty of longer-term forecasting. However, FedPC maintains its lead and robust performance with relatively modest error increases. This highlights FedPC's effective design in leveraging periodic information within a clustered federated setting, maximizing predictive power from distributed data sources and positioning it as a highly promising candidate for resource-constrained or privacy-sensitive applications requiring top-tier accuracy.

V. CONCLUSIONS

In this paper, we addressed the challenge of accurate and privacy-preserving CPU workload forecasting in heterogeneous cloud environments by proposing FedPC, a framework integrating Federated Learning with Periodicity-based Clustering. By identifying workload periodicities and grouping similar VMs, FedPC effectively mitigates the negative impacts of data heterogeneity often encountered in standard FL approaches like FedAvg. Clustering clients based on their periodicity allows the model to leverage temporal patterns efficiently. Our experimental results on the Azure Public Dataset demonstrate that FedPC significantly outperforms FedAvg and achieves accuracy comparable to, or exceeding, the non-federated PA-LSTM baseline, validating the effectiveness of the clustering strategy. Furthermore, FedPC shows competitive performance against various state-of-the-art forecasting methods, highlighting its potential as a practical solution, especially for dynamic environments with newly added VMs possessing limited historical data. Future work could explore dynamic clustering mechanisms, investigate

alternative period detection techniques, and extend the framework to predict other cloud resources.

REFERENCES

[1] X. Fu and C. Zhou, "Predicted affinity based virtual machine placement in cloud computing environments," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 246–255, 2017.

[2] A. Rossi, A. Visentin, S. Prestwich, and K. N. Brown, "Bayesian uncertainty modelling for cloud workload prediction," in *2022 IEEE 15th International Conference on Cloud Computing (CLOUD)*. IEEE, 2022, pp. 19–29.

[3] H. Tung, T. Trang, and V. Linh, "Intrusion detection using federated learning in non-iid data environments," *Journal of Science and Technology on Information Security*, vol. 3, no. 23, pp. 53–61, 2024.

[4] L. Wen, M. Xu, A. N. Toosi, and K. Ye, "Temposcale: A cloud workloads prediction approach integrating short-term and long-term information," in *2024 IEEE 17th International Conference on Cloud Computing (CLOUD)*, 2024.

[5] N. Q. Khanh, V. Tong, C. Dao, T. N. Le, and D. Tran, "Boosted regression for predicting cpu utilization in the cloud with periodicity," *The Journal of Supercomputing*, vol. 80, no. 18, pp. 26 036–26 060, 2024.

[6] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, "Workload prediction using arima model and its impact on cloud applications' qos," *IEEE transactions on cloud computing*, vol. 3, no. 4, pp. 449–458, 2014.

[7] V.-M. Tran, V. Debusschere, and S. Bacha, "Hourly server workload forecasting up to 168 hours ahead using seasonal arima model," in *2012 IEEE International Conference on Industrial Technology*. IEEE, 2012, pp. 1127–1131.

[8] A. Khan, X. Yan, S. Tao, and N. Anerousis, "Workload characterization and prediction in the cloud: A multiple time series approach," in *2012 IEEE Network Operations and Management Symposium*. IEEE, 2012, pp. 1287–1294.

[9] Z. Gong, X. Gu, and J. Wilkes, "Press: Predictive elastic resource scaling for cloud systems," in *2010 International Conference on Network and Service Management*. IEEE, 2010, pp. 9–16.

[10] E. Patel and D. S. Kushwaha, "A hybrid cnn-lstm model for predicting server load in cloud computing," *The Journal of Supercomputing*, vol. 78, no. 8, pp. 1–30, 2022.

[11] J. Dogani, F. Khunjush, and M. Seydali, "Host load prediction in cloud computing with discrete wavelet transformation (dwt) and bidirectional gated recurrent unit (bigru) network," *Computer Communications*, vol. 198, pp. 157–174, 2023.

[12] S. Islam, J. Keung, K. Lee, and A. Liu, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Generation Computer Systems*, vol. 28, no. 1, pp. 155–162, 2012.

[13] Y. Lu, J. Panneerselvam, L. Liu, and Y. Wu, "Rvlbpnn: A workload forecasting model for smart cloud computing," *Scientific Programming*, vol. 2016, 2016.

[14] F. Chen, Z. Qin, H. Zhao, and S. Deng, "Pepnet: A periodicity-perceived workload prediction network supporting rare occurrence of heavy workload," *IEEE Transactions on Cloud Computing*, 2023.

[15] N. M. Tran, T. Nam, and D. H. Epema, "Parallel workload modeling with realistic characteristics," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2138–2148, 2013.

[16] H. B. McMahan, E. Moore, D. Ramage, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

[17] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini, "Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms," in *Proceedings of the 26th Symposium on Operating Systems Principles*, pp. 153–167, 2017.

About the authors

Nguyen Quoc Khanh
Workplace: SoICT, HUST, Vietnam
Email: khanh.nq@soict.hust.edu.vn
Education: Nguyen Quoc Khanh, PhD student at the School of Information and Communication Technology, Hanoi University of Science and Technology.
Recent research interests: machine learning, cloud computing, information security.

Tên tác giả: **Nguyễn Quốc Khánh**
Cơ quan công tác: Trường Công nghệ Thông tin và Truyền thông, Đại học Bách khoa Hà Nội
Email: khanh.nq@soict.hust.edu.vn
Quá trình đào tạo: Hiện là nghiên cứu sinh Tiến sĩ tại Trường Công nghệ Thông tin và Truyền thông, Đại học Bách khoa Hà Nội.
Hướng nghiên cứu hiện nay: Học máy, điện toán đám mây, an toàn thông tin.

Tran Quang Duc
Workplace: SoICT, HUST. Vietnam
Email: ductq@soict.hust.edu.vn
Education: Tran Quang Duc received his PhD in 2015 and became an Associate Professor in 2020. Currently working at the School of Information and Communication Technology, Hanoi University of Science and Technology.
Recent research interests: machine learning, biometrics, information security.

Tên tác giả: **Trần Quang Đức**
Cơ quan công tác: Trường Công nghệ Thông tin và Truyền thông, Đại học Bách khoa Hà Nội
Email: ductq@soict.hust.edu.vn
Quá trình đào tạo: Nhận bằng Tiến sĩ năm 2015, được phong Phó Giáo sư năm 2020, hiện công tác tại Trường Công nghệ Thông tin và Truyền thông, Đại học Bách khoa Hà Nội.
Hướng nghiên cứu hiện nay: Học máy, sinh trắc học, an toàn thông tin.

Nguyen Van Toan
Workplace:
Salesforce, Inc., United States, USA
Email: nguyentoan@salesforce.com
Education: Nguyen Van Toan, Ph.D. (2018), is currently working as a researcher at Salesforce, Inc., United States. His research interests include security, infrastructure security, cloud computing, and human-computer interaction.
Recent research interests: machine learning, cloud computing, information security.

Tên tác giả: **Nguyễn Văn Toàn**
Cơ quan công tác: Salesforce, Inc., Hoa Kỳ
Email: nguyentoan@salesforce.com
Quá trình đào tạo: Nhận bằng Tiến sĩ năm 2018, hiện là nhà nghiên cứu tại Salesforce, Inc., Hoa Kỳ.
Hướng nghiên cứu hiện nay: An ninh mạng, an ninh cơ sở hạ tầng, điện toán đám mây, tương tác người máy.

Tong Van Van
Workplace: SoICT, HUST, Vietnam
Email: vantv@soict.hust.edu.vn
Education: Tong Van Van, Ph.D. (2021), is currently a faculty member at the School of Information and Communication Technology, Hanoi University of Science and Technology. His research interests include machine learning, information security, and blockchain technology.

Tên tác giả: **Tống Văn Vạn**
Cơ quan công tác: Trường Công nghệ Thông tin và Truyền thông, Đại học Bách khoa Hà Nội
Email: vantv@soict.hust.edu.vn
Quá trình đào tạo: Nhận bằng Tiến sĩ năm 2021, hiện là giảng viên tại Trường Công nghệ Thông tin và Truyền thông, Đại học Bách khoa Hà Nội.
Hướng nghiên cứu hiện nay: Học máy, an toàn thông tin, công nghệ Blockchain.