Bridging Edge and Cloud: A Knowledge-Enhanced Framework for Efficient Time Series Anomaly Detection

Shenglin Zhang, *Member, IEEE*, Jiacheng Zhang, Guohua Liu, Shiqi Chen, Chenyu Zhao, Minghua Ma, *Member, IEEE*, Yutong Chen, Yongqian Sun, *Member, IEEE*, and Dan Pei, *Senior Member, IEEE*

***Abstract*—Multivariate time series anomaly detection (MTSAD) plays a crucial role in monitoring system health and ensuring operational reliability across various domains. While deep learning models have shown promising results in this field, deploying these computationally intensive models on resource-constrained edge devices remains challenging, particularly when considering the need for continuous model adaptation to evolving data patterns. We introduce the *RefinedEdge* framework, explicitly developed to enable effective deployment of multivariate time series anomaly detection within resource-constrained edge computing environments. The *RefinedEdge* framework utilizes three key strategies: Aggregated Compression, Knowledge Refinement, and Reciprocal Edge-Cloud Learning. These strategies collectively tackle prominent challenges such as maintaining high accuracy despite significant model compression and ensuring seamless synchronization and continuous adaptation across cloud and edge systems. By reducing the complexity of models without sacrificing performance, and by adapting anomaly detection to the dynamic conditions typical of edge computing, the *RefinedEdge* framework validates its significant enhancements in detection performance through empirical testing on real-world datasets. This makes it a practical solution for real-time applications in sectors such as smart manufacturing.**

***Index Terms*—Anomaly detection, Edge computing, Knowledge distillation, Time-series analysis, Edge-cloud collaboration, Model compression**

I. Introduction

WITH the explosive growth of the Internet of Things (IoT), a multitude of sensors or devices have emerged, churning out copious amounts of time series data during operation [1]. To ensure the reliability of devices, reduce downtime, and prevent malicious attacks, efficient anomaly detection for multivariate time series (MTSAD) has become increasingly critical [2]. For instance, in automotive production, undetected anomalies in operational data can lead to severe disruptions, such as complete shutdowns of production lines. This can result in costly repairs and significant production downtime [3]. The importance of robust anomaly detection systems in manufacturing is crucial to predict and mitigate potential machinery failures before they escalate.

Previously, anomaly detection in cloud computing had evolved into the predominant paradigm, leveraging abundant computing resources to achieve robust detection performance across diverse applications [4]. In the face of the massive growth in data volume, combined with the demands for real-time processing and data privacy, the trend has been gradually migrating from cloud computing to edge computing (EC) [5]. However, the computing power and resource limitations of edge devices pose considerable obstacles to the deployment of complex anomaly detection models. For instance, state-of-the-art time series anomaly detection models typically contain millions or even tens of millions of parameters and require GPU resources for efficient operation [6], while edge devices are usually equipped with only 2.6 GHz six-core CPUs or lower computational capabilities. Through experiments, we found that such CPU-based edge devices can only effectively handle models with fewer than 0.15 M parameters.

Computational disparity creates significant challenges for deploying effective multivariate time series anomaly detection in edge environments. Capturing complex temporal dependencies and variable correlations under strict resource constraints is critical for achieving accurate anomaly detection. Current approaches face a challenging trade-off. Although statistical methods are computationally efficient, they often fail to capture the complex temporal dependencies in multivariate data [7]. Deep learning models can achieve excellent detection performance, but they face challenges in deploying on resource-constrained devices [8]. Cloud-based solutions have computational capabilities, but they also have limitations in practical applications due to latency and bandwidth constraints [9].

Recent advances in edge-cloud collaborative frameworks [10]–[12] have shown promise in balancing computational efficiency with model performance. However, these frameworks primarily focus on computer vision or general machine learning tasks. While these advances provide valuable insights, directly applying them to time series anomaly detection presents significant challenges due to the unique characteristics of temporal data and edge deployment requirements.

Based on our analysis of existing approaches and the specific demands of edge-based time series anomaly detection,

This work was supported by Alibaba Innovative Research Program. *(Corresponding author: Yongqian Sun.)*

Shenglin Zhang, Jiacheng Zhang, Shiqi Chen, Chenyu Zhao, Yutong Chen, and Yongqian Sun are with the College of Software, Nankai University, Tianjin 300071, China (e-mail: zhangsl@nankai.edu.cn; milocheung@mail.nankai.edu.cn; 2120240791@mail.nankai.edu.cn; zhaochenyu@mail.nankai.edu.cn; 2111782@mail.nankai.edu.cn; sunyongqian@nankai.edu.cn).

Guohua Liu is with Alibaba Cloud Computing Ltd, Hangzhou 310030, China (e-mail: suoni@alibaba-inc.com).

Minghua Ma is with Microsoft, Redmond, 98052 USA (e-mail: minghuama@microsoft.com).

Dan Pei is with the Department of Computer Science, Tsinghua University, Beijing 100190, China (e-mail: peidan@tsinghua.edu.cn).

we identify three key challenges that remain inadequately addressed:

Challenge I: Maintaining effective anomaly detection under data quality constraints and limited local datasets in edge environments. Edge devices often encounter issues like sensor failures, environmental interference, and connectivity problems, which lead to the data they receive being noisy, incomplete, and unbalanced. Each edge device operates with significantly smaller local datasets compared to centralized systems, making them more vulnerable to data quality degradation. The limited data volume at individual edge nodes amplifies the impact of noisy or incomplete samples, as there are insufficient clean samples within each local dataset to compensate for the corrupted data, thereby severely compromising the reliability and accuracy of anomaly detection models deployed on individual devices.

Challenge II: Constructing efficient multivariate time series anomaly detection models on resource-limited edge devices. To accommodate scarce resources, multivariate time series anomaly detection models with high complexity must undergo significant pruning or simplification. Nevertheless, a formidable challenge lies in figuring out how to make these compressed models not only proficiently capture the correlations within time series but also guarantee the reliability of detection.

Challenge III: Continuous synchronization and adaptation of models across cloud-edge platforms. Edge computing environments present unique synchronization challenges distinct from cloud-based systems. Heterogeneous edge devices with varying data characteristics necessitate personalized model adaptations rather than the uniform updates typical in cloud environments. Additionally, bandwidth constraints significantly limit the frequency and size of model synchronization between edge and cloud, contrasting sharply with cloud environments that benefit from abundant network resources. These limitations require careful orchestration of model updates to ensure continuous adaptation to evolving data patterns while maintaining real-time detection capabilities without service interruption.

To address these challenges, we adopt a knowledge distillation-based approach that leverages both cloud and edge computing capabilities through a two-model paradigm, where the teacher model is a large, high-capacity model trained on cloud servers with abundant computing resources, and the student model is a lightweight, compressed version designed for resource-limited edge devices. Building on this paradigm, we propose *RefinedEdge*, a reciprocal edge-cloud learning framework that integrates multi-strategy model compression, knowledge refinement, and continuous model adaptation to enable efficient and accurate multivariate time series anomaly detection on edge devices.

Specifically, to tackle the first challenge, we propose an **Aggregated Compression** module that consolidates data from multiple edge devices into a centralized cloud for developing a robust teacher model. By pruning this teacher model to suit resource constraints, we ensure the deployment of lightweight yet reliable anomaly detection models on edge devices. For the second challenge, we adopt a **Knowledge Refinement** process that distills a unified student model from the teacher model using aggregated data. Governed by a balanced distillation coefficient (λ_{kd}), this process enables the student model to effectively capture temporal correlations despite aggressive compression. To address the final challenge, we introduce a **Reciprocal Edge-Cloud Updating** module that establishes a continuous learning cycle between cloud and edge devices. This reciprocal update mechanism guarantees that both teacher and student models remain synchronized and up-to-date in real-time scenarios.

The major contributions are summarized as follows.

- Aggregated Compression: We develop an integrated approach that combines cloud-based data aggregation with ensemble pruning strategies to address both data quality constraints and computational limitations in edge environments, enabling effective model compression while preserving essential temporal patterns.
- Knowledge Refinement: We propose a two-phase knowledge transfer process that combines unified model distillation with personalized adaptation. By balancing reconstruction learning with teacher guidance through an optimized distillation loss function, our approach enables edge models to maintain reliable detection capabilities despite high compression ratios.
- Reciprocal Edge-Cloud Updating: We propose a dynamic learning mechanism that maintains continuous synchronization between cloud and edge models through reciprocal updates, enabling real-time adaptation to evolving data patterns while preserving model performance across distributed environments.
- Empirical Validation: Our framework demonstrates superior performance and strong generalization capabilities across multiple real-world datasets and diverse model architectures. The experiments validate that our approach works effectively with both reconstruction-based and forecasting-based models. The compressed edge models consistently outperform baseline methods while achieving significant parameter reduction compared to teacher models.

In summary, these contributions collectively address the fundamental challenges of deploying effective anomaly detection in resource-constrained edge environments while maintaining the computational advantages of cloud infrastructure. Our framework provides a practical solution that balances detection accuracy, computational efficiency, and real-time responsiveness for edge-cloud collaborative systems.

The remainder of this paper is organized as follows: Section II reviews related work in the field. Section III introduces preliminary concepts and defines the problem space. Section IV details the proposed methodologies, followed by the experimental results in Section V. Section VI discusses the limitations of the current framework and outlines promising directions for future research. Finally, Section VII concludes the paper.

II. Related Work

Time Series Anomaly Detection (TSAD) has emerged as a critical research area with widespread applications in industrial monitoring, healthcare, and finance. This section reviews key developments in TSAD methods, model compression techniques, and edge-cloud collaborative approaches.

A. TSAD Methods

The evolution of TSAD methods spans from traditional statistical approaches to advanced deep learning models. Classical methods include ARIMA [13], EWMA [14], and clustering-based approaches [15]. While these methods offer computational efficiency and interpretability, they often struggle with complex temporal patterns and high-dimensional data.

Deep learning approaches have significantly advanced TSAD through both forecasting-based and reconstruction-based paradigms. Forecasting models predict future values to detect anomalies by measuring deviations between predicted and observed values. Recent Transformer-based architectures have improved the modeling of complex temporal dependencies, with models like CrossFormer [16] and iTransformer [17] demonstrating superior performance in multivariate time series analysis.

Reconstruction methods such as autoencoders [18], VAEs [19], and GANs [20] learn normal data representations to identify anomalies through reconstruction errors. While both paradigms achieve superior detection performance, they often require substantial computational resources, limiting their deployment on edge devices.

Recent time series foundation models like Timer [6] and Moirai [21] have shown promising capabilities in anomaly detection through large-scale pre-training. However, deploying these large models on edge devices would require extreme compression ratios, likely leading to significant performance degradation.

B. Model Compression Techniques

Various compression techniques have been developed to address the computational demands of deep neural networks in resource-constrained environments. Parameter pruning reduces model size by removing unimportant weights [22], while quantization techniques reduce precision requirements [23]. Knowledge distillation transfers expertise from larger teacher models to compact student models, and low-rank factorization [24] approximates weight matrices through techniques like SVD. However, achieving high compression ratios often results in significant performance degradation, particularly for complex tasks like anomaly detection.

C. Edge-Cloud Collaborative Methods

Edge-cloud collaboration frameworks leverage both cloud computing's computational power and edge computing's low-latency capabilities. Task-based frameworks [25] distribute computational loads based on resource requirements, while data-centric approaches [26] optimize data transfer through techniques like federated learning. Dynamic frameworks [10] adapt collaboration strategies to changing operational conditions through end-edge-cloud learning systems. These approaches demonstrate promising results in balancing computational efficiency with model performance, though challenges remain in optimizing resource utilization and maintaining detection accuracy across diverse operational conditions.

III. Preliminaries

TSAD aims to identify patterns that significantly deviate from expected behavior in temporal data. According to the comprehensive survey [27], anomalies are defined as observations that do not follow the expected behavior. For multivariate time series, this involves detecting unusual patterns across multiple interdependent variables, where anomalies may manifest as deviations in individual metrics or as abnormal relationships between different metrics, distinct from routine variations. These anomalous patterns often indicate important events or system failures that require immediate attention. The primary objective is to develop a computationally efficient model that identifies these anomalies in real time while operating within the resource constraints of edge computing environments.

Our framework employs a reconstruction-based teacher model as the primary implementation that learns to reconstruct input data, where the reconstruction error serves as the anomaly score. Compared to forecasting-based models, reconstruction-based models can leverage both past and present information, which has been shown to lead to more accurate anomaly detection performance in time series tasks [28]. Therefore, we adopt this type of model as the base teacher in our primary implementation.

However, we emphasize that our framework is model-agnostic and can effectively be applied to both reconstruction-based and forecasting-based models. Our Framework Generalization Results (Section V-F) provide concrete empirical evidence demonstrating the successful application of our edge-cloud collaborative framework to forecasting-based models.

In the context of reconstruction-based time series anomaly detection, “knowledge” refers to what the model has learned about the predominant behavior present in the training data—temporal dependencies, seasonal cycles, and typical value ranges—that enables it to reconstruct such behavior well [29]. Inputs that are consistent with this predominant behavior are reconstructed with small error; inputs that deviate from it exhibit larger reconstruction gaps (i.e., the difference between the input and its reconstruction).

Separately, it is widely observed in practice that datasets collected from routine operations are dominated by normal samples. [28], [29]. Training on such data consequently equips the model with a solid understanding of normal behavior; deviations from that behavior naturally appear as larger reconstruction gaps and are flagged as anomalies.

Consider web-service telemetry (CPU, memory, disk I/O, request rate). Under routine load, CPU and request rate rise and fall together, with predictable overnight dips and brief backup-related I/O spikes. After training, the model reconstructs such sequences closely; a CPU spike without a matching request-rate increase, or sustained I/O outside backup windows, produces large reconstruction gaps and is flagged as anomalous.

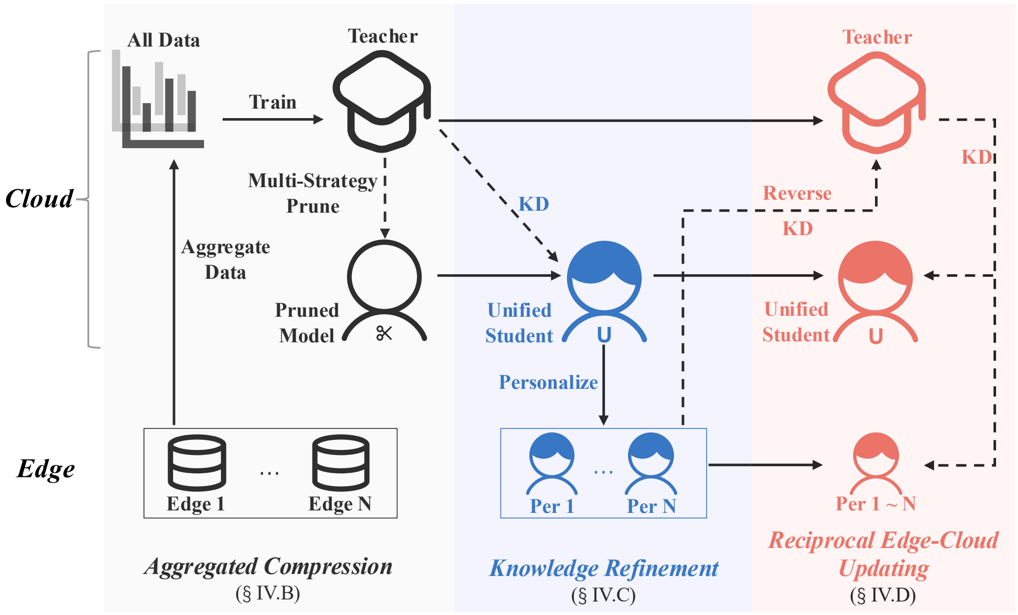


Fig. 1. Overview of *RefinedEdge* with N edge devices. Solid arrows represent model transformations, while dashed arrows (KD) indicate knowledge distillation guidance.

IV. Methodology

A. Framework Overview

Our proposed framework operates through a systematic cloud-edge collaboration mechanism for time series anomaly detection.

Fig. 1 illustrates the comprehensive framework of *RefinedEdge*, which uses an anomaly detection model as the foundation. The framework is composed of three modules, namely Aggregated Compression, Knowledge Refinement, and Reciprocal Edge-Cloud Updating, represented by black, blue, and red colors respectively.

Aggregated Compression. To resolve the data sparsity within a single edge device and enhance the training dataset, *RefinedEdge* first aggregates the data from multiple edge devices in the cloud. Subsequently, these aggregated data are utilized to train a model with strong generalization ability, which is called the “teacher model”. To reduce the complexity of the teacher model, enhance computational efficiency, and make it suitable for edge devices with limited resources, *RefinedEdge* then adopts multiple pruning strategies to obtain the “pruned model”.

Knowledge Refinement. To construct lightweight models that can be utilized on edge devices, *RefinedEdge* distill a unified student model from the teacher model by using the aggregated data in the cloud. Subsequently, this student model is fine-tuned with the local data on each edge device, thereby generating personalized models to meet the needs of different devices. These personalized models will detect anomalies efficiently and accurately.

Reciprocal Edge-Cloud Updating. To ensure the models remain effective and up-to-date, *RefinedEdge* regularly updates the teacher and student models. As the data patterns and operational conditions change at each edge device, the capabilities of personalized models need to be continuously developed. We utilize the updated cloud-based teacher model to guide the retraining of the unified and personalized student models. This cyclical process ensures that each edge device can effectively compute anomaly scores and maintain system reliability.

B. Aggregated Compression

1) Data aggregation: As shown in Fig. 1, Edge 1 to Edge N constitute $\mathbf{n}$ edge devices. To ensure the stability of each edge device, a series of monitoring metrics are continuously collected. Each metric can be represented as:

$$\xi = \{\xi_1, \xi_2, \ldots, \xi_T\} \tag{1}$$

where T is the length of the metric and $\xi_t \in \mathbb{R}$ denotes the observation at time t. These metrics then form a multivariate time series dataset $\mathbf{X}_i \in \mathbb{R}^{T\times m}$, where T is the length of each time series and m is the number of metrics. *RefinedEdge* aggregate these datasets into a comprehensive training set:

$$\mathbf{X}_{agg} = \{\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_n\} \tag{2}$$

After that, a sliding window is employed on $\mathbf{X}_{agg}$:

$$\mathbf{X}^t_{window} = \{\mathbf{X}^{t-w+1}_{agg}, ..., \mathbf{X}^t_{agg}\} \tag{3}$$

where w is the window size determining the temporal scope of historical data used for model training and subsequent updates.

2) Teacher Model Training: *RefinedEdge* employs a reconstruction-based anomaly detection model (TimesNet [30]) as the teacher model $\mathcal{M}_\mathcal{T}$. The model is trained to reconstruct the input data, where the reconstruction error serves as an indicator of anomalies. Using the aggregated data, the teacher model is trained to minimize the reconstruction loss:

$$\mathcal{L}_{recon} = \frac{1}{|\mathbf{X}_{window}|} \sum_{\mathbf{x} \in \mathbf{X}_{window}} \|\mathcal{M}_{\mathcal{T}}(\mathbf{x}) - \mathbf{x}\|_2^2 \quad (4)$$

The choice of mean squared error (MSE) as the reconstruction loss is well-established in time series anomaly detection literature [31]. We use it in this paper as it effectively measures the reconstruction quality of normal patterns while being sensitive to anomalous deviations. When the model is trained on predominantly normal data, it learns to accurately reconstruct normal patterns while producing higher reconstruction errors for anomalous samples. This property makes reconstruction-based loss functions particularly suitable for unsupervised anomaly detection, where labeled anomaly samples are scarce or unavailable during training.

3) Multi-Strategy Model Compression: Pruning method integration can be approached through various strategies: applying different methods to specific model components, optimizing strategy selection through search algorithms, or combining multiple strategies through unified scoring mechanisms [32]. Among these approaches, unified evaluation and scoring offers superior generalizability across model architectures, computational efficiency without iterative optimization overhead, and comprehensive assessment by leveraging diverse pruning perspectives.

Building on this rationale, *RefinedEdge* employs a systematic compression approach utilizing multiple pruning strategies. Different strategies focus on different aspects of model compression, reducing the risk of sub-optimal pruning decisions that may arise when relying on a single strategy.

The pruning process can be formalized as:

$$\mathcal{M}_{pruned} = \mathcal{P}(\mathcal{M}_{\mathcal{T}}, \rho, \mathcal{S}) \quad (5)$$

where $\mathcal{P}$ represents the pruning operation, ρ is the target pruning ratio, and $\mathcal{S}$ denotes the pruning strategy. We integrate four complementary pruning strategies from the TorchPruning library [33]. These strategies provide different perspectives on parameter importance:

Random-Magnitude Strategy. This strategy provides exploration. It combines random importance scoring with magnitude-based channel pruning:

$$\omega_{rand}(\theta) = \mathcal{U}(0, 1) \quad (6)$$

The random scoring helps explore diverse pruning patterns while magnitude-based channel pruning ensures structural efficiency.

Magnitude-Magnitude Strategy. This strategy ensures stability [34]. It employs a dual magnitude-based approach where both importance scoring and channel pruning are based on parameter magnitudes:

$$\omega_{mag}(\theta) = |\theta| \quad (7)$$

This simple yet effective approach identifies and removes parameters with small absolute values while maintaining network structure.

Taylor-Magnitude Strategy. This hybrid approach considers gradient information [35]. It combines Taylor expansion-based importance scoring with magnitude-based channel pruning. The Taylor criterion evaluates parameter importance through first-order gradient information:

$$\omega_{taylor}(\theta) = |\theta \cdot \frac{\partial \mathcal{L}}{\partial \theta}| \quad (8)$$

This is complemented by magnitude-based channel selection to ensure structured sparsity.

BN-Scale Group Strategy. This strategy maintains structural efficiency [36]. It leverages batch normalization scale factors for group-wise structured pruning:

$$\omega_{bn}(\theta) = |\gamma| \quad (9)$$

where γ represents the batch normalization scale parameter. This approach maintains the computational efficiency of grouped convolutions.

RefinedEdge implements an automated multi-strategy pruning process that integrates these strategies through a unified interface. The pruning ratio is gradually increased according to:

$$\rho_i = \rho_0 + (1 - \rho_0) \cdot \frac{i}{\nu}, \quad i \in [1, \nu] \quad (10)$$

where ρ_i is the pruning ratio at iteration i, ρ_0 is the initial ratio, and ν is the total number of iterations.

This gradual pruning strategy is designed to avoid the abrupt removal of a large number of parameters, which could lead to significant performance degradation. Instead, by incrementally increasing the pruning ratio, the model is allowed to adapt its internal representations and fine-tune the remaining parameters at each step. This controlled reduction helps to maintain model stability and accuracy while progressively eliminating redundancy, resulting in a more efficient and robust model [33].

RefinedEdge leverages these strategies through a weighted ensemble approach. The importance scores from each strategy are aggregated using a simple yet effective averaging method:

$$\omega_{ensemble}(\theta) = \frac{1}{|S|} \sum_{s \in S} \omega_s(\theta) \quad (11)$$

where S represents the set of pruning strategies and $\omega_s(\theta)$ is the normalized importance score from strategy s. This ensemble approach helps mitigate potential biases from individual strategies while maintaining computational efficiency.

The multi-strategy ensemble pruning process is detailed in Algorithm 1, which integrates multiple pruning strategies to achieve efficient model compression while maintaining performance. The algorithm takes a teacher model $\mathcal{M}_{\mathcal{T}}$, an initial pruning ratio ρ_0, and the total number of iterations ν as inputs. In each iteration, the pruning ratio ρ_i is gradually increased (Line 5), allowing for progressive model compression. For each layer, importance scores are computed using different strategies (Lines 7-10) and then aggregated (Line 11) to determine which channels to prune (Line 12). After pruning all layers, the model architecture is updated to reflect the removed channels (Line 14). This iterative process continues

Algorithm 1 Multi-Strategy Ensemble Pruning.

```
Input: Teacher model $\mathcal{M}_{\mathcal{T}}$,
        initial pruning ratio $\rho_0$,
        total iterations $\nu$
Output: Pruned model $\mathcal{M}_{pruned}$
    Initialize strategies $S =$
    $\{$Random-Magnitude, Magnitude-Magnitude,
    Taylor-Magnitude, BN-Scale Group$\}$
    for $i = 1$ to $\nu$ do
        $\rho_i \leftarrow \rho_0 + (1 - \rho_0) \cdot i/\nu$
        for each layer $l$ in $\mathcal{M}_{\mathcal{T}}$ do
            scores $\leftarrow \{\}$
            for strategy $s$ in $S$ do
                scores[$s$] $\leftarrow$ compute_importance_scores($l$, $s$)
            end for
            ensemble_scores $\leftarrow$ aggregate_scores(scores)
            prune_channels($l$, ensemble_scores, $\rho_i$)
        end for
        update_model_architecture($\mathcal{M}_{\mathcal{T}}$)
    end for
    return $\mathcal{M}_{pruned}$
```

*The detailed implementations of `compute_importance_scores` and `aggregate_scores` functions are provided in (6)-(9) and (11) respectively.

until the desired compression is achieved, ultimately producing a pruned model $\mathcal{M}_{pruned}$ that maintains the essential features for anomaly detection while significantly reducing computational complexity.

This ensemble approach automatically adapts the network architecture while maintaining the model's anomaly detection capabilities. The pruning process is fully automated, requiring only the specification of the target compression ratio and the model architecture, making it easily applicable to different anomaly detection models.

C. Knowledge Refinement Process

The Knowledge Refinement Process design addresses a fundamental challenge in edge environments. Aggressive model compression often fails to preserve critical temporal patterns. Early experiments revealed that traditional knowledge distillation approaches underperformed when applied to time series anomaly detection models. This limitation stems from inadequate transfer of complex temporal dependencies inherent in time series data. Simply mimicking teacher outputs proves insufficient for time series anomaly detection. The student model must internalize underlying temporal modeling capabilities to maintain detection accuracy under resource constraints. The key insight drives the design of a weighted loss function. This function explicitly encourages the student model to develop both knowledge transfer and independent reconstruction abilities, ensuring robust performance even when deployed independently on edge devices.

Unified Student Model Distillation. The unified student model distillation process employs a knowledge distillation (KD) framework that balances reconstruction learning and teacher guidance through a hyperparameter λ_{kd}. The distillation coefficient λ_{kd} plays a crucial role in balancing self-learning and teacher guidance. Empirical studies show that setting λ_{kd} within 0.4 to 0.7 achieves an effective equilibrium between self-learning and teacher guidance, with detailed analysis provided in Section V-C.

Given the pruned model as the unified student model $\mathcal{M}_{\mathcal{S}}$ and the original teacher model $\mathcal{M}_{\mathcal{T}}$, the distillation objective is formulated as:

$$\mathcal{L}_{total} = \lambda_{kd}\mathcal{L}_{recon} + (1 - \lambda_{kd})\mathcal{L}_{kd} \tag{12}$$

where λ_{kd} is the distillation coefficient that balances the two learning objectives:

$$\mathcal{L}_{recon} = \|\mathcal{M}_{\mathcal{S}}(\mathbf{x}) - \mathbf{x}\|_2^2 \tag{13}$$

$$\mathcal{L}_{kd} = \|\mathcal{M}_{\mathcal{S}}(\mathbf{x}) - \mathcal{M}_{\mathcal{T}}(\mathbf{x})\|_2^2 \tag{14}$$

To ensure the unified student model captures general patterns across all edge devices, the knowledge distillation process is conducted on the aggregated dataset $\mathbf{X}_{agg}$. This unified model serves as the foundation for subsequent personalization and provides a quick initialization for new edge devices joining the system.

Personalized Model Adaptation. Following the unified student model distillation, *RefinedEdge* implements a personalized adaptation mechanism for each edge device. For each device i, we fine-tune the unified student model using its local dataset $\mathbf{X}_i$. The personalization process focuses on adapting the model to local data characteristics through reconstruction learning. This local adaptation process allows each edge device to specialize the unified student model according to its specific data patterns while maintaining the basic structure and knowledge inherited from the unified model.

D. Reciprocal Edge-Cloud Updating

The Reciprocal Edge-Cloud Updating module establishes a continuous learning cycle between cloud and edge devices to maintain model effectiveness in dynamic operational environments. This module consists of three primary components: (1) edge data aggregation, (2) cloud-based teacher model updating, and (3) unified and personalized model updating.

1) Edge Data Aggregation: Following the data aggregation mechanism established in Section IV-B1, we continuously collect and synchronize data from edge devices using the sliding window approach. This ensures that model updates are based on recent operational patterns while maintaining sufficient historical context for effective learning.

2) Cloud-based Teacher Model Updating: This process utilizes Reverse Knowledge Distillation, where personalized student models at the edge guide the update of the cloud-based teacher model. This approach ensures that the teacher model incorporates localized knowledge while maintaining its generalization capability. The update objective is formulated as:

$$\begin{aligned}\mathcal{L}_{update}^{T} = &\sum_{i=1}^{n} \beta_i \|\mathcal{M}_{S_i}(\mathbf{x}) - \mathcal{M}_{\mathcal{T}}(\mathbf{x})\|_2^2 \\ &+ (1 - \beta_i)\|\mathcal{M}_{T}(\mathbf{x}) - \mathbf{x}\|_2^2\end{aligned} \tag{15}$$

where β_i is the importance weight for device i. When F1-scores from edge devices are available, the importance weights are dynamically adjusted to give higher priority to better-performing devices. Otherwise, equal weights are assigned to ensure balanced contribution from all devices:

$$\beta_i = \begin{cases} \frac{\exp(\gamma F_i)}{\sum_{j=1}^{n} \exp(\gamma F_j)} & \text{if F1-scores available} \\ \frac{1}{n} & \text{otherwise} \end{cases} \tag{16}$$

where F_i is the F1-score of device i (if available), γ is a temperature parameter controlling the weight distribution, and n is the total number of edge devices. This weighting mechanism ensures that when performance metrics are available, devices with higher F1-scores contribute more significantly to the teacher model updating, while maintaining fair contribution when such metrics cannot be obtained.

3) Student Model Updating: The updating process consists of two stages:

a) Unified Student Model Update: The unified student model is updated using the new teacher model through a balanced objective:

$$\begin{aligned} \mathcal{L}_{update}^{S} =& \lambda_{kd} \|\mathcal{M}_{\mathcal{S}}(\mathbf{x}) - \mathbf{x}\|_2^2 \\ &+ (1 - \lambda_{kd}) \|\mathcal{M}_{\mathcal{S}}(\mathbf{x}) - \mathcal{M}_{T_{new}}(\mathbf{x})\|_2^2 \end{aligned} \tag{17}$$

where $\mathcal{M}_{T_{new}}$ is the updated teacher model.

b) Personalized Student Model Update: Each edge device's personalized model is updated using a device-specific objective:

$$\begin{aligned} \mathcal{L}_{update}^{S_i} =& \alpha_i \|\mathcal{M}_{S_i}(\mathbf{x}) - \mathbf{x}\|_2^2 \\ &+ (1 - \alpha_i) \|\mathcal{M}_{S_i}(\mathbf{x}) - \mathcal{M}_{T_{new}}(\mathbf{x})\|_2^2 \end{aligned} \tag{18}$$

and the adaptation coefficient α_i is learned through:

$$\alpha_i = \sigma(\phi_i) \tag{19}$$

where ϕ_i is a trainable parameter and σ is the sigmoid function.

The update process can be configured with a temporal schedule, typically set to 24 hours for daily updates to balance model freshness with computational efficiency.

This reciprocal learning module ensures continuous model improvement while maintaining computational efficiency and adaptation to local operational conditions. The two-stage update strategy balances global knowledge sharing with local specialization, enabling effective anomaly detection across diverse edge environments.

From a temporal perspective, the framework operates through a complete workflow involving four key cloud-edge transitions: (1) **Edge data accumulation and cloud aggregation:** Edge devices accumulate local data over time, which is then aggregated by the cloud for centralized processing. (2) **Cloud training and edge deployment:** The cloud completes teacher model training, pruning, knowledge distillation, and student model personalization, then transitions to edge devices for real-time detection. (3) **Edge feedback collection:** After operating for a predefined period (e.g., 24 hours), edge devices send new data and detection feedback to the cloud. (4) **Cloud model update and edge redeployment:** The cloud updates both teacher and student models based on the feedback, then redeploys the updated student models to edge devices for continued operation.

V. Evaluation

In this section, we address the following research questions:

- **RQ1:** How well does *RefinedEdge* perform in multivariate time series anomaly detection?
- **RQ2:** How do hyperparameters influence the performance of *RefinedEdge*?
- **RQ3:** How different update strategies influence the performance of *RefinedEdge*?
- **RQ4:** How effectively does the knowledge distillation preserve temporal dependency structures?
- **RQ5:** How is the generalization ability of *RefinedEdge*?

A. Experimental Setup

1) Dataset: We evaluate *RefinedEdge* on four real-world multivariate time series datasets: EdgeNode, Server Machine Dataset (SMD) [29], Mars Science Laboratory (MSL) [37], and Soil Moisture Active Passive (SMAP) [37].

The EdgeNode dataset, collected from a leading edge computing service provider, contains system monitoring data from 200 edge nodes, each collecting 25 system metrics. This industrial dataset represents our target application scenario, demonstrating the framework's effectiveness in real-world edge computing environments. The dataset encompasses diverse anomaly patterns that reflect real-world operational challenges in edge computing environments. These include transient system faults (point anomalies), performance degradation under varying workloads (contextual anomalies), and sustained resource contention scenarios (collective anomalies). The feature values exhibit a heavy-tailed, right-skewed distribution with significant positive skewness (2.68) and high kurtosis (9.49), indicating the presence of outliers and extreme values characteristic of real-world system monitoring data. The noise characteristics reveal an average signal-to-noise ratio of approximately 10-15 dB, corresponding to moderate noise levels typical of industrial monitoring systems. This noise profile originates from measurement uncertainties, network fluctuations, and environmental interference, ensuring that our evaluation results reflect realistic deployment conditions rather than idealized laboratory scenarios. Due to the non-disclosure agreement, we cannot make this dataset publicly available.

To ensure reproducibility and validate the framework's generalizability, we also evaluate our method on three public datasets: SMD, MSL, and SMAP. SMD consists of telemetry data from 28 server machines, each monitoring 38 system metrics including CPU usage, memory utilization, and network traffic. MSL contains 55 metrics from the Curiosity rover's sensors and onboard equipment during its mission on Mars. SMAP contains 25 metrics from the Soil Moisture Active Passive satellite's telemetry data, providing another spacecraft system perspective for evaluation. These diverse public datasets, covering different domains and data characteristics, help demonstrate *RefinedEdge*'s broad applicability beyond edge computing scenarios.

TABLE I
DATASET STATISTICS AND PREPROCESSING DETAILS

	EdgeNode	**SMD**	**MSL**	**SMAP**
Entity type	Edge computing device	Server machine	Spacecraft system	Spacecraft system
Number of entities	200	28	27	55
Number of metrics	25	38	55	25
Data interval	15 min	1 min	1 min	1 min
Anomaly proportion	5.18%	4.16%	10.72%	13.13%
Original train set (average)	7 d	17 d	1.5 d	1.7 d
Processed train set	5 d	1 d	1 d	1 d
Train size per entity	(480, 25)	(1440, 38)	(1440, 55)	(1440, 25)
Test set (average)	7 d	17 d	1.9 d	5.4 d

Table I presents the detailed statistics and preprocessing details of these datasets. To better simulate data-limited scenarios commonly encountered in edge computing environments, we processed the training durations across datasets. Specifically, we restricted the training sets of SMD, MSL, and SMAP to the last 1 day (1440 samples per entity), and EdgeNode to the last 5 days (480 samples per entity) due to its 15-minute sampling interval. This preprocessing better demonstrates the quick-start capability of our framework in data-limited scenarios.

The anomaly proportion varies significantly across datasets: SMAP shows the highest at 13.13%, followed by MSL at 10.72%, EdgeNode at 5.18%, and SMD at 4.16%. These variations reflect the distinct operational characteristics and anomaly patterns inherent to each domain.

2) Baselines: We compare *RefinedEdge* against several baselines:

- Cloud-Train (7 M): A large TimesNet [30] model deployed centrally in the cloud.
- Edge-Train (0.12 M): Compressed TimesNet models trained from scratch for each edge device.
- EWMA [38]: Exponentially Weighted Moving Average, a statistical method that detects anomalies by tracking weighted averages of historical values.
- SVM [39]: A support vector machine model that learns decision boundaries for anomaly detection.
- AE [40]: An unsupervised learning algorithm based on neural networks that aims to reconstruct input data through an encoder-decoder structure.
- OmniAnomaly [29]: A stochastic recurrent neural network designed for multivariate time-series anomaly detection that explicitly models both temporal dependencies and stochasticity in data.
- USAD [41]: An unsupervised anomaly detection model based on adversarial training and autoencoders.

3) Implementation Details: All experiments were cork. The cloud environment utilizes an NVIDIA GeForce RTX 4090 24GB GPU for model training and updates, while edge devices are simulated using a 2.6 GHz 6-Core Intel Core i7-9750H CPU. For Cloud-Train baselines, we use the PyTorch framework with recommended hyperparameters if available, or optimal configurations determined through hyperparameter search. The models are trained on aggregated data from all edge devices and evaluated on each edge device's test set. For TimesNet, the configuration includes a sequence length of 96, input dimensions determined by the metric dimensions of each dataset, model dimensions of 64, 8 attention heads, 3 encoder layers, and 1 decoder layer. Similar optimal configurations are used for CrossFormer and iTransformer in their respective experiments.

4) Evaluation Metrics: We employ the widely-used point-adjusted F1-score [42] as our primary evaluation metric. This metric introduces an adjustment mechanism: if any point within a true anomaly sequence is detected as anomalous (i.e., the predicted anomaly score exceeds a threshold), all points in that sequence are considered correctly detected. Meanwhile, points outside the anomaly sequences are evaluated normally. Based on this mechanism, we define the detection outcomes as follows:

- True Positive (TP): A point within a true anomaly sequence where any point in that sequence is detected as anomalous
- False Positive (FP): A point predicted as anomalous that does not fall within any true anomaly sequence
- False Negative (FN): A point within a true anomaly sequence where no point in that sequence is detected as anomalous

Based on these definitions, we calculate precision and recall as:

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN} \tag{20}$$

The F1-score, which balances precision and recall, is then computed as:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{21}$$

The F1-score is calculated at the entity level, where the anomaly threshold is optimized independently for each entity to achieve its best detection performance. This entity-level best F1-score reflects the theoretical best detection capability of the model. Unless otherwise specified, all F1-scores reported in this paper refer to this entity-level best F1-score.

Additionally, we measure model training time and inference latency to evaluate the computational efficiency of *RefinedEdge*.

B. RQ1: Performance of RefinedEdge

1) Model Performance without Updates: We evaluated two baseline training strategies: *One for all* (single model trained

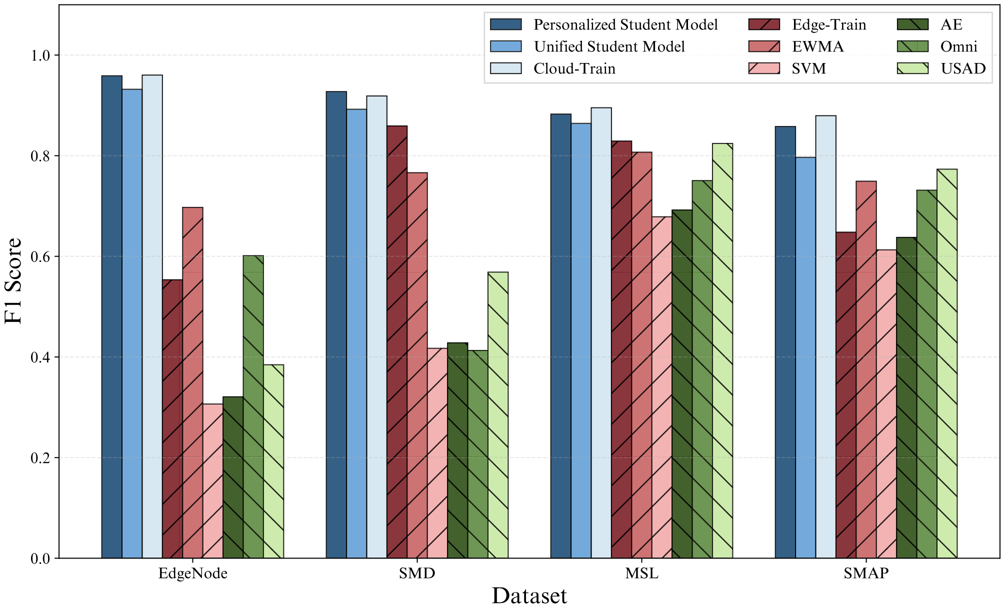


Fig. 2. F1-Score Comparison Across Methods and Datasets.

on aggregated data) and *One for one* (separate models per edge device). Since *One for one* consistently outperformed *One for all* across all methods and datasets in our experiments, we report only *One for one* results for fair comparison.

Fig. 2 presents the comprehensive F1-score comparison across different methods and datasets, where all methods are evaluated without additional update steps for fair comparison with baselines.

Our personalized student model (0.12 M parameters) achieves superior performance across all datasets: F1-scores of 0.9588, 0.9274, 0.8827, and 0.8580 on EdgeNode, SMD, MSL, and SMAP respectively, significantly outperforming edge-train baselines (0.5534, 0.8591, 0.8291, and 0.6480). Notably, on SMD, it even exceeds the large cloud-train model (7 M parameters) while using only 1.7% of its parameters. Note that while the figure includes the cloud-train model for reference, its large size renders it unsuitable for edge deployment. When considering only edge-compatible models, *RefinedEdge* consistently outperforms all baselines. The ablation of personalization shows a consistent but modest decrease in performance, highlighting the importance of personalization in edge deployment scenarios.

2) Efficiency Comparison: Table II summarizes the training and inference times using 96-point windows. While our student model's training involves knowledge distillation overhead, it achieves efficient cloud-side processing (16–25 ms per window) and acceptable edge inference times (220–310 ms per window). In contrast, edge-train requires significantly longer training times (2013–2701 ms per window), highlighting the advantage of our cloud-assisted approach.

3) Complexity Analysis: We provide a complexity analysis of *RefinedEdge* framework across cloud and edge environments. Let N denote the number of edge devices, P_T, and P_S represent the parameters of the teacher and student models respectively, and D represents the data size per training batch.

Computational Complexity: The cloud-side complexity is $O(P_T + N \cdot P_S)$ per iteration, significantly reduced from $O(N \cdot P_T)$ for independent training. Edge-side complexity is $O(P_S)$ for inference. With compression ratio $P_S/P_T \approx 0.017$, both loads remain manageable.

Communication Complexity: Edge models operate independently for real-time detection, requiring communication only during periodic updates (every 24 hours). The overhead includes data aggregation $O(N \cdot D)$, model deployment $O(N \cdot P_S)$, and periodic updates $O(N \cdot D_{update} + N \cdot P_S)$. This minimal communication burden ensures real-time performance while enabling cloud-assisted optimization. For scenarios with high-frequency sampling that may strain bandwidth, the framework can incorporate downsampling techniques or data compression methods to reduce transmission requirements while preserving essential temporal patterns.

C. RQ2: Hyperparameter Sensitivity

To investigate the effect of knowledge distillation weight (λ_{KD}) on model performance, we conduct experiments with different λ_{KD} values ranging from 0 to 1.0 on the EdgeNode dataset.

Fig. 3 reveals several key findings:

- The performance curve exhibits a bell-shaped pattern, with optimal performance achieved in the middle range of λ_{KD} values.
- The personalized student model reaches its peak performance (F1-score of 0.9588) at $\lambda_{KD} = 0.6$.
- Performance degradation is observed at both extremely low ($\lambda_{KD} < 0.2$) and high ($\lambda_{KD} > 0.8$) values, suggesting that either insufficient or excessive reliance on teacher guidance can be detrimental.
- When $\lambda_{KD} \rightarrow 0$, the student model primarily mimics the teacher's outputs without sufficient emphasis on

TABLE II
Training and Inference Time Comparison

Method	Training Time (ms/window)				Inference Time (ms/window)			
	EdgeNode	SMD	MSL	SMAP	EdgeNode	SMD	MSL	SMAP
Personalized Student Model (0.12 M)	*24.88*	*16.35*	*19.59*	*23.96*	309.00	226.28	223.64	301.53
Unified Student Model (0.12 M)	*21.95*	*14.10*	*17.69*	*22.74*	306.78	227.83	223.15	308.21
Cloud-Train (7 M)	*2.73*	*1.15*	*1.80*	*2.78*	*18.00*	*12.48*	*14.52*	*17.49*
Edge-Train (0.12 M)	2185.16	2643.08	2701.55	2013.93	175.81	221.25	226.15	159.16
EWMA	N/A†	N/A†	N/A†	N/A†	9.20	14.02	25.95	9.83
SVM	0.16	15.76	1.72	0.51	0.20	8.71	1.50	0.32
AE	18.04	15.61	15.12	17.89	1.62	1.21	1.03	1.09
Omni	69.62	64.43	66.28	63.49	16.02	17.48	15.21	12.77
USAD	36.96	40.60	41.38	37.40	0.34	0.34	0.45	0.32

†EWMA does not require training.
Underlined italic values indicate experiments involving large models executed in a cloud environment (GPU), while other values were obtained from edge environment (CPU) as described in Section V-A3.

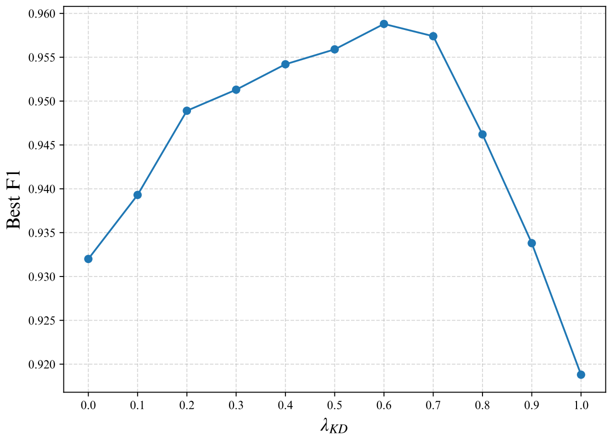


Fig. 3. Best F1-score vs λ_{KD} for Personalized Student Model on EdgeNode dataset.

developing its own reconstruction capabilities, limiting personalization potential.

- When $\lambda_{KD} \to 1$, the student model focuses excessively on reconstruction learning, potentially compromising knowledge transfer from the teacher model.

From an information-theoretic perspective, the observed bell-shaped performance curve can be understood through the information bottleneck principle [43]. The parameter λ_{KD} essentially controls the trade-off between information compression and detection accuracy. When λ_{KD} approaches 1, the student model emphasizes reconstruction learning, potentially retaining excessive input details including noise, which may violate the compression requirement of the information bottleneck. Conversely, when λ_{KD} approaches 0, the student model over-relies on teacher guidance, potentially losing sensitivity to local patterns and reducing detection accuracy. The optimal range of λ_{KD} (0.4-0.7) observed in our experiments suggests a balance where the student model effectively compresses teacher knowledge while maintaining detection capability for anomaly detection tasks, which aligns with the compression-prediction trade-off principle in information bottleneck theory.

TABLE III
Comparison of Update Strategies

Dataset	Reciprocal Update	No Update	Edge Update	Cloud Update
EdgeNode	**0.9683**	0.9588	0.9327	0.9604
SMD	**0.9564**	0.9274	0.8970	0.9316
SMAP	**0.8993**	0.8580	0.8385	0.8597

D. RQ3: Different Update Strategies

Table III compares different update strategies for handling evolving data patterns. All experiments were conducted with a strict 24-hour update cycle. Note that the MSL dataset was excluded from this experiment due to its short test duration. The results reveal several interesting findings:

- Edge-update strategy consistently shows degraded performance across all datasets. This degradation is particularly pronounced in SMD and SMAP, where F1-scores drop to 0.8970 and 0.8385 respectively. This underperformance can be attributed to the limited learning capacity of edge models and insufficient data at individual edge nodes.
- Cloud-update strategy shows better performance than edge-update strategy but still falls short of our reciprocal strategy, highlighting the benefits of combining both cloud and edge knowledge in the update process.
- For the EdgeNode dataset, update strategies show similar performance to no updating. Based on our analysis, this is because the EdgeNode dataset exhibits minimal pattern changes throughout the evaluation period, with no significant concept drift in the test set.
- For SMD and SMAP datasets, which contain concept drift and temporal variations in their test sets according to recent work [44], our reciprocal update strategy demonstrates substantial improvements. It achieves F1-scores of 0.9564 and 0.8993 respectively, outperforming both no updating and other update strategies. This demonstrates the effectiveness of our reciprocal updating mechanism in handling evolving data patterns.

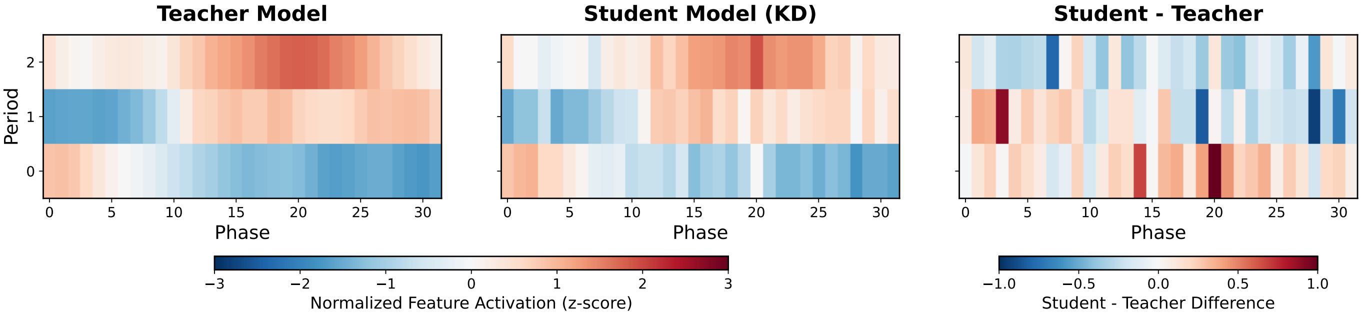


Fig. 4. Period-Phase 2D Feature Alignment between Teacher and Student Models. The horizontal axis represents the within-period position (phase), ranging from 0 to 31 for period length T=32. The vertical axis represents the period index, indicating different repetitions of the identified period pattern. Left: Teacher model activations; Middle: Student model activations; Right: Activation difference (Student - Teacher).

TABLE IV
FRAMEWORK GENERALIZATION RESULTS

Dataset	CrossFormer			iTransformer		
	Personalized Student Model	Edge-Train	Cloud-Train	Personalized Student Model	Edge-Train	Cloud-Train
EdgeNode	0.8571	0.4354	**0.8607**	0.8061	0.4046	**0.8344**
SMD	**0.8625**	0.8263	0.8607	**0.8413**	0.8274	0.8367
MSL	**0.8428**	0.7902	0.8288	**0.8342**	0.8076	0.8297
SMAP	**0.8505**	0.7473	0.8472	0.8344	0.7460	**0.8420**
Parameters	0.13 M	0.13 M	2.43 M	0.11 M	0.11 M	0.33 M

E. RQ4: Knowledge Distillation Effectiveness Analysis

Recent work shows that simple distributional alignment is insufficient to preserve temporal dependency structures in knowledge distillation [45]. Since TimesNet reshapes temporal windows into period-phase 2D maps, we evaluate structural preservation along both temporal axes.

We analyze the first TimesBlock (TimesNet's core processing unit) on the EdgeNode dataset using windows of length $w = 96$ with dominant period $T = 32$. The dominant period represents the most significant periodic component identified by TimesNet's FFT-based period detection mechanism. After averaging across all channels and applying joint z-score normalization, we obtain 2D maps H_T and H_S for teacher and student models. We propose Period-Phase Consistency (PPC) metrics:

$$PPC_{\text{intra}} = \frac{1}{R}\sum_{r=1}^{R} \text{corr}(H_T[r,:],\, H_S[r,:])\,, \tag{22}$$

$$PPC_{\text{inter}} = \frac{1}{C}\sum_{c=1}^{C} \text{corr}(H_T[:,c],\, H_S[:,c])\,. \tag{23}$$

where $\text{corr}(\cdot,\cdot)$ denotes the Pearson correlation coefficient. High PPC_{intra} indicates preserved within-period (phase) dependencies; high PPC_{inter} indicates preserved cross-period variation.

Fig. 4 visualizes representative teacher-student alignment. Across all test windows in the EdgeNode dataset (at optimal $\lambda_{KD} = 0.6$), we obtain average $PPC_{\text{intra}} = 0.9602$ and $PPC_{\text{inter}} = 0.9227$, demonstrating effective preservation of TimesNet's temporal dependency structures.

F. RQ5: Framework Generalization

To validate the generalizability of *RefinedEdge*, we extend the framework to diverse time series model architectures and anomaly detection paradigms. Specifically, we test two state-of-the-art Transformer-based forecasting models: CrossFormer [16] and iTransformer [17]. These models represent forecasting-based anomaly detection approaches where prediction errors serve as anomaly scores, in contrast to our primary implementation using TimesNet's reconstruction-based approach.

Table IV shows the performance of *RefinedEdge* applied to these forecasting-based models. The results demonstrate that *RefinedEdge* consistently improves model performance across different architectures and anomaly detection paradigms. Our personalized student models achieve comparable performance to cloud-train approaches while using significantly fewer parameters. More importantly, the framework substantially outperforms edge-train baselines across all datasets and model architectures, highlighting the effectiveness of our cloud-edge collaborative approach.

These results provide concrete empirical evidence that *RefinedEdge* is not limited to reconstruction-based CNN models like TimesNet but can effectively generalize to forecasting-based Transformer architectures. The framework demonstrates significant improvements, particularly in maintaining high accuracy with substantially reduced parameter counts. This validates that our model compression, knowledge distillation, and reciprocal updating mechanisms are truly model-agnostic and work effectively across different anomaly detection paradigms—whether using reconstruction errors or prediction errors as anomaly scores. The consistent performance gains across both CNN and Transformer architectures suggest that the framework's design principles could be extended to various neural network architectures for time series anomaly detection.

VI. Limitations and Future Work

While *RefinedEdge* demonstrates promising results in edge-cloud collaborative anomaly detection, several limitations warrant discussion and present opportunities for future research.

A. Limitations

Performance Under Extreme Data Noise: Our evaluation demonstrates robustness across typical operational noise levels, and theoretical analysis suggests the framework's inherent resilience to extreme noise conditions. The knowledge distillation process naturally provides denoising capabilities, as the teacher model learns from aggregated data across multiple edge devices, effectively averaging out device-specific noise artifacts. Additionally, the personalization component further enhances noise resilience by adapting to local signal characteristics while maintaining global knowledge from the teacher model.

Scalability to Large-Scale Edge Deployments: Our experiments involve a maximum of 200 edge devices, which may not represent the challenges of deployments with thousands of devices. Scaling to very large numbers could introduce computational bottlenecks as the teacher model training complexity scales with device numbers. However, these challenges can be effectively addressed through hierarchical cloud-edge architectures and device clustering strategies to distribute computational loads. These approaches can be naturally integrated into our existing framework without fundamental modifications.

Effect of Significant Data Heterogeneity: If local datasets are highly heterogeneous, the effectiveness of reverse knowledge distillation may weaken, as the cloud model struggles to accommodate diverse local patterns. Our current experiments use relatively homogeneous datasets, making reverse distillation more effective. For highly heterogeneous scenarios, grouping edge devices based on data similarity and assigning different cloud models to different groups could better capture diverse local characteristics.

B. Future Work

Based on the identified limitations, we envision several promising research directions that can significantly advance edge-cloud collaborative anomaly detection frameworks.

Enhanced Robustness and Scalability: To address the challenges of extreme noise and large-scale deployments, future work should develop intelligent adaptive mechanisms. This includes implementing change detection algorithms that trigger updates based on data drift, and creating robust distillation techniques that maintain effectiveness under varying noise conditions. Furthermore, hierarchical cloud-edge architectures with regional aggregation nodes could distribute computational loads while enabling federated learning principles to reduce communication overhead in thousand-device deployments.

Advanced Model Optimization: Building upon our ensemble pruning approach, future research could explore more sophisticated compression strategies through adaptive weight assignment, where the importance of different pruning criteria is dynamically learned based on specific architectures and datasets. Comparative studies of various pruning integration approaches would provide valuable insights for optimizing edge deployment efficiency across diverse hardware configurations.

Privacy-Preserving Collaborative Learning: As edge computing increasingly handles sensitive data, developing privacy-preserving mechanisms becomes crucial. Future work should explore differential privacy techniques for controlled noise injection, secure aggregation protocols that enable cloud learning without exposing individual device data, and federated learning adaptations where only model parameters or gradients are shared. These advances would expand the framework's applicability to privacy-sensitive domains such as healthcare monitoring and financial systems while preserving the benefits of collaborative learning.

VII. Conclusion

The *RefinedEdge* framework effectively addresses the challenges of implementing multivariate time series anomaly detection in resource-constrained edge computing environments. By leveraging techniques such as model compression and knowledge refinement, the framework maintains high detection accuracy even with heavily compressed models. The reciprocal updating between edge devices and cloud resources ensures ongoing model relevance through continuous adaptation to changing data patterns. The empirical results affirm the framework's capability to enhance operational efficiency in edge scenarios, promising extensive applicability across various domains requiring real-time data processing and anomaly detection.

References

[1] J. Wang, M. K. Lim, C. Wang, and M. Tseng, "The evolution of the internet of things (iot) over the past 20 years," *Comput. Ind. Eng.*, vol. 155, p. 107174, 2021.

[2] J. Yao, S. Zhang, Y. Yao, F. Wang, J. Ma, J. Zhang, Y. Chu, L. Ji, K. Jia, T. Shen, A. Wu, F. Zhang, Z. Tan, K. Kuang, C. Wu, F. Wu, J. Zhou, and H. Yang, "Edge-cloud polarization and collaboration: A comprehensive survey for ai," *IEEE Transactions on Knowledge and Data Engineering*, p. 1–1, 2022.

[3] M. Yang and J. Zhang, "Data anomaly detection in the internet of things: A review of current trends and research challenges," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 9, 2023.

[4] M. S. Islam, W. Pourmajidi, L. Zhang, J. Steinbacher, T. Erwin, and A. V. Miranskyy, "Anomaly detection in a large-scale cloud platform," in *43rd IEEE/ACM International Conference on Software Engineering: Software Engineering in Practice, ICSE (SEIP) 2021, Madrid, Spain, May 25-28, 2021.* IEEE, 2021, pp. 150–159.

[5] S. Duan, D. Wang, J. Ren, F. Lyu, Y. Zhang, H. Wu, and X. Shen, "Distributed artificial intelligence empowered by end-edge-cloud computing: A survey," *IEEE Commun. Surv. Tutorials*, vol. 25, no. 1, pp. 591–624, 2023.

[6] Y. Liu, H. Zhang, C. Li, X. Huang, J. Wang, and M. Long, "Timer: Generative pre-trained transformers are large time series models," in *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024.* OpenReview.net, 2024.

[7] L. Ren, Z. Jia, Y. Laili, and D. Huang, "Deep learning for time-series prediction in iiot: Progress, challenges, and prospects," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 35, no. 11, pp. 15 072–15 091, 2024.

[8] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, and M. S. Hossain, "Deep anomaly detection for time-series data in industrial iot: A communication-efficient on-device federated learning approach," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6348–6358, 2021.

[9] X. Yu, X. Yang, Q. Tan, C. Shan, and Z. Lv, "An edge computing based anomaly detection method in iot industrial sustainability," *Appl. Soft Comput.*, vol. 128, p. 109486, 2022.

[10] Y. Wang, C. Yang, S. Lan, L. Zhu, and Y. Zhang, "End-edge-cloud collaborative computing for deep learning: A comprehensive survey," *IEEE Communications Surveys &amp; Tutorials*, p. 1–1, 2024.

[11] H. Qi, F. Ren, L. Wang, P. Jiang, S. Wan, and X. Deng, "Multi-compression scale DNN inference acceleration based on cloud-edge-end collaboration," *ACM Trans. Embed. Comput. Syst.*, vol. 23, no. 1, pp. 16:1–16:25, 2024.

[12] R. Y. . H. H. . Y. X. . B. X. . W. . W. Zhang, "Efficient intrusion detection toward iot networks using cloud-edge collaboration," *Computer Networks*, vol. 228, p. 109724, jun 2023.

[13] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *Journal of the American Statistical Association*, vol. 65, no. 332, pp. 1509–1526, 1970.

[14] J. M. Lucas and M. S. Saccucci, "Exponentially weighted moving average control schemes: properties and enhancements," *Technometrics*, vol. 32, no. 1, pp. 1–12, 1990.

[15] M. Moshtaghi, J. C. Bezdek, C. Leckie, S. Karunasekera, and M. Palaniswami, "Evolving fuzzy rules for anomaly detection in data streams," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 3, pp. 688–700, 2015.

[16] Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting," in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[17] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "itransformer: Inverted transformers are effective for time series forecasting," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[18] Y. Yao, J. Ma, and Y. Ye, "Regularizing autoencoders with wavelet transform for sequence anomaly detection," *Pattern Recognit.*, vol. 134, p. 109084, 2023.

[19] Z. Li, Y. Zhao, J. Han, Y. Su, R. Jiao, X. Wen, and D. Pei, "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, F. Zhu, B. C. Ooi, and C. Miao, Eds. ACM, 2021, pp. 3220–3230.

[20] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S. Ng, "MAD-GAN: multivariate anomaly detection for time series data with generative adversarial networks," in *Artificial Neural Networks and Machine Learning - ICANN 2019: Text and Time Series - 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17-19, 2019, Proceedings, Part IV*, ser. Lecture Notes in Computer Science, I. V. Tetko, V. Kurková, P. Karpov, and F. J. Theis, Eds., vol. 11730. Springer, 2019, pp. 703–716.

[21] G. Woo, C. Liu, A. Kumar, C. Xiong, S. Savarese, and D. Sahoo, "Unified training of universal time series forecasting transformers," in *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[22] F. Meng, H. Cheng, K. Li, H. Luo, X. Guo, G. Lu, and X. Sun, "Pruning filter in filter," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[23] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," *CoRR*, vol. abs/2103.13630, 2021.

[24] L. Chen, X. Jiang, X. Liu, and Z. Zhou, "Logarithmic norm regularized low-rank factorization for matrix and tensor completion," *IEEE Trans. Image Process.*, vol. 30, pp. 3434–3449, 2021.

[25] C. G. . P. Y. . Y. C. . Z. W. . Y. Wang, "An edge-cloud collaboration architecture for pattern anomaly detection of time series in wireless sensor networks," *Complex &amp; Intelligent Systems*, vol. 7, no. 5, p. 2453–2468, jun 2021.

[26] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, and M. S. Hossain, "Deep anomaly detection for time-series data in industrial iot: A communication-efficient on-device federated learning approach," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6348–6358, 2021.

[27] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," *ACM Comput. Surv.*, vol. 54, no. 3, pp. 56:1–56:33, 2022.

[28] Z. Z. Darban, G. I. Webb, S. Pan, C. Aggarwal, and M. Salehi, "Deep learning for time series anomaly detection: A survey," *ACM Comput. Surv.*, vol. 57, no. 1, pp. 15:1–15:42, 2025.

[29] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi, and G. Karypis, Eds. ACM, 2019, pp. 2828–2837.

[30] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[31] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowl. Inf. Syst.*, vol. 51, no. 2, pp. 339–367, 2017.

[32] H. Cheng, M. Zhang, and J. Q. Shi, "A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 10 558–10 578, 2024.

[33] G. Fang, X. Ma, M. Song, M. Bi Mi, and X. Wang, "Depgraph: Towards any structural pruning," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 16 091–16 101.

[34] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," *CoRR*, vol. abs/1506.02626, 2015.

[35] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 11 264–11 272.

[36] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 2755–2763.

[37] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Söderström, "Detecting spacecraft anomalies using lstms and non-parametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Y. Guo and F. Farooq, Eds. ACM, 2018, pp. 387–395.

[38] S. W. Roberts, "Control chart tests based on geometric moving averages," *Technometrics*, vol. 42, no. 1, pp. 97–101, 2000.

[39] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, 2001.

[40] A. Ng *et al.*, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[41] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: unsupervised anomaly detection on multivariate time series," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, Eds. ACM, 2020, pp. 3395–3404.

[42] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, J. Chen, Z. Wang, and H. Qiao, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, P. Champin, F. Gandon, M. Lalmas, and P. G. Ipeirotis, Eds. ACM, 2018, pp. 187–196.

[43] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," *arXiv preprint physics/0004057*, 2000.

[44] D. Kim, S. Park, and J. Choo, "When model meets new normals: Test-time adaptation for unsupervised time-series anomaly detection," in *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, M. J. Wooldridge, J. G. Dy, and S. Natarajan, Eds. AAAI Press, 2024, pp. 13 113–13 121.

[45] M. Farhadi and Y. Yang, "TKD: temporal knowledge distillation for active perception," in *IEEE Winter Conference on Applications of*

Computer Vision, WACV 2020, Snowmass Village, CO, USA, March 1-5, 2020. IEEE, 2020, pp. 942–951.

Shenglin Zhang *(Member, IEEE)* received the BS degree in network engineering from the School of Computer Science and Technology, Xidian University, Xi'an, China, in 2012, and the PhD degree in computer science from Tsinghua University, Beijing, China, in 2017. He is currently an associate professor with the College of Software, Nankai University, Tianjin, China. His current research interests include failure detection, diagnosis, and prediction for service management.

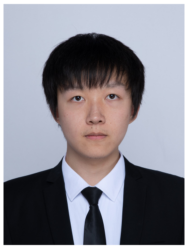
Jiacheng Zhang received the bachelor's degree in software engineering from Nankai University, Tianjin, China, in 2023. He is currently working toward the master's degree with the College of Software, Nankai University. His research interests include anomaly detection and large-scale time series models.

Guohua Liu graduated from Zhejiang University of Technology, majoring in Computer Science and Technology. He also obtained an Executive MBA degree from Zhejiang University. He is currently the general manager of the Private Cloud Department of Alibaba Cloud Computing Ltd.

Shiqi Chen received the bachelor's degree in software engineering from Nankai University, Tianjin, China, in 2024. He is currently working toward the master's degree with the College of Software, Nankai University. His research interests include anomaly detection and large-scale time series models.

Chenyu Zhao received the master's degree in software engineering from Nankai University, Tianjin, China, in 2023. Her research interests are focused on anomaly detection and root cause analysis.

Minghua Ma *(Member, IEEE)* received the PhD degree from Tsinghua University, in 2021. He is a senior researcher at Microsoft. His current research interests include cloud intelligence/AIOps.



Yutong Chen is currently working toward the bachelor's degree with the College of Software, Nankai University, Tianjin, China. His main research interests include anomaly detection and failure diagnosis.

Yongqian Sun *(Member, IEEE)* received the BS degree in statistical specialty from Northwestern Polytechnical University, Xi'an, China, in 2012, and the PhD degree in computer science from Tsinghua University, Beijing, China, in 2018. He is currently an associate professor with the College of Software, Nankai University, Tianjin, China. His research focuses on AIOps and service computing, using technologies like ML, AI, and LLM for anomaly detection and fault diagnosis to improve service quality.

Dan Pei *(Senior Member, IEEE)* received the BE and MS degrees in computer science from the Department of Computer Science and Technology, Tsinghua University, in 1997 and 2000, respectively, and the PhD degree in computer science from the Computer Science Department, University of California, Los Angeles (UCLA), in 2005. He is currently an associate professor with the Department of Computer Science and Technology, Tsinghua University. His research interests include network and service management in general. He is a senior member of the ACM.