Journal of Computer Technology and Software

ISSN:2998-2383

Vol. 3, No. 8, 2024

Federated Meta-Learning for Node-Level Failure Detection in Heterogeneous Distributed Systems

Minggu Wei
University of Saskatchewan, Saskatoon, Canada
weiowengu@gmail.com

Abstract: This paper addresses the challenges of data heterogeneity, privacy protection, and task personalization in server node failure detection within distributed systems. It proposes an intelligent detection algorithm framework that integrates federated learning and meta-learning. The method uses a task-adaptive meta-learning mechanism to build a transferable meta-model. This enables fast adaptation to different node failure patterns and improves generalization. At the same time, a personalized aggregation strategy is introduced to dynamically adjust the model parameter updates based on the local features of each node. This enhances the personalization of local models and improves overall detection accuracy. During model training, data remain stored locally to preserve privacy. This avoids the data leakage risks of traditional centralized learning. The method also enables node-level adaptive optimization while maintaining global collaboration. A series of experiments, including comparative studies, ablation tests, and robustness evaluations, are designed to validate the effectiveness and advantages of the proposed method from multiple perspectives. The results show that compared to mainstream federated learning methods, the proposed model achieves significant improvements in Accuracy, Precision, and Recall. It performs especially well under complex scenarios such as Non-IID data and inactive nodes. These findings demonstrate the method's strong stability, adaptability, and practical potential for large-scale distributed environments.

Keywords: Server node failure detection; federated optimization; personalized modeling; task generalization

1. Introduction

With the rapid development of information technology, distributed systems have become central to modern computing architectures, especially in emerging areas such as cloud computing, edge computing, and the Internet of Things[1,2]. Large-scale deployment of server nodes has become common. However, failures of server nodes remain a critical factor that limits system reliability, stability, and service continuity. When a server node fails, system performance may drop significantly or even lead to service interruptions, causing serious losses to business operations. Therefore, building an efficient and accurate server node failure detection mechanism has become a vital task in maintaining and managing distributed systems[3.4].

Traditional centralized failure detection methods show clear drawbacks under the growing scale of data and system complexity[5]. These include slow response, high communication overhead, and overload at central nodes. New approaches and techniques are urgently needed to address these issues.

Federated learning, as an emerging paradigm in distributed machine learning, enables collaborative model training without sharing raw data. It offers an effective path to address challenges in data privacy, communication efficiency, and system scalability. In server node failure detection scenarios, each node typically holds heterogeneous and widely distributed monitoring data. Directly uploading these data to a central server is neither practical nor safe due to privacy risks. Federated learning allows each node to train models locally and improves the global model through parameter aggregation. This supports efficient collaborative detection while preserving privacy, greatly enhancing the usability and scalability of failure detection systems[6].

However, federated learning still faces challenges in real-world applications. Problems such as non-independent and identically distributed data among nodes, differences in training capabilities, and slow model convergence often reduce its effectiveness. Meta-learning, a learning strategy that has developed rapidly in recent years, focuses on learning to learn. It accumulates knowledge across tasks to enable models to quickly adapt to new ones. Integrating meta-learning into federated learning can significantly enhance the generalization and adaptability of models across nodes. It also improves detection accuracy under limited samples and distribution differences. This offers a new research direction for failure detection in heterogeneous server node environments[7].

The combination of federated learning and meta-learning not only addresses the limitations of each method but also brings intelligence and personalization to server node failure detection. Under this framework, each node can learn local failure features and also acquire generalizable knowledge from the global model. This forms a detection mechanism that balances local precision with global insight. It effectively enhances the model's response to novel or rare failures. It also helps identify abnormal conditions before faults spread,

reducing system risks, improving maintenance efficiency, and meeting the demands of modern distributed systems for high availability and automated management.

Therefore, research on failure detection algorithms for server nodes based on the combination of federated learning and meta-learning holds significant theoretical and practical value. On the one hand, it advances the application of distributed intelligent learning technologies in system maintenance. It provides a new paradigm for handling large-scale heterogeneous data. On the other hand, it offers intelligent, efficient, and reliable technical support for server node failure prediction and emergency response systems. This approach is applicable in critical areas such as finance, energy, transportation, and healthcare. It contributes to improving the autonomy and security of national information systems.

2. Related work

2.1 Federated Learning

Federated learning is a distributed machine learning framework designed to address issues of data privacy and collaborative computation. In traditional centralized learning, all participating nodes must upload local data to a central server for unified modeling[8,9]. This approach carries risks of data leakage and may lead to high communication overhead and computational bottlenecks. Federated learning allows each node to train models independently on local data. Only model parameters or gradients are uploaded to the server for aggregation and updating[10]. This mechanism avoids the centralized transfer of raw data. It shows strong adaptability and flexibility in multi-source heterogeneous data scenarios. It is particularly suitable for domains with strict privacy requirements, such as finance, healthcare, and the industrial internet. As the demand for data security and intelligent collaboration continues to grow, federated learning is becoming a key supporting technology in distributed intelligent systems[11].

In the context of server node failure detection, the advantages of federated learning are especially evident. Server nodes are widely distributed, and their data are heterogeneous and unevenly sampled. A traditional unified model often fails to adapt to these differences. Federated learning enables local modeling at each node, which preserves personalized information[12]. At the same time, the iterative optimization of the global model improves overall detection performance. In real-world systems, some nodes may have unstable network connections or limited resources and cannot participate continuously in training. Federated learning supports dynamic participation through its flexible mechanisms. This ensures the stability and continuity of model training. As a result, federated learning is a preferred technical approach for building scalable, robust, and distribution-aware failure detection systems[13].

Although federated learning has shown promising applications in many fields, it still faces challenges in the specific task of server node failure detection. First, large differences in data distribution among nodes may affect model generalization and slow down convergence[14,15]. Second, uneven computational resources and varying willingness to participate can lead to instability or even failure during training. In addition, issues such as setting aggregation weights, controlling communication frequency, and designing security mechanisms require further optimization in real applications. Therefore, to extend the use of federated learning in failure detection, it is necessary to integrate other advanced learning paradigms. These may include meta-learning and adaptive optimization strategies. Such combinations can fully leverage the collaborative and intelligent potential of federated learning in heterogeneous distributed environments.

2.2 Meta-Learning

Meta-learning is a machine-learning approach aimed at improving model adaptability and generalization[16]. Its core idea is to train a model across multiple tasks so that it can quickly adapt to new tasks. In traditional supervised learning, models often rely on large amounts of labeled data to perform well[17]. This is impractical in certain tasks or environments, especially when samples are scarce or data distribution changes frequently. In such scenarios, the efficiency and accuracy of traditional methods are limited. Meta-learning extracts shared knowledge structures from multiple learning tasks. This allows the model to rapidly fine-tune with only a few samples when faced with new tasks[18]. It significantly improves learning efficiency and generalization. This ability to "learn to learn" is highly valuable in many real-world applications. It shows strong flexibility and adaptability, especially when intelligent systems face dynamic environments and personalized requirements[19].

In the task of server node failure detection, the introduction of meta-learning provides a new modeling approach. Server nodes differ significantly in operating environments, business logic, and hardware configurations. These differences lead to highly heterogeneous failure data distributions. Such inconsistency reduces the generalization of models across nodes and lowers the accuracy and robustness of the overall detection system. Meta-learning can conduct cross-task training over multiple failure detection tasks from different nodes. It extracts common features and builds a meta-model with fast adaptation ability[20]. This model can quickly adjust when encountering new nodes or new failure patterns. It improves generalization and offers clear advantages over traditional methods in situations like cold starts and limited samples.

Despite theoretical and algorithmic advances, applying meta-learning to server node failure detection still faces technical challenges. First, meta-learning relies on proper task partitioning. In real systems, the classification of failure types is often ambiguous and uncertain, which increases the difficulty of constructing meta-tasks[21,22]. Second, training meta-learning models is resource-intensive. When dealing with multiple tasks and high-dimensional parameter spaces, the demand for computational resources is high. In addition, differences in task similarity among nodes can affect the transfer efficiency of the meta-model. Therefore, for practical applications, it is essential to design efficient task generation mechanisms, optimize model update strategies, and manage training resource consumption. These are key steps to improving the usability of meta-learning in failure detection.

3. Method

This study proposes a server node failure detection method that integrates Federated Learning (FL) and Meta-Learning (ML). The goal is to address the challenges of strong data heterogeneity and high personalization demands in distributed environments. The first major innovation lies in the introduction of a task-adaptive meta-learning mechanism. This mechanism builds a fast and transferable meta-model. It enables rapid fine-tuning and efficient generalization when facing diverse failure data from different server nodes. This significantly enhances the model's ability to adapt to node-specific differences. The second innovation is the design of a Personalized Aggregation Strategy (PAS). Unlike the unified model aggregation in traditional FL, this strategy dynamically adjusts aggregation weights based on each node's feature distribution and training feedback. It strengthens local detection performance and personalized representation while maintaining global collaboration. It ensures that the model balances global consistency with local sensitivity. The overall framework retains the privacy protection and distributed coordination advantages of FL. At the same time, it incorporates the fast adaptation capabilities of ML. This provides an efficient, intelligent, and scalable solution for server failure detection in complex and heterogeneous environments. The architecture of the overall model is illustrated in Figure 1.

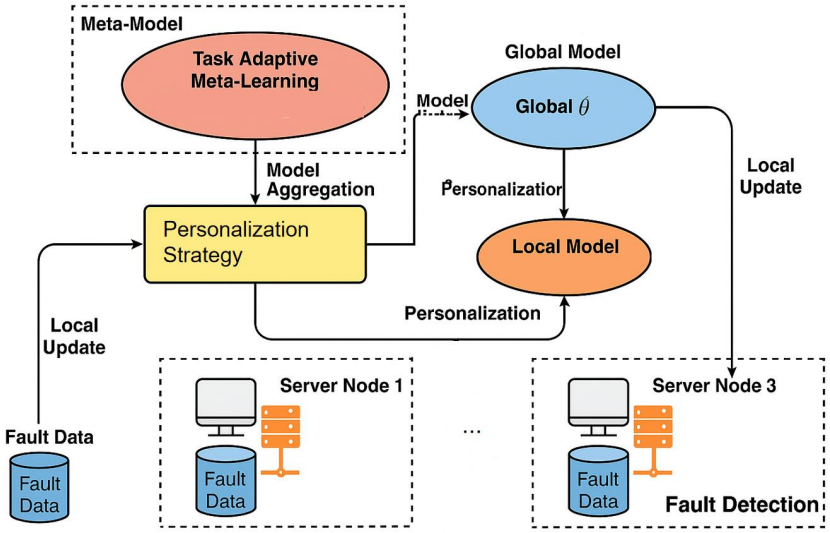

Figure 1. Overall model architecture diagram

3.1 Meta-learning mechanism based on task adaptability

In this study, we introduced a meta-learning mechanism based on task adaptability to enhance the model's ability to generalize and adapt in heterogeneous server node fault detection tasks. This mechanism is grounded in the principles of multi-task learning, where each server node is treated as a distinct task due to differences in data distribution, operational behavior, and failure patterns. By optimizing across multiple such tasks simultaneously, the approach aims to learn a model initialization that captures shared knowledge across tasks. This initialization serves as a foundation from which the model can rapidly adapt to the specific characteristics of new or unseen nodes with minimal computational effort.

The core of the meta-learning strategy lies in the optimization of a meta-objective function. This function is designed to enable the model to achieve fast convergence on new tasks through just a few gradient steps, using limited data. It effectively equips the model with a form of inductive bias that reflects commonalities across different server nodes, while maintaining flexibility for task-specific adaptation. The structure and workflow of this meta-learning module are illustrated in Figure 2, which outlines the relationship between task-level training, meta-optimization, and model parameter updates.

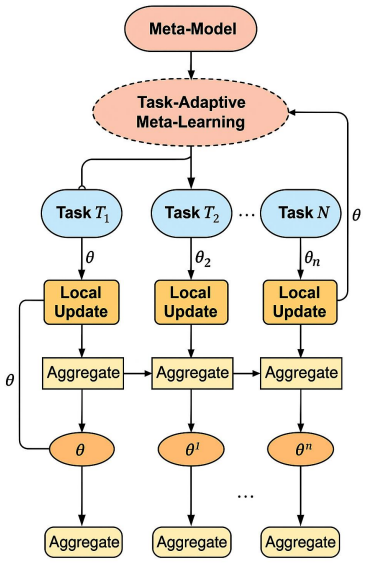

Figure 2. Meta-learning mechanism module architecture based on task adaptability

Specifically, let the task set be $T = \{T_1, T_2, ..., T_N\}$, each task corresponds to a fault detection subtask of a server node. We define the model parameter as θ, and the goal is to learn a shared initialization parameter $\theta*$ on multiple tasks so that it has good migration performance when facing new tasks.

For each task T_i, we first perform one or more gradient updates on the local training dataset D_i to obtain the task-specific model parameters θ'_i:

$$\theta'_i = \theta - \alpha \nabla_\theta L_{T_i}(\theta)$$

Where α is the learning rate and $L_{T_i}(\theta)$ represents the loss function on task T_i. Subsequently, the performance of the task-specific parameters is fed back to the meta-model to update the shared initialization parameter θ. The meta-update process is as follows:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{i=1}^{N} L_{T_i}(\theta'_i)$$

Among them, β is the meta-learning rate. This process realizes the transfer and sharing of knowledge between tasks, thereby building a meta-model that can quickly adapt to the new node failure distribution.

To further enhance task adaptability, we introduced a regularization mechanism in the model optimization process to guide the model parameters to maintain consistency with the global optimal direction when the meta-task is updated. Specifically, a parameter offset term is added to the meta-loss, and the optimization objective becomes:

$$L_{meta} = \sum_{i=1}^{N} L_{T_i}(\theta'_i) + \lambda \| \theta'_i - \theta \|^2$$

Among them, λ controls the weight of the regularization term, which helps prevent overfitting between tasks and maintain parameter stability. In addition, we also consider the impact of task importance on meta-updates and use the weight coefficient w_i to reflect the representativeness and influence of task T_i. The optimization objective is further expanded to:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{i=1}^{N} w_i L_{T_i}(\theta'_i)$$

This task-weighted optimization method effectively enhances the model's focus on core tasks while retaining its adaptability to edge tasks.

Combining the above mechanisms, task-adaptive meta-learning not only improves the model's ability to quickly learn heterogeneous node fault characteristics but also strengthens the flexibility of the model's personalized expression while maintaining overall generalization capabilities. By constructing a unified optimization path between tasks, this method provides a core learning engine with global transferability and local adaptability for fault detection under the federated learning framework.

3.2 Personalized Aggregation Strategy

To further enhance the personalized performance of the model on each server node, this study designs a Personalized Aggregation Strategy (PAS) to replace the traditional global model parameter aggregation used in federated learning. The core objective of this strategy is to build a local model with personalized features for each node. It considers task characteristics, adaptive learning capabilities, and differences in data distribution. This leads to higher detection accuracy and improved robustness. During the federated optimization process, we do not directly use the averaging strategy for model merging. Instead, we apply personalized weighting to adjust each node's contribution to the aggregation. This enables the global model to achieve both generalization and customization. Its module architecture is shown in Figure 3.

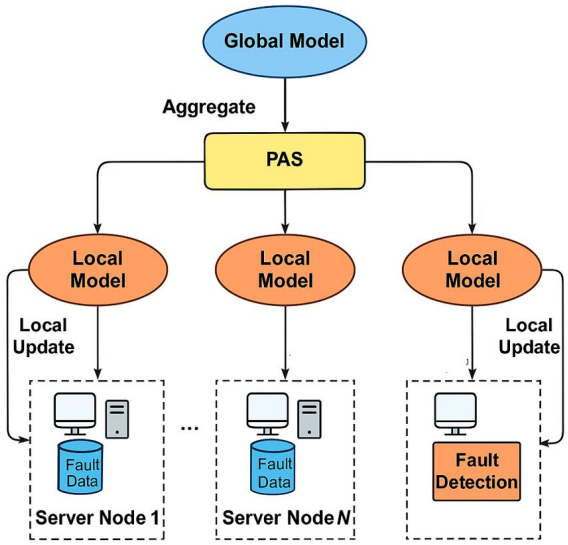

Figure 3. PAS module architecture

Assume that the local model parameter of the jth node is θ_j^t, its training loss is $L_j(\theta_j^t)$, and the aggregation goal is to build a customized global update based on the task performance of each node and the weight distribution coefficient w_j. The basic form of the personalized aggregation strategy is defined as follows:

$$L_j^{persional} = L_j(\theta_j^t) + \mu \| \theta_j^t - \theta^{t+1} \|^2$$

Where μ is a hyperparameter that adjusts the consistency between the local model and the global model.

During the parameter update process, each node fine-tunes according to its customized loss function and uploads the optimized model parameters to the center. The server then completes the next round of aggregation based on the personalized strategy. To ensure the stability and fairness of weight distribution, we introduce a dynamic weighting

mechanism based on gradient similarity, which is defined as follows:

$$w_j = \frac{\exp(-\| \nabla \theta_j L_j - \nabla_\theta L_{global} \|^2)}{\sum_{k=1}^{N} \exp(-\| \nabla \theta_k L_k - \nabla_\theta L_{global} \|^2)}$$

This mechanism enables nodes whose gradient directions are more consistent with the global objective to obtain greater weights, thereby enhancing the convergence and performance stability of the overall model.

In addition, considering the different task complexity and model expression capabilities of each node, we designed a personalized scheduling strategy to dynamically choose whether to use the global model or the local fine-tuned model for the final prediction. The final hybrid output model is as follows:

$$\theta_j^{final} = \lambda_j \theta_j^t + (1-\lambda_j)\theta_j^{t+1}$$

$\lambda_j \in [0,1]$ controls the degree of personalization and can be adjusted adaptively through the performance of the validation set. This strategy not only improves the generalization ability of the model in a non-independent and identically distributed (Non-IID) data environment but also effectively solves the different requirements of different nodes on the degree of model sharing, thus achieving a more robust and adaptable distributed fault detection system.

4. Experimental Results

4.1 Dataset

This study uses the Alibaba Cluster Trace 2018 dataset as the validation source for the server node failure detection algorithm. The dataset was collected from a real large-scale distributed system. It contains server operation data from Alibaba's production environment and reflects high levels of realism and complexity. The data include machine status, resource usage (such as CPU, memory, and disk), task scheduling information, and node-level event logs. These provide rich feature dimensions and behavioral patterns for failure detection tasks.

A key characteristic of this dataset is its time series nature and typical non-independent and identically distributed (Non-IID) properties. Different server nodes show varying load states and behavior patterns over time. The dataset includes a portion of labeled failure events, which can be used to train and evaluate anomaly detection and prediction models. The data distribution is highly imbalanced. Most periods represent normal states, while only a few indicate failures. This poses challenges to model robustness and detection capability and better reflects real-world scenarios.

To support the distributed detection framework of federated learning and meta-learning, this study divides the dataset into multiple subsets. Each subset corresponds to an individual server node. This partitioning preserves the personalized operational characteristics of each node. It also follows the data isolation principle in distributed training. It helps simulate the independence and heterogeneity of nodes in real environments. This allows for the effective evaluation of personalized aggregation and task adaptation mechanisms. The dataset is widely representative in cloud computing and resource scheduling research and serves as an ideal foundation for node-level failure detection studies.

4.2 Experimental setup

In the experimental setup of this study, a distributed simulation environment was built based on the Alibaba Cluster Trace 2018 dataset. The goal is to evaluate the performance of the proposed server node failure detection algorithm that combines federated learning and meta-learning. The data were divided into multiple subsets, with each subset representing a server node. This simulates the heterogeneity and isolation of data in real distributed systems.

During each round of federated training, each node performs local model training. Parameter exchange with the server is conducted based on predefined communication rounds and aggregation intervals. All models use the same initial architecture and hyperparameter configuration. This ensures fairness in comparison. The experiments were conducted in a Python environment using PyTorch as the deep learning framework. Federated communication was implemented through a custom lightweight message-passing module. The main hardware configuration includes multi-core CPUs and NVIDIA GPUs. Detailed specifications are shown in the table below. Its detailed configuration is shown in Table 1.

Table 1: Specific parameter diagram

Parameter name	Setting Value
Number of Nodes	20 simulated server nodes
Model Architecture	3-layer MLP (input-128-64-output)
Loss Function	Cross Entropy
Optimizer	Adam
Learning Rate	0.001
Meta-Learning Rate	0.0005
Communication Rounds	100
Batch Size	64
Hardware	Intel Xeon CPU, NVIDIA Tesla V100 GPU

4.3 Experimental Results

1) Comparative experimental results

This paper first gives the comparative experimental results, as shown in Table 2.

Table2: Comparative experimental results

Method	Accuracy	Precision	Recall
FedAvg[23]	86.9	85.2	82.1
FedProx[24]	88.3	86.1	83.4
PFedMe[25]	89.6	87.8	85.0
Ditto[26]	90.4	88.7	86.2
Ours	93.1	91.3	89.6

As shown in the comparative results in Table 2, the proposed method, which combines the Personalized Aggregation Strategy and Task-Adaptive Meta-Learning, achieves the best performance in the task of server node failure detection. It consistently outperforms existing mainstream federated learning models. In particular, the method reaches 93.1% in Accuracy, which is 6.2 percentage points higher than the baseline FedAvg. This clearly demonstrates the superior modeling capability and fault recognition accuracy of our approach in complex and heterogeneous environments.

Further analysis of the Precision metric shows that our method achieves 91.3%, significantly higher than 85.2% for FedAvg and 86.1% for FedProx. This indicates that the model is more effective in distinguishing between normal and abnormal states when identifying faulty samples, thus reducing the false positive rate. In server failure detection, false alarms can lead to unnecessary resource scheduling and even system intervention. Therefore, higher precision is critical for practical system deployment. It highlights the practicality and stability of our method in real-world applications.

Regarding Recall, our method also achieves an excellent score of 89.6%, outperforming all comparison methods. This improvement means the model has greater sensitivity in capturing potential faults. It can effectively identify more real failure events. Higher recall is especially important in production environments, as missed detections may lead to system crashes or safety risks. The task-adaptive meta-learning strategy we propose successfully enhances the model's generalization and fault response ability across diverse node data.

Overall, our method shows clear advantages across all three key metrics. This indicates that the combination of personalization and rapid adaptation is an effective path to improving federated fault detection performance in multi-node, heterogeneous, and Non-IID data settings. Compared with traditional unified modeling approaches, our method better balances global knowledge sharing and local node differences. It demonstrates strong potential and value for application in real-world server system maintenance scenarios.

2) Ablation Experiment Results

This paper also further gives the results of the ablation experiment, and the experimental results are shown in Table 3.

Table 3: Ablation Experiment Results

Method	Accuracy	Precision	Recall
Ours	93.1	91.3	89.6
w/o PAS (No Personalized Aggregation)	90.2	88.1	85.7
w/o Meta-Learning	89.5	87.4	84.2
w/o Both PAS & Meta	87.6	85.5	82.8
Global-Only	85.9	83.2	80.5

As shown in the ablation results in Table 3, the complete model proposed in this study achieves the best performance across Accuracy, Precision, and Recall. The accuracy reaches 93.1%, significantly higher than all other ablated versions. This demonstrates that the overall design, which integrates the Personalized Aggregation Strategy (PAS) and Task-Adaptive Meta-Learning, offers a clear performance advantage in server node failure detection. It effectively identifies failure states in complex systems.

Further comparison shows that removing the Personalized Aggregation Strategy (w/o PAS) leads to a noticeable drop in performance. Precision decreases from 91.3% to 88.1%. This indicates that PAS plays a critical role in enhancing the model's adaptability and discriminative accuracy across different nodes. Due to the high heterogeneity among server nodes, a unified model cannot fully capture the local characteristics of each node. PAS addresses this issue and improves overall detection precision.

When the Task-Adaptive Meta-Learning module is removed (w/o Meta-Learning), the Recall drops from 89.6% to 84.2%. This suggests that the model becomes less capable of capturing true failure events. The result highlights the importance of meta-learning in adapting to few-shot and Non-IID data. It enables the model to quickly and effectively learn node-specific failure patterns, thereby enhancing generalization.

Most notably, when both PAS and Meta-Learning are removed (w/o Both) or when a centralized unified training strategy is used (Global-Only), performance declines further. The accuracy drops to 87.6% and 85.9%, respectively. These results confirm that the two core mechanisms are not only effective individually but also highly complementary. Together, they form an efficient failure detection system that supports both global collaboration and local personalization. This fully demonstrates the soundness of the proposed design and its potential for application in distributed and heterogeneous environments.

3) Hyperparameter sensitivity experiments

Furthermore, this paper gives the experimental results of hyperparameter sensitivity. First, the experimental results of the learning rate are given, as shown in Table 4.

Table 4: Hyperparameter sensitivity experiment results (learning rate)

Learning Rate	Accuracy	Precision	Recall
0.004	90.3	88.1	85.0
0.003	91.7	89.6	87.1
0.002	92.4	90.5	88.3
0.001	93.1	91.3	89.6

As shown in the learning rate sensitivity results in Table 4, the proposed model maintains relatively stable performance across different learning rate settings. The best results are achieved when the learning rate is set to 0.001. In this setting, the model reaches 93.1% Accuracy, 91.3% Precision, and 89.6% Recall. This indicates that a lower learning rate helps guide the model to converge more stably on complex Non-IID server node data. It also allows the model to learn feature representations with stronger generalization.

When the learning rate is increased to 0.004, the model performance degrades. Accuracy drops to 90.3%, while

Precision and Recall fall to 88.1% and 85.0%, respectively. This suggests that an excessively high learning rate may cause the model to oscillate or overshoot near local optima. As a result, it struggles to capture fine-grained differences in node failure data, which lowers detection accuracy. In heterogeneous task settings, rapid parameter updates may weaken the model's ability to learn personalized features and reduce its task adaptation capability.

As the learning rate gradually decreases to 0.003 and 0.002, the model performance improves. This demonstrates strong training adaptability and sensitivity to hyperparameter settings. The observed trend also indirectly validates the training mechanism used in this study. The combination of the Personalized Aggregation Strategy (PAS) and Task-Adaptive Meta-Learning can reliably enhance model performance when properly tuned. It effectively activates useful features from local data and improves the generalization ability of the global model.

In summary, a well-chosen learning rate is critical to the proposed method. In multi-task, multi-node, and highly heterogeneous environments, stable convergence is essential for ensuring model generalization and robustness. The excellent performance at a learning rate of 0.001 shows that with fine training control, the model can effectively balance local adaptation and global knowledge sharing. This leads to improved efficiency and accuracy in failure detection.

4) Experiment on the impact of meta-learning update frequency on performance

This paper also gives an experiment on the impact of meta-learning update frequency on performance, and the experimental results are shown in Figure 4.

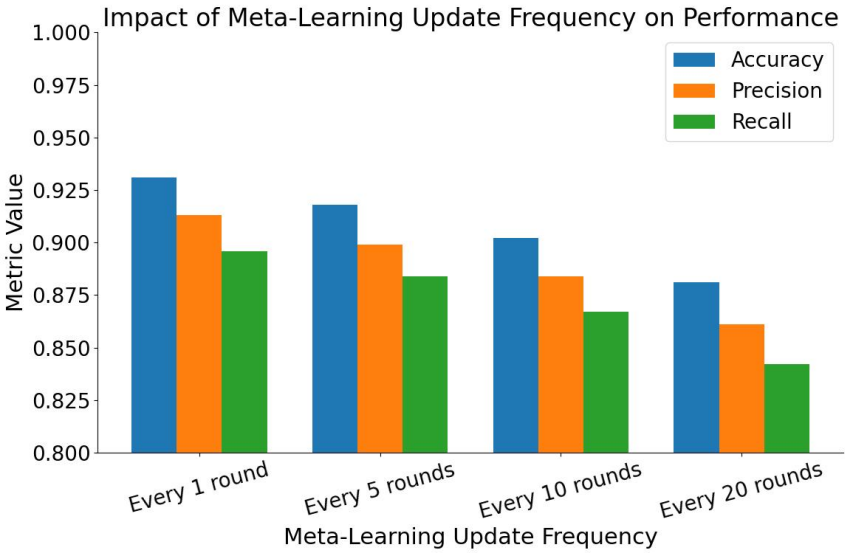

Figure 4. Experiment on the impact of meta-learning update frequency on performance

As shown clearly in the results of Figure 4, the update frequency of meta-learning has a significant impact on model performance in the failure detection task. As the update frequency decreases—that is, as meta-updates occur less frequently—the model shows a consistent decline in Accuracy, Precision, and Recall. This suggests that frequent meta-learning updates help the model maintain strong task adaptability across dynamic and diverse node data. They also enable more effective extraction and transfer of failure-related knowledge.

When the meta-update is performed every single round, the model achieves the best performance. The Accuracy reaches 0.931, Precision is 0.913, and Recall is 0.896. High-frequency updates allow the model to quickly integrate personalized information from different nodes. This keeps the meta-model highly responsive to multi-task environments. As a result, it can more sensitively identify potential failure behaviors across complex and heterogeneous server nodes. This improves the overall completeness and reliability of the detection.

In contrast, when the update frequency is reduced to every 10 or 20 rounds, model performance drops significantly. Recall decreases to 0.842. This trend reflects a delayed response in the meta-model's ability to adapt to new task distributions. Lower update frequencies slow the model's reaction to local changes in node data, which weakens its ability to capture abnormal behavior in time. In real-world system maintenance, such delays may cause late detection of failures and increase operational risks.

Therefore, these experimental results further confirm the critical role of task-adaptive meta-learning in server failure detection. Maintaining a proper and frequent meta-learning update schedule not only improves the model's ability to transfer across Non-IID data but also enhances the timeliness and effectiveness of personalized modeling. This is a key factor in building high-performance distributed intelligent detection systems.

5) Model robustness experiment when nodes are offline

This paper also gives a model robustness experiment when the node is offline.

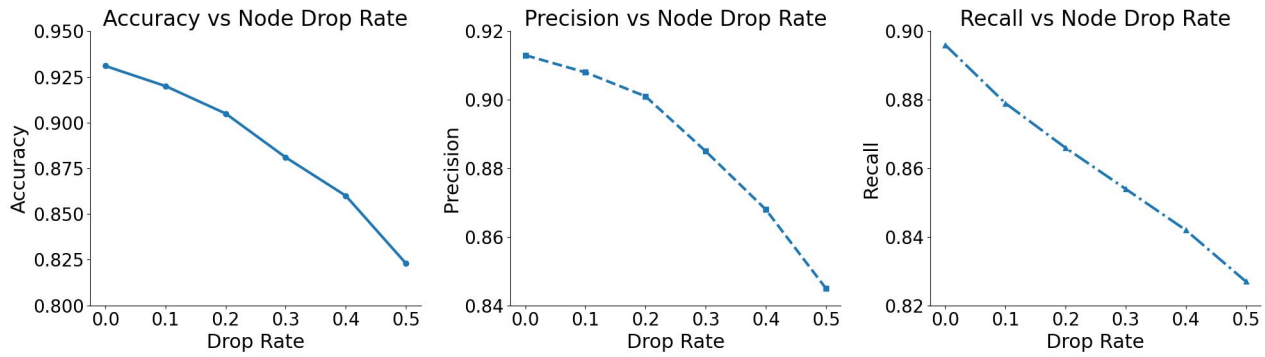

Figure 5. Model robustness experiment when nodes are offline

As shown in the node dropout robustness results in Figure 5, model performance on Accuracy, Precision, and Recall declines to varying degrees as the node dropout rate increases. This indicates that when some server nodes fail to participate in training or updating, the overall performance of the model is significantly affected. It highlights the importance of node activity in the federated learning framework. In heterogeneous node environments, the absence of key nodes may prevent the model from learning complete global features.

In the Accuracy trend, the model drops from 0.931 at 0 percent dropout to 0.823 at 50 percent dropout. This shows that the system's ability to detect failures is weakened when fewer nodes participate. Since federated learning relies on contributions from each node's local data, a high dropout rate leads to information loss and biased aggregation. As a result, the model's generalization ability is reduced, making it harder to detect faults accurately.

The Precision metric declines more gradually, but a noticeable drop occurs when the dropout rate exceeds 30 percent. This suggests that with low node participation, the model becomes more prone to false positives. It tends to misclassify normal states as faults, reducing the reliability of predictions. This behavior is especially problematic for deploying automated failure detection in real systems. Frequent false alarms may waste maintenance resources and disrupt system scheduling.

In contrast, the decline in Recall shows a more linear pattern. This means the model becomes more sensitive to missing real failures. It further confirms that maintaining a sufficient level of node participation is essential for model robustness and comprehensive fault coverage. Overall, the experiment confirms that the proposed method has a certain degree of robustness. However, challenges remain in environments with inactive nodes. Future work may consider integrating active node selection, data augmentation, and model compensation strategies to improve stability under high dropout conditions.

6) Experiment with the influence of parameter synchronization frequency on the detection effect.

Finally, this paper also gives an experiment on the influence of parameter synchronization frequency on the detection effect, and the experimental results are shown in Figure 6.

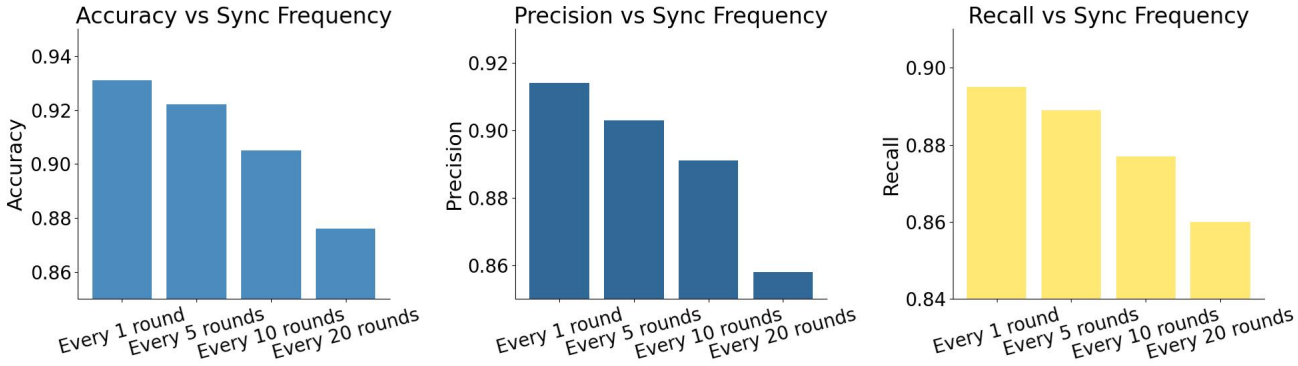

Figure 6. Experiment with the influence of parameter synchronization frequency on the detection effect

As shown in the results of Figure 6, the frequency of parameter synchronization has a significant impact on model performance in the server node failure detection task. As the synchronization interval increases—that is, when the model synchronizes with the global server after more local training rounds—the overall performance in Accuracy, Precision, and Recall shows a declining trend. This indicates that frequent synchronization helps the global model quickly integrate local updates from all nodes, thereby improving detection effectiveness.

Specifically, when the synchronization is performed every round, the model achieves the highest Accuracy at 0.931. When the synchronization frequency decreases to every twenty rounds, the Accuracy drops to 0.876, showing a clear degradation. This performance gap suggests that frequent communication helps mitigate data distribution differences across nodes under Non-IID conditions. It allows the model to more reliably combine learning outcomes from diverse sources.

In terms of Precision, the trend is slightly different but still consistent. As synchronization becomes less frequent, the model's precision gradually declines. This implies that when the model does not receive updated information from other nodes for an extended period, it may overfit local patterns. This leads to an increase in false positives. For failure detection tasks, such behavior reduces the credibility and responsiveness of the alerting system.

Recall also declines more linearly. This shows that the model becomes less effective at capturing real failures as synchronization frequency decreases. Insufficient communication between nodes reduces the model's awareness of global fault patterns in multi-source environments. As a result, more faults go undetected. Overall, the results confirm that the proposed method depends heavily on parameter synchronization. They also highlight that balancing communication cost and detection performance is a key design challenge in building efficient and stable distributed detection systems.

5. Conclusion

This paper proposes an intelligent algorithmic framework that integrates federated learning and meta-learning for the task of server node failure detection. The goal is to address key challenges in distributed environments, such as data heterogeneity, privacy constraints, and personalized task requirements. By introducing a task-adaptive meta-learning mechanism, the model achieves rapid transfer and efficient generalization across diverse failure data sources. This improves detection performance in new nodes or limited-sample scenarios. At the same time, the personalized aggregation strategy enhances the local model's ability to fit node-specific features. It increases the robustness and accuracy of the system under complex node differences.

Through various experimental settings, including comparative analysis, ablation studies, parameter sensitivity evaluations, and robustness tests, the proposed method shows significant advantages in key metrics such as Accuracy, Precision, and Recall. These results validate the theoretical soundness of the design and confirm its practical value in real-world server maintenance scenarios. Especially under Non-IID data conditions, the method demonstrates strong personalized learning and fault response capabilities. It provides effective technical support for intelligent monitoring and autonomous maintenance in distributed systems.

This study has important implications for promoting the integration of federated learning and meta-learning in system-level applications. It expands the research scope of distributed intelligent algorithms and offers a feasible path for collaborative model optimization. In high-availability and real-time response industries such as finance, telecommunications, transportation, and energy, the proposed method can serve as a scalable and deployable intelligent detection solution. It helps ensure the stable operation of large-scale infrastructure. Future research can proceed in several directions. One direction is to explore the model's adaptability in highly dynamic systems, such as environments with frequent node changes or sudden task shifts. Another direction is to extend the method to multi-task collaborative learning or federated transfer learning frameworks, enhancing generalization across systems and scenarios. In addition, emerging techniques such as self-supervised learning and federated neural architecture search can be introduced to optimize model structure and reduce computational and communication costs. These efforts will support the development of more efficient, secure, and intelligent distributed maintenance systems.

References

[1] Zhao, C., Ma, M., Zhong, Z., Zhang, S., Tan, Z., Xiong, X., ... & Zhang, D. (2023, August). Robust multimodal failure detection for microservice systems. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 5639-5649).

[2] Soldani J, Brogi A. Anomaly detection and failure root cause analysis in (micro) service-based cloud applications: A survey[J]. ACM Computing Surveys (CSUR), 2022, 55(3): 1-39.

[3] Zhao C, Ma M, Zhong Z, et al. Robust multimodal failure detection for microservice systems[C]//Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2023: 5639-5649.

[4] Wang Y, Crankshaw D, Yadwadkar N J, et al. SOL: Safe on-node learning in cloud platforms[C]//Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems. 2022: 622-634.

[5] Yahyaoui A, Abdellatif T, Yangui S, et al. READ-IoT: reliable event and anomaly detection framework for the Internet of Things[J]. IEEE Access, 2021, 9: 24168-24186.

[6] Yang L, Moubayed A, Shami A, et al. Multi-perspective content delivery networks security framework using optimized unsupervised anomaly detection[J]. IEEE Transactions on Network and Service Management, 2021, 19(1): 686-705.

[7] Ruba S B, Yellas N E H, Secci S. Anomaly detection for 5g softwarized infrastructures with federated learning[C]//2022 1st International Conference on 6G Networking (6GNet). IEEE, 2022: 1-4.

[8] Wen J, Zhang Z, Lan Y, et al. A survey on federated learning: challenges and applications[J]. International Journal of Machine Learning and Cybernetics, 2023, 14(2): 513-535.

[9] Banabilah S, Aloqaily M, Alsayed E, et al. Federated learning review: Fundamentals, enabling technologies, and future applications[J]. Information processing & management, 2022, 59(6): 103061.

[10] Zhang C, Xie Y, Bai H, et al. A survey on federated learning[J]. Knowledge-Based Systems, 2021, 216: 106775.

[11] Beltrán E T M, Pérez M Q, Sánchez P M S, et al. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges[J]. IEEE Communications Surveys & Tutorials, 2023, 25(4): 2983-3013.

[12] Li Q, He B, Song D. Model-contrastive federated learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 10713-10722.

[13] Nguyen D C, Ding M, Pathirana P N, et al. Federated learning for internet of things: A comprehensive survey[J]. IEEE Communications Surveys & Tutorials, 2021, 23(3): 1622-1658.

[14] Li, Q., Thapa, C., Ong, L., Zheng, Y., Ma, H., Camtepe, S. A., ... & Gao, Y. (2023). Vertical federated learning: taxonomies, threats, and prospects. arXiv preprint arXiv:2302.01550.

[15] Xu, X., Deng, H. H., Chen, T., Kuang, T., Barber, J. C., Kim, D., ... & Yan, P. (2024, January). Federated cross learning for medical image segmentation. In Medical Imaging with Deep Learning (pp. 1441-1452). PMLR.

[16] Sharma S, Guleria K. A comprehensive review on federated learning based models for healthcare applications[J]. Artificial intelligence in medicine, 2023, 146: 102691.

[17] Wang J X. Meta-learning in natural and artificial intelligence[J]. Current Opinion in Behavioral Sciences, 2021, 38: 90-95.

[18] Gharoun, H., Momenifar, F., Chen, F., & Gandomi, A. H. (2024). Meta-learning approaches for few-shot learning: A survey of recent advances. ACM Computing Surveys, 56(12), 1-41.

[19] Huisman M, Van Rijn J N, Plaat A. A survey of deep meta-learning[J]. Artificial Intelligence Review, 2021, 54(6): 4483-4541.

[20] Chi Z, Gu L, Liu H, et al. Metafscil: A meta-learning approach for few-shot class incremental learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022: 14166-14175.

[21] Chen Y, Zhong R, Zha S, et al. Meta-learning via language model in-context tuning[J]. arXiv preprint arXiv:2110.07814, 2021.

[22] Lake B M, Baroni M. Human-like systematic generalization through a meta-learning neural network[J]. Nature, 2023, 623(7985): 115-121.

[23] Collins L, Hassani H, Mokhtari A, et al. Fedavg with fine tuning: Local updates lead to representation learning[J]. Advances in Neural Information Processing Systems, 2022, 35: 10572-10586.

[24] An T, Ma L, Wang W, et al. Consideration of fedprox in privacy protection[J]. Electronics, 2023, 12(20): 4364.

[25] T Dinh C, Tran N, Nguyen J. Personalized federated learning with moreau envelopes[J]. Advances in neural information processing systems, 2020, 33: 21394-21405.

[26] da Silva, F. R., Camacho, R., & Tavares, J. M. R. (2023). Federated learning in medical image analysis: A systematic survey. Electronics, 13(1), 47.