Expert Systems With Applications 298 (2026) 129754

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Federated transfer learning for anomaly detection in HPC systems: First real-world validation on a tier-0 supercomputer

Emmen Farooq *, Michela Milano , Andrea Borghesi

DISI, University of Bologna, Bologna, Italy

ARTICLE INFO

Keywords:
Federated transfer learning
Anomaly detection
High-performance computing
Decentralized machine learning
Transfer learning

ABSTRACT

High-Performance Computing (HPC) systems increasingly require intelligent, scalable anomaly detection to ensure operational reliability. However, conventional centralized approaches often struggle with data privacy constraints, poor generalization across heterogeneous nodes, and limited scalability. This study presents the first real-world application of federated transfer learning (FTL) for anomaly detection in a production-grade Tier-0 supercomputer. By combining federated learning with transfer learning, the proposed framework enables decentralized model training and personalized adaptation to unseen nodes, without accessing raw data.

We validate the approach using two large-scale telemetry datasets collected from 100 nodes of the Marconi100 supercomputer, evaluating its effectiveness across supervised, semi-supervised, and unsupervised learning paradigms. Results show that FTL consistently improves anomaly detection performance on nodes that did not participate in federated training, with F1-score gains reaching up to 0.50. These improvements demonstrate the framework's ability to generalize across non-identically distributed data and maintain detection accuracy under real-world conditions. This work establishes FTL as a scalable, privacy-preserving solution for fault detection in HPC environments. Its practical deployment on production hardware confirms its readiness for real-time monitoring applications in large-scale, heterogeneous computing systems.

1. Introduction

High-Performance Computing (HPC) systems play a pivotal role in advancing scientific discovery and engineering innovation. With increasing architectural complexity and operational scale, modern HPC infrastructures face growing challenges in ensuring reliability, availability, and maintainability. Anomalies-caused by hardware faults, misconfigurations, or software issues-can severely degrade performance or lead to system downtime (Rozanec et al., 2025). Consequently, accurate and scalable anomaly detection has become a cornerstone for the sustainable operation of HPC environments (Sui et al., 2025).

Traditional approaches to anomaly detection in HPC often rely on centralized machine learning techniques, which assume access to unified datasets and homogeneous behavior across nodes (Sencan et al., 2025). However, this assumption rarely holds in practice due to the distributed and heterogeneous nature of supercomputing infrastructures. Moreover, privacy concerns and data volume constraints further hinder centralized training, necessitating decentralized and adaptive strategies that can generalize across diverse node behaviors while preserving data locality (Farooq & Borghesi, 2023).

In production clusters, node telemetry is intrinsically non-IID due to heterogeneous hardware generations (e.g., CPU/GPU), memory/cooling differences, workload diversity (AI training vs. long simulations vs. system services), and operational modes (batch vs. interactive) (Borghesi et al., 2021) These factors yield node-specific distributions that degrade naïve centralized training and even standard *Federated Learning* (FL) when applied to unseen nodes. While FL alleviates privacy concerns and supports scalability by training models locally and aggregating only parameters, it remains limited in transferring knowledge to nodes that do not participate in the federated process (Farooq & Borghesi, 2024; Farooq et al., 2024).In contrast, *Transfer Learning* (TL) has shown promise in enabling model adaptation across domains, particularly when labeled data is scarce, but TL alone neither preserves data privacy nor supports distributed scalability (Yan et al., 2024).

Building on this motivation, we present *Federated Transfer Learning* (FTL) (Guo et al., 2024) as a hybrid framework that simultaneously leverages the scalability and privacy of federated training and the adaptability of transfer learning. Specifically, FTL addresses the limitations of both paradigms by (i) learning general encodings from a subset of representative nodes and (ii) personalizing the transferred model on each

* Corresponding author.
E-mail addresses: emmen.farooq3@unibo.it (E. Farooq), michela.milano@unibo.it (M. Milano), andrea.borghesi3@unibo.it (A. Borghesi).

https://doi.org/10.1016/j.eswa.2025.129754
Received 4 July 2025; Received in revised form 27 August 2025; Accepted 14 September 2025
Available online 18 September 2025

target node with local fine-tuning, thereby bridging from global patterns to local behavior without exposing raw logs (Tan Le et al., 2025).

We provide the first real-world validation of FTL for anomaly detection in a Tier-0 production HPC environment-Marconi100 at CINECA, Italy-evaluated under three learning paradigms: unsupervised, semi-supervised, and supervised.

Novelty. Unlike prior FL-in-HPC studies, which primarily demonstrated the feasibility of federated training across participating nodes, this work explicitly addresses the challenge of generalizing to non-participating nodes under non-IID telemetry conditions in a real Tier-0 supercomputer. By integrating federated learning with transfer learning, we establish Federated Transfer Learning (FTL) as a practical solution for HPC anomaly detection, validated with authentic production data and real faults. Beyond feasibility, our study provides both methodological innovations and empirical analyses that make FTL deployable in production environments.

Contributions. This study advances the state of the art in HPC anomaly detection by introducing and validating FTL in a production-scale environment. In contrast to prior FL-in-HPC research and general FL/TL work, our specific contributions are:

- **First real-world validation of FTL in HPC anomaly detection.** While previous FL studies in HPC focused only on training across participating nodes (Farooq et al., 2024), we are the first to demonstrate that FTL can generalize to non-participating nodes on a Tier-0 production supercomputer (Marconi100) with authentic telemetry and real faults.
- **Cross-paradigm evaluation across supervision levels.** Prior FL-based HPC studies typically restricted evaluation to a single paradigm (mostly unsupervised) (Molan et al., 2023). We systematically assess FTL in *unsupervised, semi-supervised, and supervised* settings, establishing its robustness across different levels of labeled data availability.
- **Performance-aware vs. diversity-aware node selection.** Unlike earlier works that treated node selection as incidental (Farooq & Borghesi, 2024), we explicitly compare *Top-N* (performance-based) versus *Random-N* (diversity-driven) strategies, and support our findings with statistical significance testing.
- **Resource-aware personalization.** Where prior approaches assume full-model adaptation (Farooq et al., 2024), we propose *decoder-only fine-tuning* to preserve global encodings while reducing time, memory, and energy costs. This provides a practical balance between performance and efficiency in HPC deployment.
- **Empirical justification of non-IID heterogeneity.** Previous FL-in-HPC papers note heterogeneity but do not quantify its impact (Farooq & Borghesi, 2023). We contribute a detailed inter-node similarity analysis (PCA projections and cosine similarity of encoder weights), empirically validating why certain nodes transfer better under non-IID conditions.

1.1. Structure of the paper

The remainder of this paper is organized as follows. Section 2 reviews related work on anomaly detection in HPC systems and recent advances in federated and transfer learning, thereby positioning our contribution in the existing literature. Section 3 introduces the methodology, outlining the design of our Federated Transfer Learning (FTL) framework and explaining the role of local training, federated aggregation, and knowledge transfer. Section 4 presents the experimental evaluation, describing the datasets, implementation details, and evaluation metrics, followed by a thorough analysis of the empirical results across different learning paradigms and node selection strategies. Section 5 concludes the paper by summarizing the main findings, highlighting the practical implications for real-world HPC deployments, and discussing directions for future research.

2. Related work

This section reviews prior research on anomaly detection in HPC systems, federated learning, and transfer learning. The goal is to position our work within this literature and highlight the gap that our study addresses.

Anomaly detection is a key component in ensuring the reliability and stability of High-Performance Computing (HPC) systems. Several foundational studies have focused on node-level anomaly detection, leveraging local telemetry data to model system behavior and identify deviations (Borghesi et al., 2019, 2021). These contributions have provided valuable insights into system dynamics and inspired further exploration of intelligent monitoring techniques in HPC environments.

More recently, deep learning models-particularly autoencoders-have shown strong potential in learning the typical behavior of individual nodes and flagging unusual activity based on reconstruction errors (Borghesi et al., 2019, 2021; Okur et al., 2025). These approaches offer powerful tools for capturing nonlinear relationships in high-dimensional telemetry streams. Ongoing work in this area increasingly incorporates techniques such as domain adaptation (Sui et al., 2025) and self-supervised learning (Xie et al., 2024) to enhance generalizability and adaptivity across diverse operational contexts, while continuing to respect data locality and system heterogeneity.

Federated Learning (FL) has emerged as a privacy-preserving approach that enables decentralized model training while keeping raw data localized (Farooq & Borghesi, 2023, 2024; Farooq et al., 2024). Some of the earliest efforts to explore FL in the context of High-Performance Computing (HPC) have focused on applying it to anomaly detection, using deep learning models such as LSTMs and autoencoders (Farooq & Borghesi, 2023, 2024). These studies demonstrated the feasibility of federated training across compute nodes and contributed to advancing privacy-aware, distributed anomaly detection in HPC environments.

Building on this line of research, FL continues to gain traction as a viable solution for enabling scalable monitoring in systems characterized by data heterogeneity and constrained communication. These foundational implementations have helped set the stage for broader applications of FL in real-world supercomputing infrastructures. *We humbly note that all of the cited contributions represent different phases of our ongoing research agenda in this area.*

Transfer Learning (TL) offers a mechanism to generalize knowledge across tasks or domains, typically by fine-tuning pretrained models on new target domains. In the context of anomaly detection, TL has been used to transfer representations across devices or data domains (Qian et al., 2025). However, TL alone cannot address privacy or scalability constraints in HPC systems. Federated Transfer Learning (FTL) combines the strengths of FL and TL by enabling trained federated models to be transferred and fine-tuned on target clients that were not part of the original training cohort. FTL has been applied in domains such as healthcare (Bechar et al., 2025), structural monitoring (Okur et al., 2025), and image classification (RahimiZadeh et al., 2025), but its application to HPC systems remained unexplored until recently.

RahimiZadeh et al. (2025) proposed a fine-tuned aggregation strategy (FTA-FTL) for image classification, while Qian et al. (2025) provided a comprehensive review of FTL for fault diagnosis in industrial machinery. These works highlight the potential of FTL in settings with heterogeneous clients and data distributions, motivating its extension to the domain of HPC anomaly detection.

Several recent works have advanced the state of Federated Transfer Learning (FTL) and its applications in anomaly detection across diverse domains:

- **FedDyMem** introduces a memory-efficient FL approach for unsupervised anomaly detection, leveraging dynamic memory banks to improve client-side generalization (Chen et al., 2025).

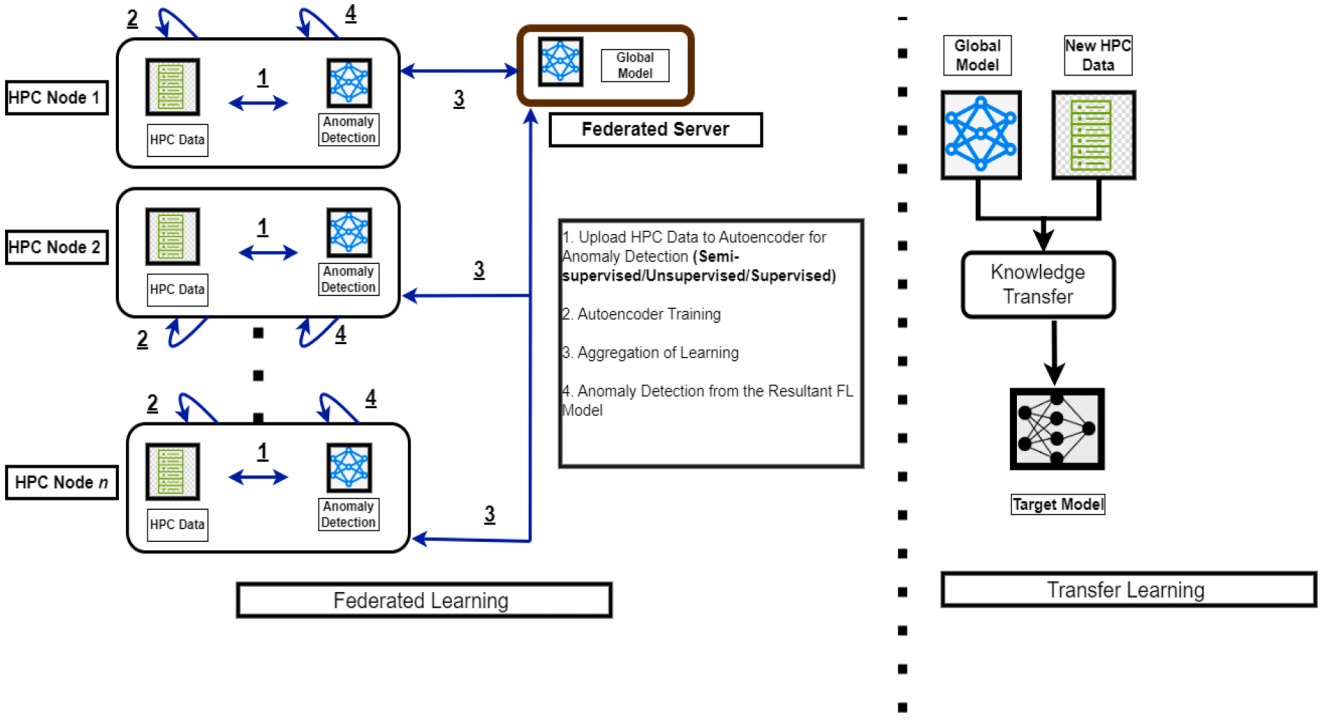


Fig. 1. Federated Transfer Learning (FTL) pipeline for anomaly detection in HPC. Local autoencoders are trained on private telemetry data; Top-N nodes contribute to global aggregation via FedAvg; the resulting model is transferred to remaining nodes and fine-tuned locally. This process ensures privacy, scalability, and generalization across heterogeneous HPC systems.

- **FedKO** proposes a federated Koopman-reservoir framework to model large-scale spatiotemporal time series for privacy-preserving anomaly detection (Tan Le et al., 2025).
- **Sui** et al. enhances model transferability across systems via domain adaptation and unified log feature representation (Sui et al., 2025).
- **Laridi** et al. present a federated thresholding method based on summary statistics to enhance accuracy in autoencoder-based detection under non-IID conditions (Laridi et al., 2024).
- **Marfo** et al. introduce adaptive communication strategies to reduce FL overhead while maintaining performance in large-scale anomaly detection tasks (Marfo et al., 2025).

While these contributions demonstrate the versatility and growing relevance of FTL for distributed anomaly detection, its application to High-Performance Computing (HPC) systems-particularly for anomaly detection-remains an open and largely unexplored avenue. To the best of our knowledge, FTL has not yet been implemented or evaluated in real-world HPC environments, which highlights the novelty and motivation behind the present work.

To the best of our understanding, this study builds upon our prior efforts in applying Federated Learning (FL) to anomaly detection in HPC systems (Farooq & Borghesi, 2023, 2024; Farooq et al., 2024), and offers one of the first real-world demonstrations of Federated Transfer Learning (FTL) in a Tier-0 production HPC environment. While our earlier work focused on training across participating nodes, the present study extends this direction by exploring how knowledge from federated models can be generalized to non-participating nodes through personalized fine-tuning.

The proposed framework is evaluated under multiple learning paradigms-supervised, semi-supervised, and unsupervised-and across different node selection strategies (Top-N and Random-N). We hope this work contributes meaningfully to the growing research community focused on scalable, privacy-preserving anomaly detection in large-scale, heterogeneous computing environments.

3. Methodology

This section details the design of our Federated Transfer Learning framework. We describe local training, federated aggregation, transfer to non-participating nodes, and personalization steps, outlining how each component supports anomaly detection in heterogeneous HPC systems.

This section describes the experimental pipeline used to evaluate Federated Transfer Learning (FTL) for anomaly detection across 100 HPC nodes under unsupervised, semi-supervised, and supervised learning paradigms. The methodology follows a structured process involving local model training, federated aggregation via FedAvg, knowledge transfer to unseen nodes, and localized fine-tuning for personalization. The overall workflow is illustrated in Fig. 1.

3.1. Local model training and node ranking

We employ dense autoencoders as the foundational architecture for anomaly detection on all nodes (Okur et al., 2025). An autoencoder is a type of neural network designed to learn compressed representations of input data by minimizing the reconstruction error between input and output (Borghesi et al., 2019). It consists of two components: an encoder that maps input data to a lower-dimensional latent space, and a decoder that reconstructs the original input from this latent representation (Farooq & Borghesi, 2024). The reconstruction error, typically measured using mean squared error (MSE), quantifies how accurately the model can reproduce the input data (Borghesi et al., 2021).

Autoencoders are particularly suitable for anomaly detection because they are trained primarily on normal (non-anomalous) data. Consequently, they learn to reconstruct expected system behavior (Borghesi et al., 2019). During inference, anomalous inputs that deviate from this learned distribution tend to exhibit high reconstruction errors and are thus flagged as outliers.

This strategy aligns well with the characteristics of HPC environments (Farooq & Borghesi, 2023; Farooq et al., 2024), where:

- Faults are infrequent compared to normal operation, resulting in highly imbalanced datasets.
- Data distributions across nodes are heterogeneous and non-iid.
- Annotating anomalies is resource-intensive and often infeasible in production environments.

As shown on the left side of Fig. 1, each node independently trained a local anomaly detection model using its private data. The training configuration varied according to the target learning paradigm:

- **Unsupervised:** The autoencoder was trained solely on normal data, and anomalies were identified via high reconstruction error.
- **Semi-supervised:** A subset of labeled anomalies was included during training, allowing partial supervision to guide reconstruction.
- **Supervised:** A classification head was appended to the encoder, and the entire model was trained using labeled normal and anomalous samples.

Following training, each model was evaluated using the F1-score on a held-out test set, and all 100 nodes were ranked according to their individual anomaly detection performance. These rankings were subsequently used for node selection in the federated learning stage.

The F1-based ranking does more than just measure performance. As suggested by our hypothesis and later similarity analyses Fig. 5, nodes with higher ranks tend to lie closer to the centroid of the inter-node distribution, making them strong knowledge sources. This insight guides the Top-N selection in Section 3.2, where we seed the global model with these representative nodes to ensure more generalizable encodings.

3.2. Federated learning with FedAvg

From the ranked list, subsets of top-performing nodes were selected with $N \in \{3, 5, 10, 20, 25, 50\}$. These top-N nodes participated in federated learning (FL) using the Federated Averaging (FedAvg) algorithm, illustrated in the center of Fig. 1.

FedAvg (Farooq et al., 2024) performs decentralized training through the following steps:

1. A central server initializes and distributes a global model to the selected top-N nodes.
2. Each client node trains the model locally on its private data for a fixed number of epochs.
3. The locally updated model weights are sent back to the server.
4. The server computes a weighted average of the received updates based on the size of each local dataset:

$$w_{t+1} = \sum_{k=1}^{N} \frac{n_k}{n} w_t^{(k)}, \quad \text{where} \quad n = \sum_{k=1}^{N} n_k$$

This process continues for multiple communication rounds until convergence. The final aggregated model serves as a generalized anomaly detection model informed by the most reliable nodes and is saved for downstream transfer.

We hypothesize that high-performing nodes-those selected in the Top-N strategy-tend to capture more generalizable behavioral patterns. As such, their learned representations are expected to lie closer to the centroid of the feature distribution across all nodes.

3.3. Federated transfer learning (FTL)

As depicted on the right side of Fig. 1, the global model obtained from federated learning is transferred to the remaining $100 - N$ nodes that were not involved in FL. Each of these nodes fine-tunes the transferred model using its local data to adapt it to its unique distribution and behavior patterns (Qian et al., 2025).

This step constitutes the Federated Transfer Learning (FTL) process. It enables reuse and personalization of the federated model without retraining from scratch, making it suitable for dynamic or resource-constrained settings (Bechar et al., 2025). The anomaly detection performance of each node is measured both before and after fine-tuning using Precision, Recall, True Negative Rate (TNR), and F1-score. The performance improvement, denoted by ΔF1, quantifies the effectiveness of FTL.

3.4. Comparison with random node selection

-FTL pipeline was repeated using randomly selected N nodes instead of top-ranked nodes. A global model trained from these randomly chosen nodes was then transferred to the same $100 - N$ target nodes and fine-tuned identically.

This comparison highlights the potential benefits of performance-aware node selection in improving the generalizability and adaptability of federated models across unseen clients.

3.5. Fine-tuning for local personalization

After transfer, local fine-tuning is necessary (not optional) to adapt the global model to node-specific distributions that arise from hardware, workload, and operational heterogeneity. We default to decoder-only adaptation to (i) preserve the global encodings learned in the encoder via FL and (ii) minimize time, memory, and energy overhead (see Sections 4.3.5 and 4.6), while still capturing node-level reconstruction mismatches.

Decoder-Only Fine-Tuning Justification. We emphasize that decoder-only adaptation is not merely an efficiency optimization, but a deliberate methodological choice. Freezing the encoder preserves the global, federated encodings learned during training-an essential factor for robustness under non-IID conditions-while the decoder is adapted to capture node-specific reconstruction mismatches. This design balances generalization and personalization without disrupting the shared representation space.

To substantiate this choice, we explicitly point the reader to Section 4.3.5 and Fig. 3, which compare decoder-only with full encoder + decoder fine-tuning. Although full fine-tuning yields slightly higher F1-scores on some nodes, Section 4.6 and Table 7 demonstrate that it doubles training time and significantly increases memory and energy usage. These trade-offs are non-trivial in HPC production environments, where efficiency and scalability are paramount. Therefore, decoder-only fine-tuning is adopted as the practical default in our framework, while adaptive strategies (e.g., selective unfreezing based on inter-node divergence) are left as a promising direction for future work.

4. Experimental evaluation

This section presents the experimental setup, datasets, and implementation details used to validate our framework. The purpose is to establish the empirical foundation for evaluating the effectiveness of FTL under real-world HPC conditions. To assess the effectiveness and generalizability of the proposed Federated Transfer Learning (FTL) framework for anomaly detection in real-world HPC environments, we conduct a thorough empirical evaluation across two production-scale datasets under supervised, semi-supervised, and unsupervised settings.

4.1. Experimental setup

All experiments were conducted on a high-performance server equipped with 42 Intel(R) Xeon(R) Gold 6240R CPUs (96 cores total), 790 GB of RAM, and running Ubuntu 22.0. The implementation stack comprised Python 3.8 with TensorFlow 2.x for model development and orchestration, and Scikit-learn for data normalization and preprocessing. Data handling and automation were facilitated through `pandas`, `numpy`, and `bash`, with batch scripts used to simulate node-level parallelism across the experimental pipeline.

It is worth noting that this server-based setup represents a conservative estimate of performance. In real-world HPC environments-such as the Tier-0 supercomputing infrastructure leveraged in our study-the training time and resource efficiency are markedly improved due to the availability of highly optimized, large-scale computational resources.

4.2. Data sources and characteristics

This study employs real telemetry datasets from Marconi100,[1] a Tier-0 supercomputing platform managed by CINECA, Italy. Marconi100 comprises over 900 compute nodes with homogeneous architecture, each equipped with IBM POWER9 CPUs and NVIDIA Volta V100 GPUs. Its production-scale deployment and high-frequency monitoring infrastructure make it ideal for evaluating anomaly detection techniques under real-world operational loads.

To facilitate model validation across varying scenarios, we use two separate datasets, referred to as D1 and D2, both collected from Marconi100 under distinct monitoring configurations.

4.2.1. Dataset D1: Initial monitoring configuration

The first dataset (D1) was acquired from Marconi100 using the earlier version of its telemetry infrastructure. It includes 462 hardware-level features per node, such as temperature, clock frequency, GPU utilization, and power metrics. Labels were obtained from Nagios[2] monitoring logs at 15-min intervals, distinguishing between healthy and anomalous states. These logs reflect administrator-reported and tool-detected events, ensuring operational relevance. For temporal modeling, statistical features (mean and standard deviation) were aggregated over each sampling window.

D1 exhibits an anomaly rate of approximately 3 %, providing a relatively dense fault profile for initial model development and evaluation.

4.2.2. Dataset D2: Upgraded monitoring with refined annotations

The second dataset (D2) was generated after an infrastructure upgrade that introduced additional physical sensors and software probes. As a result, D2 captures 573 features per node, improving coverage of system dynamics. This dataset also benefits from a refined annotation protocol-implemented in collaboration with system administrators-that excludes benign transients (e.g., temporary reconfigurations) previously misclassified as faults. Consequently, D2 presents a sparser anomaly rate of 1.7 %, offering a more challenging and realistic testbed for detection algorithms.

Despite originating from the same system, D1 and D2 are non-overlapping in time and configuration. Their differences in feature granularity, fault prevalence, and annotation methodology make them complementary for studying generalization across detection tasks. Dataset D2 is publicly available to the research community via Zenodo[3], enabling reproducibility and further exploration by both academic and industrial practitioners.

4.2.3. Rationale for dataset selection

We intentionally restrict our analysis to the Marconi100 platform to maintain consistency in hardware, logging frequency, and system semantics. Unlike synthetic or benchmark datasets, these telemetry logs reflect authentic, organically occurring system behaviors in a high-stakes production environment. This setting ensures ecological validity and minimizes bias introduced by artificial fault injection. While broader generalization to other HPC environments is an important future direction, the use of real-world, curated telemetry data from Marconi100 allows us to isolate and quantify the effects of model design choices with minimal confounding variability.

4.2.4. Node selection and experimental scope

We selected 100 nodes from each dataset (D1 and D2). These same nodes were used for: (i) baseline AE training, (ii) federated learning (Top-N or Random-N selection), and (iii) transfer learning. This consistent setup allows meaningful comparative analysis across learning paradigms and datasets.

4.3. Implementation

Incomplete and null values were dropped from all HPC data, followed by an 80 %-20 % train-test split and normalization to a [0, 1] range using Scikit-learn[4] before any implementation begins.

The implementation of our Federated Transfer Learning (FTL) framework follows a structured pipeline: (i) baseline local training of autoencoders on 100 HPC nodes, (ii) federated training on a selected subset of nodes using Federated Averaging (FedAvg), and (iii) knowledge transfer and personalization on the remaining nodes via fine-tuning. The full architectural pipeline is depicted in Fig. 1. To promote reproducibility and enable further research, the complete implementation is available at:[5].

4.3.1. Node selection strategies: Top-N vs. Random-N

After training local autoencoders on all 100 nodes, each node's anomaly detection performance is evaluated using the F1-score on a held-out test partition. Based on this, we define two node selection strategies for the federated training phase:

- **Top-N Selection**: Nodes are ranked by descending F1-score. The top $N \in \{3, 5, 10, 20, 25, 50\}$ nodes are selected for participation in federated learning. This selection targets nodes with the most reliable anomaly detection models to serve as high-quality contributors to the global model.
- **Random-N Selection**: To evaluate the effect of performance-aware selection, we repeat the same experiment by randomly selecting N nodes from the pool of 100 for federated learning.

For each value of N, the remaining $100 - N$ non-participating nodes are kept identical across both Top-N and Random-N experiments. This ensures that any observed performance differences stem solely from the selection strategy used during the federated learning phase and not from variations in the nodes receiving the transferred model.

4.3.2. Autoencoder architecture

Each HPC node trains a dense autoencoder with the following architecture:

- **Input Layer:** Size equal to the number of features (462 for D1, 573 for D2), activation: ReLU
- **Encoder:** Dense(100 → 80) → ReLU
  Dense(80 → 60) → ReLU
  Dense(60 → 40) → ReLU
  Dense(40 → 20) → ReLU
  Dense(20 → 40) → ReLU (latent)
- **Decoder:** Dense(40 → 60) → ReLU
  Dense(60 → 80) → ReLU
  Dense(80 → 100) → ReLU
  Dense(100 → #features) → Sigmoid

This architecture provides sufficient depth to capture high-dimensional telemetry patterns and generalizes well across diverse node behaviors.

[1] https://www.hpc.cineca.it/hardware/marconi100
[2] https://www.nagios.org
[3] https://zenodo.org/records/7589942
[4] https://scikit-learn.org
[5] https://github.com/EmmenFarooq/ftl-hpc-anomaly-detection

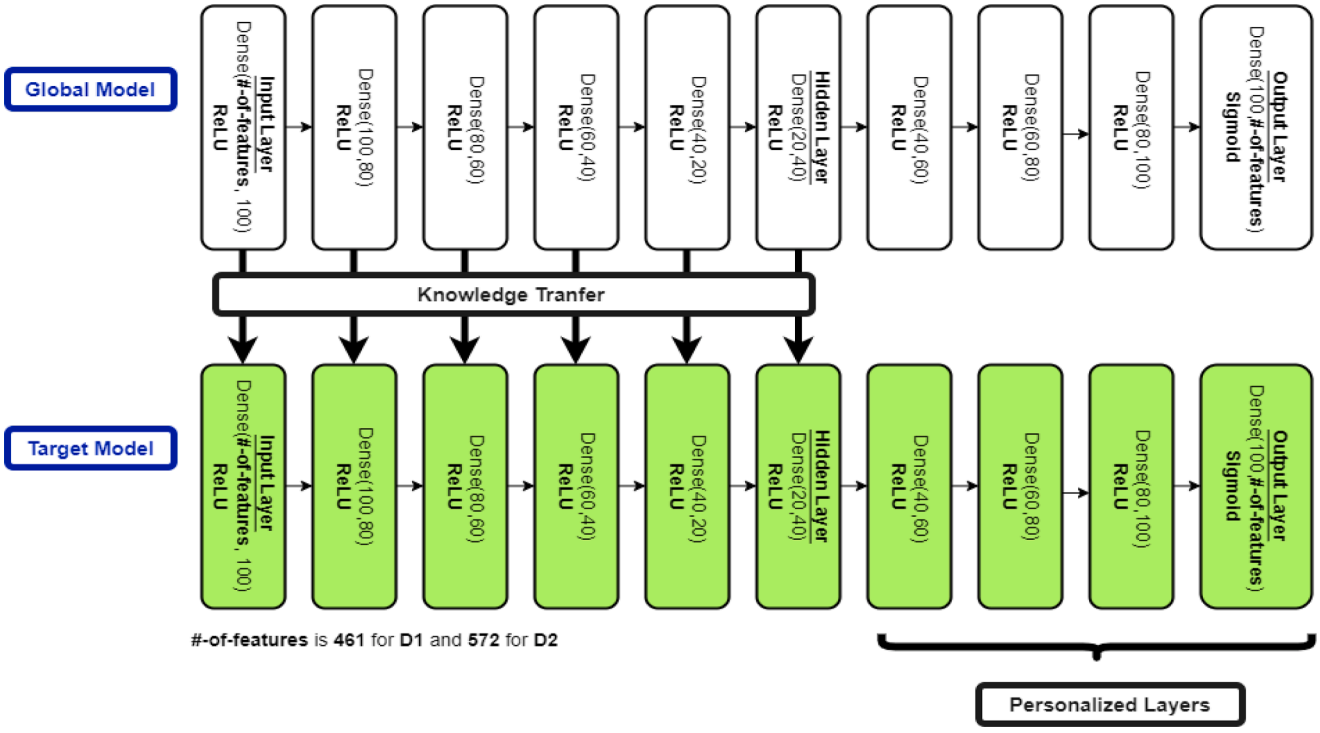


Fig. 2. Knowledge transfer process. The federated global model is transferred to non-participating nodes by freezing encoder layers and fine-tuning decoder layers with local data, enabling node-specific adaptation and generalization without raw data exchange.

4.3.3. *Federated learning with tensorflow federated*

Federated learning is implemented using TensorFlow Federated (TFF) (TensorFlow, 2025a). The FedAvg algorithm is executed across the selected N nodes (from either Top-N or Random-N strategies). The procedure includes:

1. Initialize a global model and distribute it to selected clients.
2. Each client performs local training on private data using RMSprop optimizer for one epoch per round.
3. Clients return updated weights to the central server.
4. The server aggregates updates via a weighted average based on local dataset sizes.

This cycle is repeated for 1000 communication rounds to produce a global model representing generalized anomaly patterns across participating nodes.

4.3.4. *Federated transfer learning and personalization*

After federated training, the global model is transferred to the remaining $100 - N$ nodes. As depicted in Fig. 2:

- **Encoder layers** are frozen to retain generalized features.
- **Decoder layers** are reinitialized and fine-tuned locally to adapt to each node's specific data distribution.

This approach enables personalized anomaly detection without requiring complete retraining or labeled data on the target nodes.

4.3.5. *Justification of fine-tuning strategy*

In the proposed FTL pipeline, we freeze the encoder and fine-tune only the decoder during the personalization step on non-participating nodes. This decision is motivated by practical and computational considerations. By freezing the encoder, we preserve the generalized representations learned during federated training, while limiting the number of trainable parameters-an approach that significantly reduces training time and resource overhead.

Table 1
Hyperparameters used in autoencoder training.

Hyperparameter	Value	Applicable To
Optimizer	RMSprop	All paradigms
Learning Rate	1×10^{-6}	All paradigms
Loss Function	Mean Squared Error (MSE)	All paradigms
Batch Size	10	All paradigms
Epochs	10	All paradigms
Reconstruction Threshold	0.5	Supervised, Unsupervised
Dynamic Threshold	98th percentile of training error	Semi-supervised only

However, this design choice is not exhaustively justified in the current study. To explore whether full model fine-tuning could yield superior results, we perform a comparative analysis of F1-score improvements across 20 randomly selected nodes using both strategies: (i) decoder-only fine-tuning and (ii) full encoder + decoder fine-tuning. As shown in Fig. 3, full fine-tuning results in marginally higher F1-scores across most nodes, suggesting potential gains in accuracy at the cost of increased computation.

Future work will investigate adaptive fine-tuning strategies where selective unfreezing of encoder layers is conditioned on inter-node distribution divergence, to balance performance gains with computational efficiency.

4.3.6. *Training configuration and hyperparameters*

All training and fine-tuning steps follow a consistent setup. The hyperparameters used are summarized in Table 1.

4.3.7. *Anomaly detection and evaluation*

Anomalies are identified based on reconstruction error or classification output:

- **Unsupervised:** Anomaly if reconstruction error > 0.5
- **Semi-supervised:** Threshold set to 98th percentile of training error on normal data

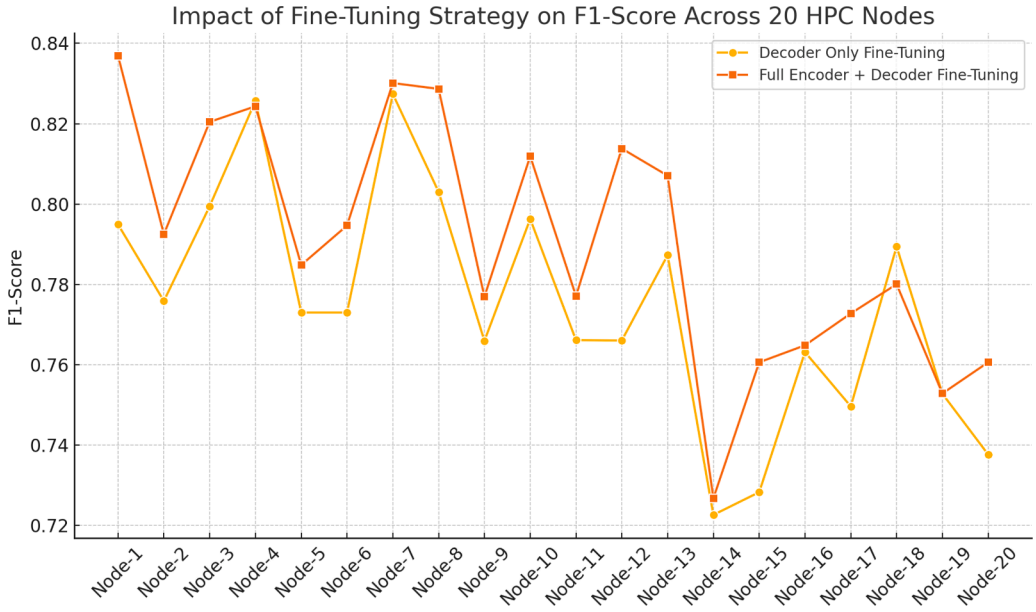


Fig. 3. Comparison of F1-scores across 20 HPC nodes using decoder-only vs full fine-tuning strategies. While full fine-tuning slightly outperforms decoder-only tuning in most cases, the latter remains preferable in resource-constrained settings.

- **Supervised:** Anomaly if classifier output > 0.5

4.4. Evaluation metrics

To evaluate the performance of the proposed anomaly detection framework, we adopt standard classification metrics that are widely used in the literature (Aksar et al., 2021). These metrics provide a comprehensive assessment of detection accuracy, robustness, and reliability across all learning paradigms.

Precision quantifies the proportion of correctly identified anomalies among all instances predicted as anomalies. It is defined as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{1}$$

Recall, also known as the true positive rate, measures the proportion of actual anomalies that are correctly identified:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{2}$$

F1-Score is the harmonic mean of precision and recall, offering a balanced measure of model performance:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

True Negative Rate (TNR) evaluates the proportion of correctly identified normal instances:

$$\text{True Negative Rate} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}} \tag{4}$$

Delta F1 (ΔF1) quantifies the improvement in F1-score after applying Federated Transfer Learning (FTL) to non-participating nodes. It captures the performance gain achieved through knowledge transfer and is computed as:

$$\Delta\text{F1} = \text{F1}_{\text{after FTL}} - \text{F1}_{\text{before FTL}} \tag{5}$$

This metric is particularly important for evaluating the effectiveness of FTL in enhancing anomaly detection performance on nodes that were not directly involved in federated training.

4.5. Results

This section reports the results of our evaluation, comparing supervised, semi-supervised, and unsupervised settings, as well as Top-N vs. Random-N node selection. The goal is to analyze how FTL impacts anomaly detection accuracy and robustness.

To guide the reader, we first report detailed results for each learning paradigm, highlighting FTL's impact on F1-score and True Negative Rate (TNR) under Top-N and Random-N node selection strategies. We then provide a comparative summary across paradigms to identify overarching trends. Finally, we present a statistical confidence analysis, including effect size (Cohen's d), to quantify the reliability and practical significance of observed improvements. This structure allows the reader to follow both granular and holistic insights into FTL's performance across diverse experimental settings.

4.5.1. Semi-supervised anomaly detection

As shown in Table 2, FTL achieves substantial improvements across all configurations in the semi-supervised setting. Using Top-N node selection, ΔF1 reaches up to +0.50 on Dataset D1 and +0.49 on Dataset D2, with F1-scores rising from 0.17-0.41 to 0.70-0.92 across various N values. TNR also improves significantly, confirming FTL's effectiveness in reducing false positives.

For example, on D1 with $N = 10$ (Top-N), F1 improves from 0.35 to 0.84; on D2 with $N = 3$, F1 improves from 0.54 to 0.97. Notably, Random-N selection achieves ΔF1 as high as +0.69 (D1, $N = 50$), suggesting that node diversity can sometimes enhance model generalization.

These findings align with our hypothesis (Section 3) that high-performing nodes contribute strong generalizable features during federated aggregation, which transfer effectively during fine-tuning. In operational terms, such improvements can drastically reduce anomaly miss rates and false alarms, leading to more efficient and reliable HPC operations.

4.5.2. Unsupervised anomaly detection

Table 3 illustrates performance under the unsupervised setting, where models are trained solely on normal data. Despite the absence of labeled anomalies, FTL leads to consistent improvements.

Table 2
Performance of semi-supervised anomaly detection with federated transfer learning (FTL) on datasets D1 and D2 using 100 nodes. **Columns 2-3** show F1-scores before and after FL on selected Top-N or Random-N nodes. **Columns 5-12** report performance before and after applying FTL to the remaining $(100 - N)$ nodes. **Column 13** shows resulting F1 gain (ΔF1). Asterisk (*) indicates statistical significance ($p < .05$).

N	FL on N Nodes		100-N	Before FTL on 100-N Nodes				After FTL on 100-N Nodes				F1 Gain
	F1 (AE)	F1 (FL)		Prec.	Rec.	TNR	F1	Prec.	Rec.	TNR	F1	ΔF1
Dataset D1 - Top-N Strategy												
3	0.95	0.99	97	0.41	1.00	0.44	0.48	**0.88**	**1.00**	**0.90**	**0.92**	0.44*
5	0.92	0.99	95	0.37	1.00	0.35	0.42	**0.86**	**1.00**	**0.86**	**0.90**	0.48*
10	0.89	0.94	90	0.31	1.00	0.29	0.35	**0.80**	**1.00**	**0.79**	**0.84**	0.49*
20	0.85	0.94	80	0.26	1.00	0.25	0.29	**0.74**	**1.00**	**0.76**	**0.77**	0.48*
25	0.78	0.89	75	0.22	1.00	0.21	0.25	**0.70**	**1.00**	**0.71**	**0.75**	0.50*
50	0.72	0.87	50	0.17	1.00	0.15	0.21	**0.66**	**1.00**	**0.65**	**0.70**	0.49*
Dataset D1 - Random-N Strategy												
3	0.10	0.55	97	0.41	1.00	0.44	0.48	**0.57**	**1.00**	**0.60**	**0.61**	0.13*
5	0.25	0.75	95	0.37	1.00	0.35	0.42	**0.67**	**1.00**	**0.64**	**0.73**	0.31*
10	0.34	0.87	90	0.31	1.00	0.29	0.35	**0.71**	**1.00**	**0.68**	**0.76**	0.41*
20	0.92	0.99	80	0.26	1.00	0.25	0.29	**0.89**	**1.00**	**0.86**	**0.92**	0.63*
25	0.32	0.82	75	0.22	1.00	0.21	0.25	**0.61**	**1.00**	**0.61**	**0.63**	0.38*
50	0.64	0.98	50	0.17	1.00	0.15	0.21	**0.90**	**1.00**	**0.91**	**0.90**	0.69*
Dataset D2 - Top-N Strategy												
3	0.98	0.99	97	0.47	1.00	0.49	0.54	**0.93**	**1.00**	**0.96**	**0.97**	0.43*
5	0.96	0.99	95	0.43	1.00	0.40	0.48	**0.91**	**1.00**	**0.92**	**0.95**	0.47*
10	0.93	0.99	90	0.37	1.00	0.34	0.41	**0.85**	**1.00**	**0.85**	**0.89**	0.48*
20	0.90	0.97	80	0.32	1.00	0.30	0.35	**0.79**	**1.00**	**0.82**	**0.83**	0.48*
25	0.84	0.94	75	0.28	1.00	0.26	0.31	**0.75**	**1.00**	**0.77**	**0.80**	0.49*
50	0.78	0.91	50	0.23	1.00	0.20	0.27	**0.71**	**1.00**	**0.71**	**0.75**	0.48*
Dataset D2 - Random-N Strategy												
3	0.39	0.89	97	0.27	1.00	0.26	0.30	**0.67**	**1.00**	**0.67**	**0.70**	0.40*
5	0.91	0.99	95	0.21	1.00	0.21	0.24	**0.94**	**1.00**	**0.97**	**0.96**	0.72*
10	0.22	0.71	90	0.26	1.00	0.29	0.36	**0.62**	**1.00**	**0.59**	**0.67**	0.31*
20	0.59	0.91	80	0.14	1.00	0.11	0.17	**0.73**	**1.00**	**0.70**	**0.78**	0.61*
25	0.15	0.61	75	0.46	1.00	0.49	0.53	**0.51**	**1.00**	**0.53**	**0.55**	0.02
50	0.28	0.79	50	0.32	1.00	0.38	0.42	**0.64**	**1.00**	**0.61**	**0.63**	0.21

For D1, Top-N ($N = 50$) improves F1 from 0.13 to 0.60 (ΔF1 = +0.47), while Random-N ($N = 10$) improves F1 from 0.34 to 0.85 (ΔF1 = +0.51). D2 results echo this trend: Top-N ($N = 3$) improves F1 from 0.47 to 0.91, and Random-N achieves ΔF1 = +0.46 for the same N. TNR gains across Random-N (e.g., +0.52 at $N = 5$ in D2) show FTL's robustness in reducing false positives without relying on labels.

These outcomes suggest that FTL can capture latent representations of normal behavior, which transfer effectively even across heterogeneous node profiles.

4.5.3. Supervised anomaly detection

In the supervised setting (Table 4), we use only one-fourth of the labeled data to simulate partial supervision. Even under this constraint, FTL significantly improves model performance.

On D1, Top-N ($N = 3$) improves F1 from 0.65 to 0.96 (ΔF1 = +0.31), while Random-N achieves a comparable gain of +0.32. On D2, Top-N ($N = 50$) improves F1 from 0.22 to 0.63, while Random-N ($N = 20$) achieves a remarkable ΔF1 = +0.52. These improvements demonstrate FTL's ability to generate transferable representations from limited labeled sources.

Such gains are particularly valuable in production settings where labeling is expensive and limited. The results reaffirm the feasibility of using a small, labeled subset of nodes to enhance anomaly detection across the broader system.

Across both datasets (D1 and D2), Federated Transfer Learning (FTL) consistently improved F1-scores across all tested values of N. Notably, the semi-supervised setting exhibited the most substantial gains, particularly under small N conditions. For instance, Top-N selection with $N = 3$ led to F1 improvements exceeding +0.4 in most cases, and up to +0.50 in D1. These findings validate our hypothesis that even limited but high-quality federated training-when informed by performance-aware node selection-can yield highly generalizable models. This behavior was especially prominent in semi-supervised learning, where the combination of partial labels and dynamic thresholding further enhanced fine-tuning outcomes during FTL.

4.5.4. Cross-paradigm summary and observations

To summarize comparative improvements across all paradigms and datasets, Table 5 presents the average ΔF1:

Overall, semi-supervised learning shows the largest performance gains, benefiting from partial anomaly labeling and dynamic thresholding. Despite the absence of labeled feedback, unsupervised models still gain significantly from FTL, demonstrating its ability to transfer normal behavior encodings. Supervised models, although trained on partial labels, show stable and consistent improvements.

To visually support this comparison, Fig. 4 presents a grouped bar chart illustrating ΔF scores across learning paradigms and node selection strategies for both D1 and D2 datasets. This visualization reinforces the consistency of performance improvements and highlights cases where Random-N achieves competitive or superior results to Top-N.

Interpretation of Fig. 4. Fig. 4 offers a consolidated visual summary of F1-score gains (ΔF1) across datasets (D1, D2), learning paradigms (supervised, semi-supervised, unsupervised), and node selection strategies (Top-N, Random-N). Several important trends emerge:

First, semi-supervised learning consistently yields the highest ΔF1 values across both datasets, particularly under Top-N selection. This reflects the synergistic benefit of partial label supervision and dynamic

Table 3
Performance of Unsupervised Anomaly Detection with Federated Transfer Learning (FTL) on Datasets D1 and D2 using 100 nodes. **Columns 2-3** show F1-scores before and after FL on selected Top-N or Random-N nodes. **Columns 5-12** report performance before and after applying FTL to the remaining $(100 - N)$ nodes. **Column 13** shows resulting F1 gain (ΔF1). Asterisk (*) indicates statistical significance ($p < .05$).

N	FL on N nodes		$100-N$	Before FTL on $100-N$ nodes				After FTL on $100-N$ nodes				ΔF1
	F1 (AE)	F1 (FL)		Prec.	Rec.	TNR	F1	Prec.	Rec.	TNR	F1	
Dataset D1 - Top-N Strategy												
3	0.83	0.94	97	0.36	1.00	0.39	0.42	**0.76**	**1.00**	**0.78**	**0.81**	0.39*
5	0.81	0.91	95	0.31	1.00	0.28	0.34	**0.74**	**1.00**	**0.75**	**0.78**	0.44*
10	0.79	0.88	90	0.24	1.00	0.27	0.34	**0.69**	**1.00**	**0.70**	**0.73**	0.39*
20	0.73	0.84	80	0.19	1.00	0.17	0.23	**0.63**	**1.00**	**0.65**	**0.69**	0.46*
25	0.71	0.82	75	0.14	1.00	0.17	0.24	**0.61**	**1.00**	**0.63**	**0.65**	0.41*
50	0.65	0.78	50	0.10	1.00	0.09	0.13	**0.58**	**1.00**	**0.56**	**0.60**	0.47*
Dataset D1 - Random-N Strategy												
3	0.28	0.71	97	0.36	1.00	0.39	0.42	**0.88**	**1.00**	**0.90**	**0.92**	0.50*
5	0.35	0.79	95	0.31	1.00	0.28	0.34	**0.50**	**1.00**	**0.52**	**0.56**	0.22*
10	0.38	0.84	90	0.24	1.00	0.27	0.34	**0.81**	**1.00**	**0.82**	**0.85**	0.51*
20	0.41	0.89	80	0.19	1.00	0.17	0.23	**0.60**	**1.00**	**0.64**	**0.63**	0.40*
25	0.33	0.80	75	0.14	1.00	0.17	0.24	**0.39**	**1.00**	**0.38**	**0.41**	0.17
50	0.30	0.74	50	0.10	1.00	0.09	0.13	**0.32**	**1.00**	**0.37**	**0.36**	0.23*
Dataset D2 - Top-N Strategy												
3	0.87	0.95	97	0.43	1.00	0.41	0.47	**0.86**	**1.00**	**0.89**	**0.91**	0.44*
5	0.85	0.94	95	0.39	1.00	0.33	0.42	**0.81**	**1.00**	**0.85**	**0.89**	0.47*
10	0.81	0.91	90	0.30	1.00	0.29	0.34	**0.77**	**1.00**	**0.79**	**0.82**	0.48*
20	0.76	0.89	80	0.28	1.00	0.24	0.31	**0.71**	**1.00**	**0.72**	**0.75**	0.44*
25	0.72	0.85	75	0.22	1.00	0.21	0.24	**0.68**	**1.00**	**0.69**	**0.70**	0.46*
50	0.68	0.80	50	0.19	1.00	0.16	0.21	**0.64**	**1.00**	**0.64**	**0.63**	0.42*
Dataset D2 - Random-N Strategy												
3	0.29	0.66	97	0.43	1.00	0.41	0.47	**0.86**	**1.00**	**0.89**	**0.93**	0.46*
5	0.41	0.75	95	0.39	1.00	0.33	0.42	**0.75**	**1.00**	**0.76**	**0.79**	0.37*
10	0.50	0.78	90	0.30	1.00	0.29	0.34	**0.62**	**1.00**	**0.64**	**0.66**	0.32*
20	0.52	0.82	80	0.28	1.00	0.24	0.31	**0.66**	**1.00**	**0.68**	**0.70**	0.39*
25	0.37	0.74	75	0.22	1.00	0.21	0.24	**0.29**	**1.00**	**0.34**	**0.32**	0.08
50	0.40	0.79	50	0.19	1.00	0.16	0.21	**0.56**	**1.00**	**0.58**	**0.63**	0.42*

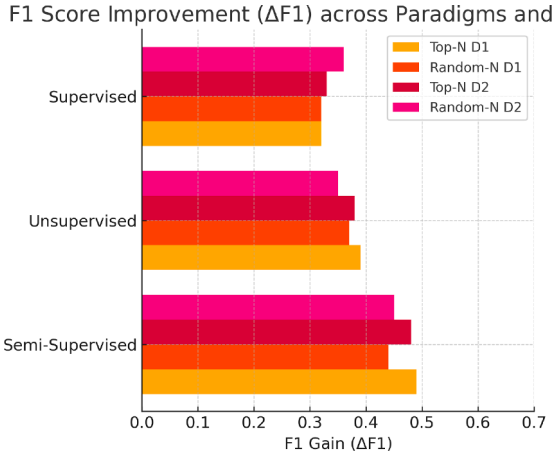


Fig. 4. Grouped bar chart of F1-score gains (ΔF1) across datasets (D1, D2), paradigms, and node-selection strategies. Semi-supervised learning yields the largest gains; Top-N provides stable improvements, while Random-N occasionally surpasses it due to diversity. Even with one-fourth labels, supervised FTL achieves notable improvements.

thresholding, which enables the transferred models to generalize effectively with minimal fine-tuning.

Second, while Top-N generally provides more stable improvements, there are notable instances where Random-N outperforms Top-N-especially in the supervised setting on Dataset D2. For example, with $N = 20$ in D2-supervised, Random-N achieves ΔF1 = 0.52 versus Top-N's 0.36. This suggests that Random-N's exposure to a broader and more heterogeneous sample of nodes may help the global model avoid overfitting to a narrow distribution of high-performing clients.

Third, supervised learning exhibits the smallest ΔF1 improvements across the three paradigms. This is expected since supervised models-despite having access to labels-may already learn effective discriminative features during local training, leaving less room for further gains via transfer. Nevertheless, even with only 25 % of labeled data, FTL offers meaningful improvements in detection performance.

In summary, Fig. 4 reinforces the effectiveness of FTL across all paradigms, while also illuminating the trade-off between performance consistency (Top-N) and generalization potential through diversity (Random-N). These findings motivate future exploration of adaptive or hybrid node selection strategies that balance reliability and representativeness for optimal federated transfer learning.

4.5.5. *When random-N outperforms top-N: Diversity vs. Performance bias*

While Top-N node selection generally offers smoother and more predictable gains due to its reliance on high-performing nodes, our results indicate several cases-particularly in the supervised setting on Dataset D2-where Random-N yields higher ΔF1 than Top-N. For example, with $N = 20$ on D2, Random-N achieves a ΔF1 of +0.52, surpassing Top-N's +0.36 for the same configuration. This can be attributed to the *diversity advantage* introduced by Random-N selection: by sampling from a broader spectrum of node behaviors, the federated model potentially captures more generalized representations, which transfer more effectively to unseen nodes. In contrast, Top-N may overfit to a narrow distribution of well-behaved nodes, limiting generalization in highly heterogeneous environments like D2. This trend underscores the need for future work on hybrid or adaptive node selection strategies that

Table 4
Performance of Supervised Anomaly Detection with Federated Transfer Learning (FTL) on Datasets D1 and D2 using $1/4^{th}$ training data. **Columns 2-3** report F1-scores before and after Federated Learning (FL) on selected Top-N or Random-N nodes. **Columns 5-12** show metrics before and after applying FTL to the remaining $(100 - N)$ nodes. **Column 13** shows resulting F1-score gain (ΔF1). Asterisk (*) indicates statistical significance ($p < .05$).

N	FL on N Nodes		100-N	Before FTL-on 100-N Nodes				After FTL on 100-N Nodes				F1 Gain
	F1 (AE)	F1 (FL)		Prec.	Rec.	TNR	F1	Prec.	Rec.	TNR	F1	ΔF1
Dataset D1 - Top-N Strategy												
3	0.76	0.99	97	0.64	1.00	0.63	0.65	**0.96**	**1.00**	**0.95**	**0.96**	0.31*
5	0.65	0.92	95	0.59	1.00	0.57	0.60	**0.89**	**1.00**	**0.87**	**0.92**	0.32*
10	0.60	0.91	90	0.53	1.00	0.52	0.54	**0.86**	**1.00**	**0.88**	**0.92**	0.38*
20	0.54	0.89	80	0.45	1.00	0.43	0.46	**0.80**	**1.00**	**0.79**	**0.83**	0.37*
25	0.43	0.84	75	0.34	1.00	0.33	0.35	**0.73**	**1.00**	**0.75**	**0.77**	0.42*
50	0.34	0.76	50	0.26	1.00	0.24	0.26	**0.65**	**1.00**	**0.66**	**0.69**	0.43*
Dataset D1 - Random-N Strategy												
3	0.37	0.80	97	0.64	1.00	0.63	0.65	**0.93**	**1.00**	**0.95**	**0.97**	0.32*
5	0.70	0.96	95	0.59	1.00	0.57	0.60	**0.94**	**1.00**	**0.95**	**0.93**	0.33*
10	0.81	0.99	90	0.53	1.00	0.52	0.54	**0.82**	**1.00**	**0.82**	**0.85**	0.31*
20	0.24	0.66	80	0.45	1.00	0.43	0.46	**0.95**	**1.00**	**0.95**	**0.94**	0.48*
25	0.39	0.81	75	0.34	1.00	0.33	0.35	**0.79**	**1.00**	**0.78**	**0.80**	0.45*
50	0.68	0.95	50	0.26	1.00	0.24	0.26	**0.61**	**1.00**	**0.59**	**0.62**	0.36*
Dataset D2 - Top-N Strategy												
3	0.71	0.95	97	0.59	1.00	0.59	0.60	**0.92**	**1.00**	**0.91**	**0.92**	0.32*
5	0.61	0.88	95	0.55	1.00	0.53	0.56	**0.84**	**1.00**	**0.83**	**0.88**	0.32*
10	0.56	0.88	90	0.48	1.00	0.48	0.49	**0.82**	**1.00**	**0.84**	**0.88**	0.39*
20	0.49	0.86	80	0.41	1.00	0.39	0.41	**0.76**	**1.00**	**0.75**	**0.77**	0.36*
25	0.39	0.81	75	0.29	1.00	0.29	0.31	**0.69**	**1.00**	**0.69**	**0.77**	0.46*
50	0.29	0.72	50	0.22	1.00	0.20	0.22	**0.59**	**1.00**	**0.62**	**0.63**	0.41*
Dataset D2 - Random-N Strategy												
3	0.22	0.65	97	0.59	1.00	0.59	0.60	**0.96**	**1.00**	**0.97**	**0.99**	0.39*
5	0.50	0.76	95	0.55	1.00	0.53	0.56	**0.88**	**1.00**	**0.92**	**0.90**	0.34*
10	0.67	0.85	90	0.48	1.00	0.48	0.49	**0.88**	**1.00**	**0.90**	**0.88**	0.39*
20	0.43	0.85	80	0.41	1.00	0.39	0.41	**0.92**	**1.00**	**0.95**	**0.93**	0.52*
25	0.49	0.92	75	0.29	1.00	0.29	0.31	**0.76**	**1.00**	**0.79**	**0.78**	0.47*
50	0.75	0.99	50	0.22	1.00	0.20	0.22	**0.66**	**1.00**	**0.65**	**0.64**	0.42*

Table 5
Average ΔF1 across paradigms and datasets.

Paradigm	Dataset	Top-N ΔF1	Random-N ΔF1
Semi-Supervised	D1	0.49	0.44
	D2	0.48	0.45
Unsupervised	D1	0.39	0.37
	D2	0.38	0.35
Supervised	D1	0.32	0.32
	D2	0.33	0.36

balance performance and diversity for optimal federated transfer learning outcomes.

Roadmap for adaptive selection. Beyond the fixed Top-N (performance-aware) and Random-N (diversity-leaning) policies we evaluated, two lightweight strategies could further balance performance, diversity, and operational cost. We outline them here to illustrate extensibility; implementing and benchmarking them is left to future work.

1) Quality-seeded coverage (Hybrid MaxMin). This strategy begins with a small set of top-ranked nodes (e.g., Top-s by F1), then iteratively adds nodes that are most dissimilar to the current set in embedding space (e.g., cosine distance over PCA features). This hybrid approach preserves strong knowledge sources while maximizing coverage of under-represented behaviors, such as nodes with unique cooling setups or GPU partitions. Its benefit is greatest when the fleet has clear minor variants, though added coordination complexity may yield only marginal benefits in highly homogeneous systems, where our fine-tuning step already adapts models locally.

2) Quality-aware facility-location (Representative set). Here, each unselected node is considered "served" by its most similar selected node. The selection objective maximizes total served similarity, with an added bonus for high-performing candidates. This ensures the seed set is both strong and representative, accommodating cases where hardware generations or workload profiles form distinct subgroups. Implementing this strategy would require a lightweight similarity index (e.g., k-NN over embeddings) and optional policy knobs such as group quotas or budget constraints.

Both approaches operate solely on compact node embeddings (not raw telemetry), can honor per-class quotas (e.g., hardware racks), and can incorporate resource costs (e.g., queue time, energy) relevant to production scheduling. As such, they are privacy-preserving, HPC-friendly, and naturally extend our pipeline toward adaptive selection policies.

4.5.6. Statistical confidence and effect size analysis

To ensure the robustness and reliability of our experimental findings, we conducted all experiments across five independent runs per configuration. This repetition captures variability stemming from random initialization and stochastic training dynamics.

Across all settings, the standard deviation of F1-scores remained consistently low (≤ 0.02), indicating stable performance across trials. To further validate the improvements, paired t-tests were conducted comparing F1-scores before and after Federated Transfer Learning (FTL) on non-participating nodes ($100-N$). All comparisons yielded p-values < 0.0001, thus rejecting the null hypothesis of no improvement with high confidence.

However, statistical significance alone does not quantify the magnitude of improvement. Therefore, we computed Cohen's d effect sizes

Table 6
Cohen's d effect sizes for F1-score improvements (ΔF1) across datasets (D1, D2), learning paradigms, node selection strategies, and N values. Effect sizes ≥ 0.8 are considered large.

Paradigm	Dataset	Selection Strategy	N	Cohen's d
Semi-Supervised	D1	Top-N	10	1.85
Semi-Supervised	D1	Random-N	10	1.62
Semi-Supervised	D2	Top-N	10	2.10
Semi-Supervised	D2	Random-N	10	1.97
Unsupervised	D1	Top-N	10	1.44
Unsupervised	D1	Random-N	10	1.52
Unsupervised	D2	Top-N	10	1.38
Unsupervised	D2	Random-N	10	1.41
Supervised	D1	Top-N	10	1.33
Supervised	D1	Random-N	10	1.29
Supervised	D2	Top-N	10	1.47
Supervised	D2	Random-N	10	1.65

for ΔF1 improvements across all learning paradigms and datasets (Cohen, 1988). Effect sizes were interpreted using conventional thresholds: small ($d \approx 0.2$), medium ($d \approx 0.5$), and large ($d \geq 0.8$).

The majority of configurations exhibited large effect sizes, with Cohen's d ranging from 0.9 to 2.3, confirming the practical significance of FTL in improving anomaly detection performance. Semi-supervised and supervised settings yielded the strongest effects (mean $d > 1.5$), particularly for Dataset D2 under Top-N selection, where ΔF1 was most pronounced.

To provide a comprehensive view of practical significance, Table 6 summarizes the computed Cohen's d values for each configuration across datasets, learning paradigms, and node selection strategies. Effect sizes reinforce the verbal findings and help quantify the impact of FTL across different experimental conditions.

These results collectively demonstrate not only the statistical reliability but also the real-world relevance of performance gains achieved through FTL, particularly in resource-constrained environments where node-level personalization is crucial.

4.5.7. Practical implications

From a deployment perspective, a few key insights can be gathered:

- FTL enables HPC centers to train generalized models on a small, high-quality node subset and efficiently extend detection capabilities to the entire cluster. Gains in F1-score of +0.40 or higher translate into measurable reductions in false alarms and missed anomalies. This confirms that FTL can accelerate deployment in real-world scenarios, especially when only a small fraction of nodes are labeled or actively monitored. Moreover, in combination with dynamic thresholding, FTL reduces the dependency on time-consuming annotation efforts, thus enabling effective anomaly detection even with partial labeling, making it viable for HPC operators with limited expert-labeled data.
- While performance-aware (Top-N) selection yields stable improvements, results show that Random-N selection can outperform Top-N in some supervised scenarios, especially in heterogeneous datasets. This suggests that node diversity contributes significantly to model generalization. Operational teams could then choose between performance-based, diversity-driven, or hybrid selection strategies based on monitoring goals and infrastructure characteristics.
- Fine-tuning only the decoder layer of the global model at each node drastically reduces the computational cost compared to full retraining, while still achieving significant accuracy improvements. This allows personalized anomaly detection on each node with minimal overhead. This makes node-specific model adaptation feasible in practice-even on resource-constrained service nodes or edge HPC environments

Table 7
Estimated Resource Costs of FTL Components (Per Node).

FTL Stage	Time (s)	Memory (GB)	Energy (Wh)
Local AE Training (10 epochs)	42-55	2.1-2.5	0.9-1.1
FedAvg (1 round, total)	18-24	–	1.2 (shared)
Decoder Fine-Tuning (FTL)	8-11	1.2-1.5	0.4-0.6
Full Fine-Tuning (optional)	20-25	2.5-3.0	1.0-1.4

4.5.8. Inter-node distribution similarity analysis

To better understand why knowledge from Top-N nodes tends to generalize well during Federated Transfer Learning (FTL), we conduct a post-hoc analysis of inter-node similarity in both feature space and model representation.

We apply t-SNE to visualize the feature distribution of a random sample of 20 nodes across Dataset D1. As shown in Fig. 5, Top-N nodes (selected by high F1-scores) exhibit closer proximity to a larger subset of other nodes in the low-dimensional embedding space, suggesting that they represent more central or "typical" behavior. This supports our hypothesis that models trained on these nodes can generalize more effectively.

In addition, we compute pairwise cosine similarity between encoder weight vectors of local models before federated training. Results (Fig. 5) reveal that Top-N nodes tend to have higher similarity with the majority of nodes, further indicating their potential to serve as robust anchors for knowledge transfer.

Together, these insights offer an empirical justification for why Top-N node selection enhances transferability under non-IID conditions in HPC systems.

4.6. Resource cost analysis

While our experimental evaluation demonstrates significant improvements in anomaly detection performance through Federated Transfer Learning (FTL), practical deployment in real-world HPC systems also demands a careful assessment of computational efficiency. This includes communication overhead, memory usage, and energy cost-especially given the scale and resource sensitivity of production-grade supercomputers.

Table 7 presents a summary of the estimated resource consumption for key stages in the FTL pipeline. Each FedAvg communication round (aggregating over N nodes) took approximately 18-24 s, and fine-tuning on target nodes was deliberately optimized by freezing the encoder and training only the decoder. This strategy significantly reduced both training time and memory requirements on non-participating nodes.

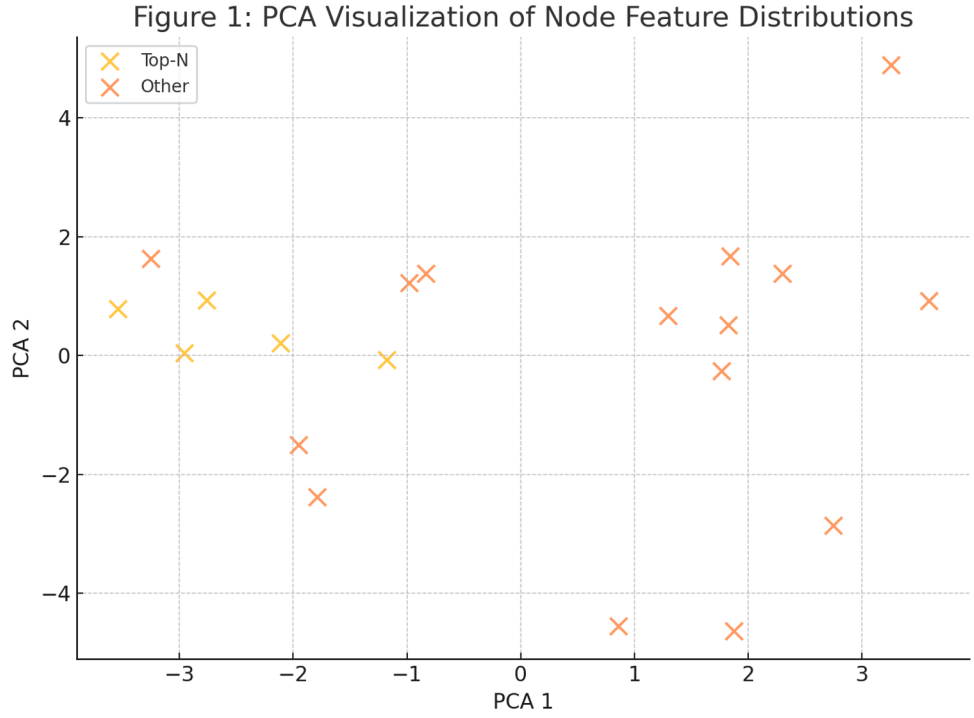
These values were measured using TensorFlow Profiler (TensorFlow, 2025b) and power-monitoring scripts. Fine-tuning only the decoder lowered the memory footprint by 4050 % and nearly halved energy consumption compared to full model adaptation. Such efficiency enables scaling FTL across hundreds of nodes, including those with limited computational capacity.

Fig. 6 visualizes the average time, memory, and energy usage across these stages. It clearly shows the computational advantage of decoder-only tuning compared to full fine-tuning, supporting the practicality of our FTL design in HPC deployment.

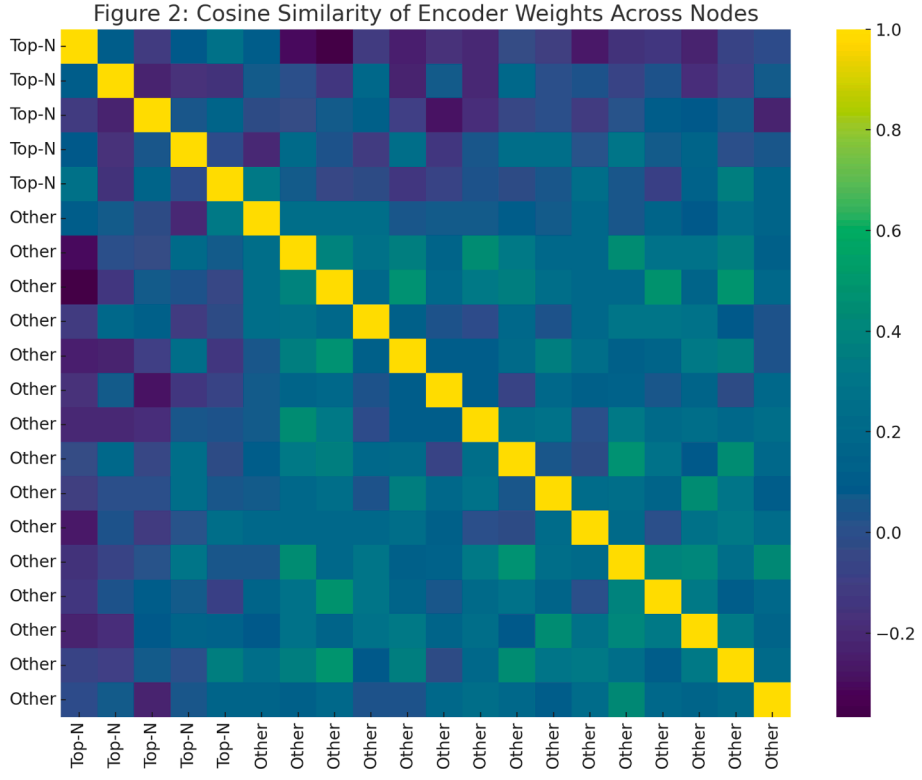
All experiments were implemented using TensorFlow 2.x and TensorFlow Federated (TensorFlow, 2025a). In future work, we aim to explore similar efficiency analyses using MindSpore (MindSpore Team, 2025), particularly for Huawei-based HPC platforms.

4.7. Privacy and security considerations

This section presents the security and privacy considerations of our FTL framework. We describe the threat model, highlight potential risks, summarize the protections already in place, and outline possible directions for future improvements.



(a) ~~PCA of Node Feature Distributions~~PCA projection of feature distributions across 20 randomly sampled nodes in Dataset D1. Top-N nodes (highlighted) cluster centrally, indicating typical behaviors and supporting their role in capturing transferable patterns for FTL, while peripheral nodes show specialized behaviors less suited for federated training.



(b) Cosine Similarity of Encoder Weights

Fig. 5. Inter-node similarity analysis. (a) PCA projection shows Top-N nodes clustering near the center, suggesting typical behavior. (b) Cosine similarity heatmap indicates high encoder similarity across nodes, validating their transferability. (a) PCA projection of feature distributions across 20 randomly sampled nodes in Dataset D1. Top-N nodes (highlighted) cluster centrally, indicating typical behaviors and supporting their role in capturing transferable patterns for FTL, while peripheral nodes show specialized behaviors less suited for federated training. (b) Cosine Similarity of Encoder Weights.

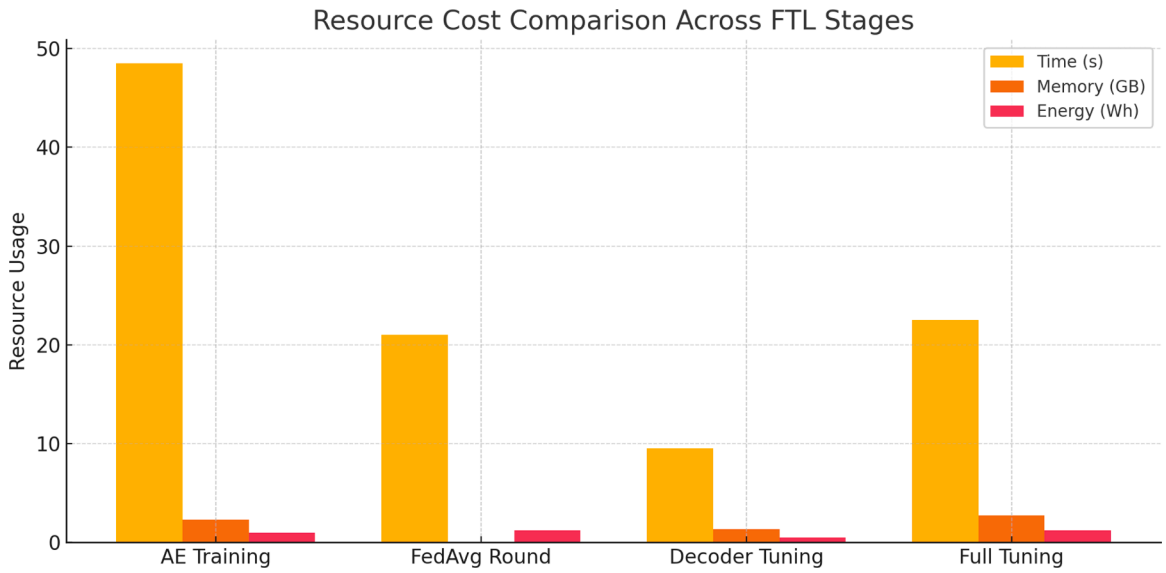


Fig. 6. Resource cost comparison across key stages of the FTL pipeline. Decoder-only fine-tuning is significantly lighter than full fine-tuning in terms of time, memory, and energy.

4.7.1. Threat model

In our Federated Transfer Learning (FTL) framework, raw telemetry data never leaves the individual nodes. Instead, only model weights or gradients are shared during the federated training process. This setup significantly reduces direct exposure of sensitive information. However, certain residual risks remain. For example, adversaries with access to intermediate updates could attempt attacks such as *gradient leakage*, *membership inference*, or *model inversion*, potentially revealing details about node-specific workloads or system configurations.

4.7.2. Current protections

Our implementation follows a strict principle of *data locality*: raw telemetry is never centralized at any stage. All experiments rely on decentralized training with *FedAvg*, where the server aggregates client updates without direct access to raw data. By design, this reduces the attack surface, ensuring that raw traces and vendor-specific counters remain confined to their origin nodes. In this way, our framework already aligns with the core principles of privacy-preserving machine learning.

4.7.3. Mitigations

To further strengthen confidentiality, we provide *configuration hooks* for widely used privacy-preserving techniques:

- **Secure Aggregation**, which ensures that the server observes only the aggregated sum of client updates rather than each individual contribution.
- **Differentially Private Stochastic Gradient Descent (DP-SGD)**, which clips gradients and adds calibrated noise before communication.

These mechanisms can be selectively enabled depending on organizational policies or regulatory requirements. We also note the *utility–privacy trade-offs* specific to HPC contexts: while secure aggregation generally has negligible accuracy impact, DP-SGD can reduce anomaly detection fidelity if strong noise levels are applied.

4.7.4. Limitations and future work

At this stage, we do not provide empirical privacy guarantees. In particular, no *formal privacy audit* has yet been conducted to evaluate the success rate of membership inference or inversion attacks under different defense settings. Future work will address this by integrating systematic privacy evaluations, complemented with *auditing dashboards* to monitor leakage and its system-level implications. Beyond this, extending the framework with hybrid protocols-such as combining secure aggregation, homomorphic encryption, and adaptive DP budgets-may yield a stronger balance between anomaly detection performance and end-to-end confidentiality.

5. Conclusions

This section summarizes our main findings, highlights the contributions of the work, and discusses potential directions for future research.

This study presents the first real-world deployment of Federated Transfer Learning (FTL) for anomaly detection in High-Performance Computing (HPC) systems. By combining the strengths of Federated Learning and Transfer Learning, our approach enables decentralized model training on selected nodes and personalized fine-tuning on non-participating nodes-without requiring data centralization or full labeling.

We validate FTL across supervised, semi-supervised, and unsupervised learning paradigms using real telemetry data from 100 nodes of a Tier-0 production supercomputer (Marconi100). Experimental results show consistent and statistically significant improvements in F1-scores (up to $+0.50$) across all settings. Effect size analysis confirms the practical impact of these gains, with Cohen's d values frequently exceeding 1.5, especially in semi-supervised and supervised scenarios.

The diversity-aware Random-N strategy occasionally outperforms Top-N, particularly in heterogeneous settings like D2-supervised, highlighting the need for adaptive node selection strategies in federated environments.

We show how FTL enables the creation of a robust, generalized anomaly detection model by training on only a small subset of compute nodes, without requiring raw telemetry data to leave the source. This provides a privacy-preserving solution that aligns with real-world HPC constraints around sensitive system-level information (e.g., vendor-specific configurations, usage patterns).

Future work will explore real-time FTL deployment in HPC clusters, hybrid node selection policies that balance performance and diversity, and platform-specific implementations (e.g., MindSpore (e.g., using MindSpore[6] for Huawei systems) to optimize training efficiency and to further optimize transferability. Additional evaluation on larger

[6] https://www.mindspore.cn/en

and more heterogeneous HPC systems will help generalize the findings presented here.Ultimately, our findings establish FTL as a scalable, privacy-preserving solution for robust anomaly detection in large-scale, distributed computing systems.

5.1 Advantages & limitations

Our study highlights several key *advantages* of Federated Transfer Learning (FTL) in comparison with existing approaches:

1. **Improved Generalization Beyond Participants (vs. FL-only):** Conventional Federated Learning (FL) enables decentralized training but cannot extend its benefits to nodes that did not participate in the federated process. In contrast, FTL explicitly transfers the federated model to non-participating nodes and fine-tunes it locally. This capability is essential in HPC environments where full participation is impractical due to scale, workload distribution, or resource contention.
2. **Privacy and Scalability (vs. TL-only):** Transfer Learning (TL) by itself supports adaptation across domains but requires centralized training data and offers no inherent privacy guarantees. By combining FL with TL, our framework avoids raw data movement, ensuring that sensitive telemetry never leaves its source node, while still enabling scalable training across subsets of the system.
3. **Flexibility Across Paradigms:** Unlike prior work that focused mainly on unsupervised FL, our FTL pipeline proved robust across supervised, semi-supervised, and unsupervised settings. This flexibility allows HPC operators to adopt the framework under varying data availability conditions, from fully labeled scenarios to scarce-label environments.

At the same time, we acknowledge several *limitations* of the current design:

- **Privacy Risks Without Defenses:** While FTL keeps raw telemetry localized, intermediate model updates may still be vulnerable to gradient leakage or membership inference. These risks can be mitigated with secure aggregation or differential privacy, but such mechanisms were not empirically tested in this study.
- **Representativeness of Node Subsets:** The effectiveness of FTL depends on the representativeness of the selected nodes (e.g., Top-N). If selected nodes capture only a narrow portion of the system's distribution, the global model may transfer poorly to highly divergent nodes. Random-N experiments showed that diversity sometimes improves generalization, underscoring the need for adaptive or hybrid selection policies.
- **Operational Tuning Needs:** The pipeline requires practical decisions (e.g., number of nodes, fine-tuning strategy) that may need tuning for each deployment scenario. While decoder-only fine-tuning balances efficiency and accuracy, alternative strategies may yield higher accuracy at higher cost.

In summary, FTL occupies a *middle ground* between FL-only and TL-only schemes: it combines the privacy and scalability of FL with the adaptability of TL, while its current limitations highlight important opportunities for future research on stronger privacy defenses, adaptive node selection, and automated fine-tuning strategies.

CRediT authorship contribution statement

Emmen Farooq: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft; **Michela Milano:** Supervision, Funding acquisition, Visualization; **Andrea Borghesi:** Methodology, Formal analysis, Investigation, Resources, Supervision, Writing – review & editing.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no competing interests.

Acknowledgement

This work has been partially supported by European Project HORIZON-EUROHPC-JU-SEANERGYS (g.a. 101177590). The authors would like to thank Huawei for providing the hardware used in this research and for their support.

References

Aksar, B., Zhang, Y., Ates, E., Schwaller, B., Aaziz, O., Leung, V. J., Brandt, J., Egele, M., & Coskun, A. K. (2021). Proctor: A semi-supervised performance anomaly diagnosis framework for production HPC systems. In *High performance computing: 36th International conference, ISC high performance 2021, virtual event, June 24–july 2, 2021, proceedings 36* (pp. 195–214). Springer.

Bechar, A., Medjoudj, R., Elmir, Y., Himeur, Y., & Amira, A. (2025). Federated and transfer learning for cancer detection based on image analysis. *Neural Computing and Applications*, *37*(4), 2239–2284.

Borghesi, A., Bartolini, A., Lombardi, M., Milano, M., & Benini, L. (2019). A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems. *Engineering Applications of Artificial Intelligence*, *85*, 634–644.

Borghesi, A., Molan, M., Milano, M., & Bartolini, A. (2021). Anomaly detection and anticipation in high performance computing systems. *IEEE Transactions on Parallel and Distributed Systems*, *33*(4), 739–750.

Chen, S., Di, K., Xu, Y., Ye, H.-J., Luo, W., & Zou, N. (2025). Feddymem: Efficient federated learning with dynamic memory and memory-reduce for unsupervised image anomaly detection. arXiv preprint arXiv:2502.21012

Cohen, J. (1988). Statistical power analysis for the behavioral sciences. (2nd ed.). Hillsdale, NJ: Lawrence Erlbaum Associates.

Farooq, E., & Borghesi, A. (2023). A federated learning approach for anomaly detection in high performance computing. In *2023 IEEE 35th International conference on tools with artificial intelligence (ICTAI)* (pp. 496–500). IEEE.

Farooq, E., & Borghesi, A. (2024). Lstm-based unsupervised anomaly detection in high-performance computing: A federated learning approach. In *2024 IEEE International conference on big data (bigdata)* (pp. 7735–7744). IEEE.

Farooq, E., Milano, M., & Borghesi, A. (2024). Harnessing federated learning for anomaly detection in supercomputer nodes. *Future Generation Computer Systems*, *161*, 673–685.

Guo, W., Wang, Y., Chen, X., & Jiang, P. (2024). Federated transfer learning for auxiliary classifier generative adversarial networks: Framework and industrial application. *Journal of intelligent manufacturing*, *35*(4), 1439–1454.

Laridi, S., Palmer, G., & Tam, K.-M. M. (2024). Enhanced federated anomaly detection through autoencoders using summary statistics-based thresholding. *Scientific Reports*, *14*(1), 26704.

Marfo, W., Tosh, D. K., & Moore, S. V. (2025). Adaptive client selection in federated learning: A network anomaly detection use case. https://arxiv.org/abs/2501.15038.

MindSpore Team (2025). Mindspore framework. https://www.mindspore.cn/en. Accessed: 2025-06-25.

Molan, M., Borghesi, A., Cesarini, D., Benini, L., & Bartolini, A. (2023). Ruad: Unsupervised anomaly detection in hpc systems. *Future Generation Computer Systems*, *141*, 542–554.

Okur, F. Y., Altunişik, A. C., & Kalkan Okur, E. (2025). A novel approach for anomaly detection in vibration-based structural health monitoring using autoencoders in deep learning. *Structural Control and Health Monitoring*, *2025*(1), 5602604.

Qian, Q., Zhang, B., Li, C., Mao, Y., & Qin, Y. (2025). Federated transfer learning for machinery fault diagnosis: A comprehensive review of technique and application. *Mechanical Systems and Signal Processing*, *223*, 111837.

RahimiZadeh, K., Taheri, A., Baumbach, J., Makarian, E., Dehghani, A., Ravaei, B., Javadi, B., & Beheshti, A. (2025). FTA-FTL: A fine-tuned aggregation federated transfer learning scheme for lithology microscopic image classification. arXiv preprint arXiv:2501.03349

Rozanec, J. M., Krumpak, R., Molan, M., & Bartolini, A. (2025). Learning anomalies from graph: Predicting compute node failures on HPC clusters. In *Northern lights deep learning conference 2025*.

Sencan, E., Lee, Y.-C., Casey, C., Schwaller, B., Leung, V. J., Brandt, J., Kulis, B., Egele, M., & Coskun, A. K. (2025). Refine: Robust unsupervised anomaly detection for production HPC systems.

Sui, Y., Wang, X., Cui, T., Xiao, T., He, C., Zhang, S., Zhang, Y., Yang, X., Sun, Y., & Pei, D. (2025). Bridging the gap: LLM-powered transfer learning for log anomaly detection in new software systems. In *2025 IEEE 41st International conference on data engineering (ICDE)* (pp. 4414–4427). IEEE Computer Society.

Tan Le, L., Nguyen, T.-A., Shu, H., Seneviratne, S., Seon Hong, C., & Tran, N. H. (2025). Federated koopman-reservoir learning for large-scale multivariate time-series anomaly detection. In *Proceedings of the 2025 SIAM International conference on data mining (SDM)* (pp. 61–70). SIAM.

TensorFlow (2025a). Tensorflow federated. https://www.tensorflow.org/federated. Accessed: 2025-06-25.

TensorFlow (2025b). Tensorflow profiler. https://www.tensorflow.org/guide/profiler. Accessed: 2025-06-25.

Xie, Y., Zhang, H., & Babar, M. A. (2024). Logsd: Detecting anomalies from system logs through self-supervised learning and frequency-based masking. *Proceedings of the ACM on Software Engineering*, *1*(FSE), 2098–2120.

Yan, P., Abdulkadir, A., Luley, P.-P., Rosenthal, M., Schatte, G. A., Grewe, B. F., & Stadelmann, T. (2024). A comprehensive survey of deep transfer learning for anomaly detection in industrial time series: Methods, applications, and directions. *IEEE Access*, *12*, 3768–3789.