IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 May 2025, accepted 19 May 2025, date of publication 22 May 2025, date of current version 4 June 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3572605

RESEARCH ARTICLE

PPFL-DCS: Privacy-Preserving Federated Learning Using Neural Transformer and Leveraging Dynamic Client Selection to Accommodate Data Diversity

NAKUL MEHTA[1], NITESH BHAROT[1], (Senior Member, IEEE), JOHN G. BRESLIN[1], (Senior Member, IEEE), AND PRIYANKA VERMA[2], (Senior Member, IEEE)

[1]Data Science Institute, University of Galway, Galway, H91 TK33 Ireland
[2]School of Computer Science, University of Galway, Galway, H91 TK33 Ireland

Corresponding author: Nitesh Bharot (nitesh.bharot@universityofgalway.ie)

This work was supported in part by the Taighde Éireann—Research Ireland under Grant 12/RC/2289_P2 (Insight) and Grant 21/FFP-A/9174 (SustAIn), and in part by the Interreg Atlantic Area co-funded by European Union under Grant EAPA 0016/2022 (ENEPORTS).

ABSTRACT The vulnerabilities and security issues of industrial Cyber-Physical Systems (CPSs), such as Intrusion Detection Systems (IDSs), have significantly increased due to the rapid integration of conventional industrial setups with advanced networking and computing technologies like 5G, software-defined networking, and artificial intelligence. Coping strategies for such challenges frequently involve transferring data to a central location, which raises concerns about latency, efficiency, and privacy. To address these issues, Federated Learning (FL) was developed as a solution to mitigate both the privacy concerns of organizations and the complexities of networked systems. However, FL-based techniques still have shortcomings, FedAvg equally weights weak models, risking suboptimal results; FL also faces Membership Inference privacy attacks. To address these challenges, we propose PPFL-DCS, an FL framework that incorporates a weighted mechanism for dynamic client selection, accounting for the performance of each local model and data size of each client in integration with a Neural Transformer System (NTS) that enhances the system's robustness against the MIA attacks. The NTS limits the impact and gains of attackers, thereby reducing the effectiveness of MIAs. Extensive experiments demonstrate that PPFL-DCS achieves a high detection accuracy of 97.424% for cyber threats in industrial CPSs, and highlight its efficiency over state-of-the-art techniques.

INDEX TERMS Industry 4.0, intrusion detection system, federated learning, stacked AutoEncoders, cyber threats, membership inference attacks, neural transformer, DL, ML.

I. INTRODUCTION

The growing developments in Artificial Intelligence (AI) and Machine Learning (ML) have revolutionized diverse industries, such as Consumer Electronics, enabling the development of powerful predictive models and data-driven solutions [1], [2]. However, the effectiveness of these techniques heavily relies on accessing large-scale and diverse datasets. Traditional approaches often involve centralized data collection, raising significant concerns regarding privacy and ethical issues surrounding data security and ownership [3].

To address these challenges, a promising paradigm known as Federated Learning (FL) has emerged. FL enables collaborative model training [4] without the need for centralized data aggregation. Introduced by the Google research team

The associate editor coordinating the review of this manuscript and approving it for publication was Peter Langendoerfer.

in 2016, FL is a distributed framework that simultaneously achieves data expansion and privacy protection, FL [5], entities such as companies, factories, consumer devices, or edge devices with their respective private datasets act as clients and construct a local model with an identical neural structure & utilize its private dataset for training. A global model with a matching neural structure is established on the cloud central server. The exchange of global and local models takes place through continuous communication between the central server and multiple clients participating in the training process. Eventually, a high-performing global model is collectively created to accomplish specific learning tasks [6].

In contrast to Centralized Learning (CL), FL differs in the way it transfers the deep model itself instead of the local data of the client. This approach inevitably expands available data while preventing the original data leakage ensuring the privacy and security of sensitive information. However, the traditional FL framework assumes equal contributions from all participants [7], disregarding variations in data distributions and the significance of each participant's data and output. These limitations could result in suboptimal performance, particularly when participants possess imbalanced datasets or when certain participants contribute more valuable data [8]. The presence of poor-performing local models can impact the global model during the FL process, causing abruptness in their accuracy and hindering rapid convergence [8].

Furthermore, the FL training process introduces security issues and vulnerabilities, as the FL server can expose the model and become a target for various security threats. Shokri et al. [9] have shown the effectiveness of inference attacks, specifically Membership Inference Attacks (MIA), in revealing private information about data owners. These MIA attacks are capable of determining whether specific target data was used in training the target model (victim model) [10]. Rising privacy concerns have led to the creation of privacy-preserving methods to counter MIA, including Differential Privacy (DP) and encryption techniques. However, implementing these approaches often leads to federated models with considerable computation overhead and compromised classification performance. Consequently, FL necessitates a new solution that offers improved security and privacy while maintaining good performance.

In response to these challenges, a specialized area called privacy-preserving FL (PPFL) has emerged which aims to incorporate differential weighting mechanisms into the FL process. These methods assign appropriate weights to the local updates of each participant, taking into account the significance, utility, and performance of their data. By leveraging these weights, the FL process becomes more flexible and adaptive, leading to improved model performance while maintaining data privacy [11].

Therefore, taking all these issues and solutions into account, we propose an enhanced FL framework called PPFL-DCS, which applies a filtering mechanism to identify local models with subpar accuracy based on predetermined dynamic thresholds. In this, the aggregated weights of the local models are calculated based on their detection performance, and are dynamically adjusted and aggregated to determine the final weights. This approach ensures that only high-quality local models contribute significantly to the global model, enhancing its overall performance and convergence. Additionally, it works with, the integration of a Neural Transformer System (NTS), which aims to minimize security threats, such as MIA, and foster collaborative improvement in the performance of the global FL model. The main contributions of this paper are summarized as follows:

CONTRIBUTIONS

This paper introduces a novel distributed learning-based approach called PPFL-DCS, that aims to handle reverse engineering attacks (MIAs) during training while maintaining cost-effectiveness and enhancing stability. Additionally, the study presents a weighted FL framework, focusing on each client's contribution to the FL process. The main contributions of this paper are summarized as follows:

1) To effectively tackle the challenges arising from heterogeneous and non-IID data distributions across clients, the study introduces a Weighted Federated Framework enhanced with a Dynamic Client Selection (DCS) mechanism. This adaptive strategy prioritizes the inclusion of clients based on the relevance, quality, and representativeness of their local data. By doing so, the framework ensures that participants contributing more informative data have a proportionally greater impact on the aggregated global model, thereby improving learning efficiency and overall model accuracy.
2) A key innovation of this work is the introduction of a Neural Transformer System (NTS), specifically designed to counter Membership Inference Attacks (MIA). Unlike conventional methods such as differential privacy or cryptographic techniques, which often impose significant computational burdens and degrade model utility, NTS offers a practical alternative. It achieves robust privacy protection, making it suitable for real-world, resource-constrained environments.
3) Beyond privacy preservation, the proposed NTS framework also functions as a data quality enhancement module. It intelligently mitigates the influence of noisy or low-quality data, thereby reducing the dependence on extensive preprocessing steps. This capability not only simplifies data handling pipelines but also enhances the overall quality of insights derived from the federated training process, further amplifying the utility of the framework in practical deployments.

II. RELATED WORK

Focusing on cyber-physical systems (CPSs), this section explores one of CPS named Intrusion Detection Systems (IDS). It presents the existing methodologies and techniques for providing secure and privacy-preserving IDS.

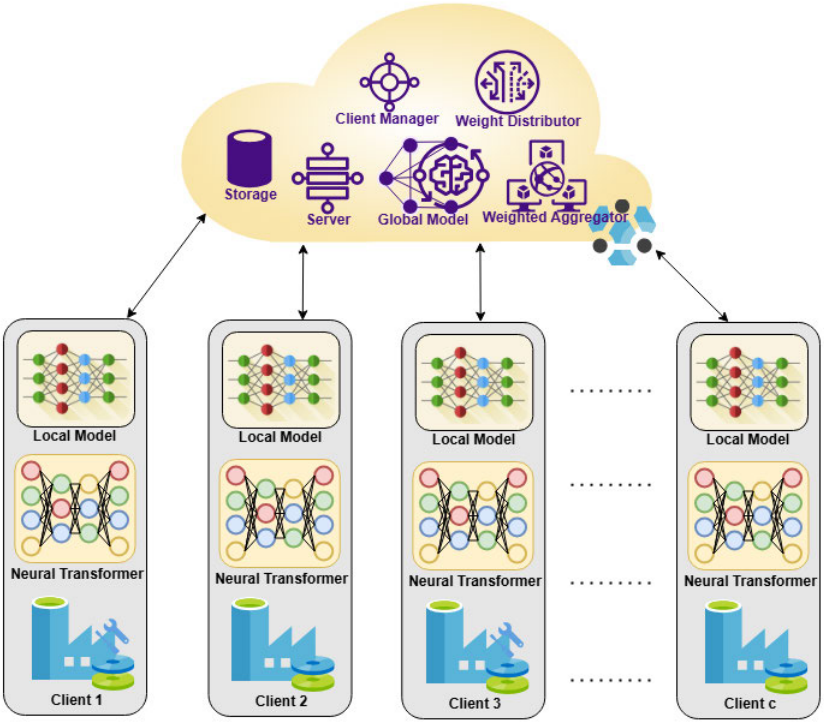

FIGURE 1. Architecture of proposed PPFL-DCS framework.

A. CONVENTIONAL ML/DL MODELS

In the context of IoT systems, Ge et al. [12] introduced an IDS approach using a DL model. It utilizes a Feed-forward Neural Network (FNN) for both binary and multi-class classification of various attacks, such as reconnaissance, DoS, DDoS, and information theft attacks. Through experiments, the results demonstrated that it surpassed the accuracy and training time achieved by the Support Vector Machine (SVM) in these classification tasks. In the field of network security, numerous researchers have developed various detection algorithms and proposed solutions for network intrusion detection. As an example, Teng et al. [13] presented an improved genetic algorithm aimed at optimizing intrusion detection models utilizing SVM. Their approach incorporated a fitness function that considered classification accuracy, false alarm rate, and data feature dimensions. Additionally, Ren et al. [14] built a weighted Naive Bayes (NB) IDS model that integrated PSO, rough set theory, and an enhanced PSO algorithm. This combination significantly enhanced the detection capacity of the system they developed.

ML and Deep Learning (DL) models offer efficient detection of attacks by identifying patterns in data. Unlike handcrafted IDS, these models can adapt better to new types of attacks. Dong et al. [15] introduced an IDS based on multivariate correlation analysis and Long Short-Term Memory (LSTM). MCA algorithm was implemented for feature selection, and intrusion classification was carried out using LSTM. Folino et al. [16] proposed an ensemble model that combined four bases of Deep Neural Network (DNN) classifiers trained on different sections of data. The meta-classifier employed the predictions from the base classifiers, along with the original instance features, to both train and predict tasks. Through experiments conducted with two datasets, the proposed ensemble model proved to be highly effective for streaming IDS.

For DDoS attack detection, Aamir et al. [17] presented a semisupervised model consisting of two steps. Firstly, Principal Component Analysis (PCA) was employed to reduce data and employ feature selection. Then, a clustering algorithm was applied to label the data based on its clusters. Subsequently, several supervised models were trained and applied for attack detection. Experiments indicated that the RF model outperformed k-Nearest Neighbors (KNN) and SVM in terms of accuracy. Similarly, Shiomoto et al. [18] built an IDS utilizing a semi-supervised learning model. It employs an Adversarial AutoEncoder (AAE) for feature extraction. Experimental findings showcased the high accuracy achieved by their model using the least number of labeled data, compared to DNNs. However, it's worth noting that the results were evaluated on the NSL-KDD dataset, lacking modern network behaviors as it is an older dataset.

B. FL BASED MODELS

To address privacy concerns in centalized ML and DL model training, FL emerged as a solution. In the field of IDS, Nguyen et al. [19] introduced DÏoT, the first FL system for

detecting compromised IoT devices. The system is composed of security gateways and IoT security services. The security gateways utilize Gated Recurrent Unit (GRU) to train local models. Subsequently, the security service aggregates these local models to form a global model. FLIDS, proposed by Friha et al. [20], utilizes FL for cybersecurity in agricultural IoT networks. FLIDS employs DNNs, Convolutional Neural Networks (CNN), & Recurrent Neural Network (RNN) models. Experimental results demonstrate its competitive performance compared to centralized models.

In the context of Industrial IoT attack detection, Aouedi et al. [21] proposed FLUIDS, a federated semi-supervised learning approach. It utilized an AE model to extract features and reduce dimensions. Followed by global aggregation, the global encoder was attached to a neural network layer for fine-tuning and supervised learning. Experimental results with real datasets demonstrated the efficacy of FLUIDS compared to centralized DL models.

Guo et al. [22] proposes TFL-DT, a novel trust evaluation scheme for Federated Learning (FL) in Digital Twin for Mobile Networks (DTMN). To address limitations in existing trust mechanisms, such as relying on single-factor evaluations and coarse-grained trust metrics the authors introduce a multi-attribute user behavior model that captures trust-related behavior in a fine-grained and comprehensive way. The trust value of a user is computed using both direct trust and recommended trust from other virtual twins. The model accounts for multiple behavioral factors, including interaction outcomes, time decay, stability, and reliability. Experimental results demonstrate the scheme's effectiveness in accurately assessing trust and detecting malicious users, particularly those with alternating good/bad behavior patterns. Another paper by Guo et al. [23] proposes a Lightweight Contribution Evaluation method for Federated Learning (LCEFL) that avoids using a server-side test dataset and reduces computational complexity through model compression. It enables trusted aggregation by weighting participants based on their contributions, achieving similar accuracy to Shapley-based methods while improving convergence speed and model performance.

In another study, Mothukuri et al. [24] introduced an ensemble FL model for IDS in IoT environments. Local training of the GRU model for different window sizes was conducted on the client, and the FL server calculated global model. An ensemble model (RF) was then applied in the FL server to enhance classification performance. Similarly, Attta et al. [25] proposed MV-FLID, an ensemble FL-based IDS. Three FNN models were developed for different views (biflow view, packet view, and uniflow view). Predictions from different models were combined using an ensemble model (RF) for instance classification.

In the pursuit of enhanced privacy preservation for end-users, Al et al. [26] introduced a federated mimic learning approach that combines FL and mimic learning. By leveraging NSL-KDD dataset, experiments demonstrated that the federated mimic learning method achieved an accuracy of 98%, comparable to centralized DL models. Importantly, this approach significantly improved the privacy preservation of user data. It should be noted that the NSL-KDD dataset may lack modern IoT-based attack scenarios.

Currently, several research studies explore the application of FL in intrusion detection. For instance, Zhao et al. [27] introduced MT-DNN-FL, a multitask FL model that addresses VPN traffic recognition, abnormal traffic detection, and traffic classification tasks. In Li et al. [28], a federated DL model was proposed for detecting attacks in industrial cyber-physical systems. Additionally, secure communication protocols were designed, ensuring safe model parameter transfers. Sun et al. [29] presented an adaptive IDS based on piecewise FL. It allowed multiple participants to share multiple global models and train similar networks under the same global model. In Zhao et al. [30], an intelligent intrusion detection model aided by LSTM was proposed based on FL, which exhibited higher precision and improved consistency compared to traditional models. Man et al. [31] introduced FedACNN for network intrusion detection, which performed weighted aggregation of local models based on the Euclidean distance. It helped to reduce the number of rounds by half in comparison to FedAvg.

C. PRIVACY PRESERVING SOLUTIONS

In order to enhance the security and privacy of user data in FL [32], various methods are employed, including differential privacy (DP), data encryption and zero-day defence frameworks [33]. However, these techniques have certain drawbacks when directly applied to FL-enabled IDS. Firstly, encryption-based methods require high computation, requiring users to manage encryption keys and perform whole processes, which can increase time and CPU usage. Secondly, DP may introduce additional noise to the model parameters, potentially compromising the overall system performance. Also, applying the same noise to all clients in FL is unfair and inefficient due to varying data quantities among clients.

Li et al. [28] presents a novel FL method for identifying cyber threats in CPSs. It combines a DL-based intrusion detection model with an FL framework to enable collaborative model building. It demonstrates high efficacy in detecting various threats while preserving privacy and outperforming existing approaches. However, it should be noted that the scheme utilizes Paillier-based encryption, which can introduce computational overhead and reduce speed.

Verma et al. [34] introduced the FL-enabled Deep Intrusion Detection (FLDID) framework for smart manufacturing industries. It addresses the challenge of detecting cyber threats in complex and heterogeneous environments through collaborative model building. Privacy preservation is achieved using Paillier-based encryption, and the framework utilizes a hybrid model combining CNN, LSTM, & Multi-Layer Perceptron (MLP). Extensive experiments validate the effectiveness of FLDID in detecting cyber threats compared to existing approaches. However, it should be considered that

the use of Paillier encryption may introduce computational overhead and reduce speed.

Ibrahem et al. [35] proposes a novel, privacy-preserving, and decentralized aggregation scheme for detecting energy theft in smart grid Advanced Metering Infrastructure (AMI) networks using FL. The approach leverages functional encryption to Enable Detection Stations (ETDSs) to securely send encrypted local model parameters to an aggregator. Without requiring a trusted Key Distribution Center (KDC), the aggregator computes aggregated parameters and updates the global model without learning individual data, preserving customer privacy. Extensive experiments demonstrate that the proposed scheme achieves high detection accuracy with low computational and communication overhead. Another paper by Ibrahem et al. [36] introduces FedSafe, a decentralized and efficient FL framework that enhances privacy using functional encryption (FE) without relying on a trusted KDC. FedSafe allows participants to share encrypted model parameters with an aggregator, enabling secure global model training without revealing local data. It overcomes the limitations of existing homomorphic encryption and secure multi-party computation methods, which suffer from high overhead. Experimental results on real-world data show that FedSafe achieves superior privacy, security, scalability, and performance compared to existing FL privacy-preserving schemes.

Aouedi et al. [37] proposes F-BIDS, a Federated Blending model-driven IDS for IoT & Industrial IoT. With the surge in network traffic and concerns over user privacy, FL is introduced to enhance attack detection, privacy preservation, and cost reduction. The F-BIDS framework utilizes Decision Tree and RF as its base classifiers for generating meta-data, which is used by a Neural Networks meta-classifier during federated training. It describes how inference attacks could be avoided by modifying the data and using that to train in a federated way. This way, no attacker would be able to access the original data, thus maintaining privacy.

Zhang et al. [38] proposed a similar data-changing methodology by introducing dummy input to the original model thus providing privacy. It addresses privacy concerns in FL, where gradient-based inversion attacks can compromise client data. Existing perturbation-based privacy methods suffer utility loss. The proposed SLGP and RLGP pruning-based defense mechanisms protect privacy efficiently, maintaining model utility.

III. THEORETICAL BACKGROUND

A. AUTOENCODERS (AE)

An AE is an unsupervised neural network used in making a tuple from its latent space. It basically transforms the data from one form to another. Firstly, a tuple is used as input $v^{(i)} \in \{0, 1\}^k$, which refers to latent space as $l^{(i)} \in [0, 1]^k$. The encoding function, demonstrated by $p = \{\omega, b\}$, is defined by Eq. (1)

$$l^{(i)} = fp(v^{(i)}) = \sigma(\omega v^{(i)} + b) \tag{1}$$

where σ is a nonlinear activation function such like sigmoid or tanh, b is a bias vector, and ω is a k*k weight matrix. The latent representation $l^{(i)} \in [0, 1]^k$ is then mapped back onto a reconstructed tuple $v^{(i)} \in [0, 1]^k$, in input space (2)

$$\bar{v}^{(i)} = gu(t^{(i)}) = \sigma(\omega t^{(i)} + b) \tag{2}$$

ω of the reverse mapping may be constrained by $\omega = \omega T$,indicating tied weights [39]. The main aim to train is to understand the parameters $u = \{\omega, b\}$, & $z = \{\omega, b\}$ for reducing average reconstruction error over a set of input tuples, given by $v^{(1)}, v^{(2)}, \ldots\ldots\ldots, v^{(s)}$, Eq. (3) & Eq. (4)

$$(u.\bar{z}) = argmmax_{u.\bar{z}} \frac{1}{s} \sum_{i=1}^{s} \xi(v^{(i)}.\bar{v}^{(i)}) \tag{3}$$

$$(u.\bar{z}) = argmmin_{u.\bar{z}} \frac{1}{s} \sum_{i=0}^{s} \xi(v^{(i)}.g\bar{z}(fu(v^{(i)})) \tag{4}$$

where ξ is the loss function such that cross-entropy parameters u and z can be optimized by stochastic or mini-batch gradient descent.

B. FedAvg

FedAvg, short for federated averaging, is an FL algorithm that updates a global model by aggregated local model weights. FedAvg is performed on the cloud server described in Eq. (5)

$$\omega_i = \frac{1}{C} \sum_{c=1}^{C} \omega_{i-1}^{c} \tag{5}$$

IV. METHODOLOGY

A. ARCHITECTURAL OVERVIEW

This section presents the overview of the proposed approach for intrusion detection in industrial scenarios. It consists of various components that collaboratively execute to bring the PPFL-DCS architecture into work as shown in Fig. 1. The proposed approach enables different industries to train a collaborative model for intrusion detection without sharing their personal data.

The proposed approach consists of two main entities i.e. the clients and the cloud server. A client is a data owner (or individual industry) involved in a federated process. It has its own local model, which is trained individually on its private data. It helps in improving the global model, present at the cloud server, by sending its validation accuracy and local model gradients for updating the global model gradients in weighted order. On the other hand, the cloud server takes on the responsibility of constructing a comprehensive FL-enabled IDS through the participation of each industrial client. Multiple rounds of communication in the cloud server & the agents are needed to achieve optimal IDS.

B. MODEL ASSUMPTIONS

- Data privacy: It assumes that each participant (client) in the federated model is committed to preserving the privacy & confidentiality of their local data during the training.

TABLE 1. Notations used.

Meanings	Notation	Meanings	Notation	Meanings	Notation
Neural Transformer System (NTS)	$\Im$	Base AE encoding function	e	Learning rate for client c	η
Local model	ℓ	Base AE decoding function	d	Exponential decay rates	$\oint_1$ & $\oint_2$
Training dataset for client c	$D_{tr}^{(c)}$	Total communication rounds	R	Small constant for numerical stabilization	$\prec$
Global model	$\eth$	Total no. of clients	C	Loss function for local model	ξ
Weighted aggregator	ϖ	List of scaling factor	α_i	Batch	*kappa*
Communication round	i	List of scaling factor for client c	$\alpha_i^{(c)}$	Gradient	δ
Learning rate for NTS	lr	Reward/Penalty factor	β	Size of dataset of client c	s_c
Normalized client data	$X\_normalized^{(c)}$	Scaled weights at i^{th} round	$SW^{(i)}$		

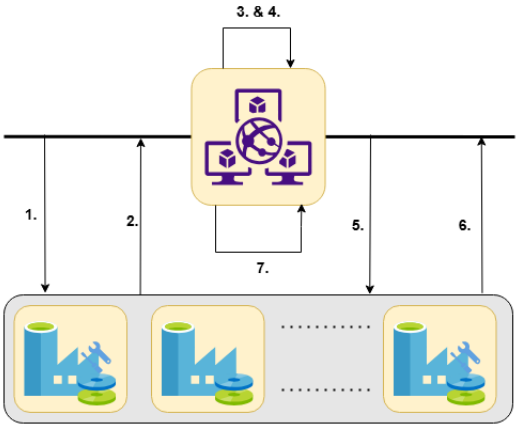

FIGURE 2. Dynamic client selection workflow in proposed framework ((1) Server shares initial model for training, (2) Client trains and returns the model weights, accuracy, and data size, (3)-(4) Weighted aggregator scales and aggregates the weights and saves the performance in storage for client selection, (5) Weight distributor sends the updated model weights, (6) Client trains on new model and sends the new weights, accuracy and data size, (7) Weighted aggregator selects the client with improvement and depreciation and scale weights accordingly. Next, steps (3)-(7) are repeated for further rounds).

- Data heterogeneity: It is assumed that the participating clients have different and diverse datasets that collectively represent a comprehensive view of the problem domain. This diversity helps in capturing a wide range of patterns and improving the overall model's performance
- Computational capacity: Each participating client has sufficient computational resources for performing local model training and contributing in FL process without significant bottlenecks or performance limitations
- Communication reliability: It is assumed that there is reliable and stable communication infrastructure between the clients and the central server to exchange model updates and parameters securely and efficiently
- Trustworthy collaboration: All participants in the federated model are trustworthy and adhere to the agreed-upon protocols and rules of collaboration, ensuring the integrity of the overall training process

C. WORKFLOW

The workflow of the system is described using Algo. 1 and Fig. 2. Initially, every client passes their data through the NTS to change the data representation and additionally do the data pre-processing. After passing the data from the NTS every client is ready to take part in the federated process. Communication round 1 begins with each client training their local model (Algo. 2) on neural transformed data and thereafter sending the model weights, the validation accuracy, and the size_of_the dataset to the cloud server. The cloud server next stores the model's current performance value and compares it with the performance value of the previous round (i!=1), selects the clients with increased performance, and increases client's scaling factor. Moreover, it also selects the client's with decreased performance and decreases their scaling factors, $\alpha_c^{(i)}$. Next, the weighted aggregator ϖ aggregates received local model weights according to Eq. (6) & (7). Further, the weight distributor distributes the weights to clients for training. These steps continue until the total communication round ends.

D. PPFL-DCS COMPONENTS

PPFL-DCS is a weighted FL framework designed to revolutionize the idea of privacy and security. It aims to provide security and privacy for a federated framework without involving higher computational expenses. It also aims to include the client's contribution to the global model by assigning weights to them in accordance with their validation accuracy and data size. This in turn helps to reduce the impact of poorly performing models and class-imbalance problems within the clients. The components of the proposed PPFL-DCS framework are the client manager, weight distributor, server, storage, DL model, NTS, and weighted aggregator.

- The client manager helps to manage clients. It is responsible for the initial registration and enrollment of clients into the FL system. This involves authenticating clients, verifying their eligibility, and collecting necessary information such as client capabilities, computational resources, and available datasets.
- The weight distributor has a pivotal role in the FL process of distributing the aggregated weights to clients after the aggregation phase. The weights are typically shared through secure and privacy-preserving methods to ensure the confidentiality of the model parameters.

Algorithm 1 PPFL-DCS Framework

Input: Client: Training dataset $D_{tr}^{(c)} = \{X_i, y_i\}, (X_i \in X, y_i \in y)$, Neural Transformer $\Im$, Local model ℓ.
Server: Global Model $\eth$, Weighted aggregator ϖ
Output: Comprehensive PPFL-DCS framework
Phase 1: Neural transformer meta-data generation
Initialize $\Im$ with batch size 16, input_dim, lr, encoding function e, decoding function d, no. of epochs for client $c^{(i)}$
Normalize the data:
$X\_normalized = \frac{X^{(c)} - \min(X^{(c)})}{\max(X^{(c)}) - \min(X^{(c)})}$.
for j in range(no. of stacked layers) **do**
    Train the AE ϱ:
    $\Im(X\_normalized^{(j)}, d^{(j)}(e^{(j)}(X\_normalized^{(j)})))$
    $X\_normalized^{(j)}$ = pd.concat ($[X\_normalized^{(j)}, \Im.predict(X\_normalized^{(j)})]$)
end for
Phase 2: Weighted federated learning
for i in R **do**
    At client
    for c in C **do**
        Train the local DL model
        Send the weights, validation_accuracy, & size of data to server
        $\varpi.receive(\ell_c.get\_weights(), val\_acc, s_c)$
    end for
    At server
    total_dataset_size = sum(size of $data_c$)
    if $i == 0$ **then**
        $\alpha_i = \frac{1}{C}, \beta = 0.1, val\_acc_{prev} = val\_acc_{curr}$
    else
        for c in C **do**
            if $val\_acc_{curr} \geq val\_acc_{prev}$ **then**
                $\alpha_i^{(c)} = \alpha_i^{(c)} + (\alpha_i^{(c)} * \beta)$
            else if $val\_acc_{curr} \leq val\_acc_{prev}$ **then**
                $\alpha_i^{(c)} = \alpha_i^{(c)} - (\alpha_i^{(c)} * \beta)$
            end if
        end for
    end if
    $\alpha_i = \frac{\alpha_i}{sum(\alpha_i)}$
    At weighted aggregator
    for c in C **do**
        $SW^{(i)}(W_c, s_c, total\_dataset\_size, \alpha_i^{(c)})$
    end for
    Aggregate the weights & distribute
    $SW_{new}^{(i)} = \varpi(SW^{(i)})$
    $SW_{new}^{(i)}.distribute()$
    $\eth.set\_weights^{(i)} = SW_{new}$
    At client
    Update the local model weights
    $\ell.set\_weights(SW_{new}^{(i)})$
    $i \leftarrow i + 1$
end for

- Cloud server acts as a coordinating entity that facilitates communication and coordination between the clients and orchestrates the training and aggregation of models. is responsible for initializing the initial global model or providing a starting point for the federated training process and act as a coordination entity.
- Storage (existing at cloud server) is used to save the performance (validation accuracy), data size, and gradients of the global model received from the participating clients during the training iterations. These gradients are updated global model parameters in accordance with the local training performed by each client. It is utilized to store the validation accuracy of each client's locally trained model. This info is helpful to evaluate and compare the performance of the individual clients' models and could aid in the selection of models for aggregation or further training.
- The model used for local and global training is designed using DL techniques. In the proposed approach, the DL model used is a combination of CNNs, LSTMs, and MLP networks. The architecture is tailored to the specific problem domain, attending to the nature of the data & the desired output.
- The weighted aggregator in PPFL-DCS assigns appropriate weights to each client's model based on two primary factors: validation accuracy and local data size. Clients whose models achieve higher validation accuracy are considered to have learned more meaningful and generalizable patterns, and are thus assigned greater weight. Additionally, clients with larger local datasets contribute more representative information, and this is also reflected in the weighting. By combining these two factors, the aggregator ensures that clients with both high-quality performance and substantial data have a stronger influence on the global model. The final aggregation is performed by computing a weighted average of the model parameters (or gradients), where the assigned weights guide the influence of each client's contribution. This strategy helps to improve the robustness and effectiveness of the global model, particularly in heterogeneous data scenarios, as described in Eq. (6) & (7)

$$\omega_{size\_scaled} = [\omega_c . \frac{s_c}{\sum(s_c)}]_{c=1}^{C} \tag{6}$$

$$SW_i = \frac{1}{C} \sum_{c=1}^{C} \alpha_i^{(c)} . \omega_{size\_scaled}^{(c)} \tag{7}$$

E. NEURAL TRANSFORMER SYSTEM (NTS)

The NTS component is designed as a critical preprocessing unit within the proposed architecture, primarily aimed at protecting sensitive input data while retaining its utility for downstream FL tasks. At the core of NTS lies the use of Stacked AutoEncoders (SAE), a deep neural architecture

known for its ability to perform unsupervised feature learning and dimensionality reduction. The SAE consists of multiple layers of autoencoders stacked sequentially, where each layer learns to encode the output of the previous one into a compressed latent representation. During the training phase, the encoder-decoder architecture is optimized to minimize reconstruction error, thereby ensuring that the learned latent features effectively capture the underlying data patterns.

The output from the final encoder layer of the SAE is treated as a transformed representation or metadata of the original input. This transformation process serves a dual purpose: it obfuscates the original form of the input, thus acting as a privacy-preserving shield, and simultaneously enhances feature abstraction for improved learning. Importantly, this transformed data cannot be easily reverted to its original form, which helps mitigate the risk of MIAs and other privacy threats commonly associated with FL systems.

The conceptual foundation of NTS is inspired by the work of Zhang et al. [38], where the authors employed a RF model to generate metadata that could be used in an FL framework to defend against MIAs. Building on this idea, our approach replaces the traditional model with a neural mechanism, SAE, that is more adaptive and capable of capturing non-linear correlations within high-dimensional data. By placing the SAE-based NTS module at the initial stage of the data processing pipeline, the FL system benefits from both robust feature extraction and enhanced privacy guarantees.

Figure 3 provides a schematic illustration of the SAE architecture used in the NTS module. It includes multiple encoding and decoding layers, with the encoded latent space acting as the transformed output. This output is then fed into the FL pipeline for subsequent training or inference tasks. By decoupling the raw input from the learning model, NTS ensures that sensitive attributes remain concealed, thereby aligning with the privacy-preserving goals of the framework.

The SAE is a semi-supervised model utilizing AEs interconnected in a layered fashion [40]. The h^{th} layer AE is exemplified. In Eq. (8), the output o_h of the hidden layer is obtained by mapping the input data $o_h - 1$, representing the encoding method. Subsequently, the decoding method in Eq. (9) generates a reconstructed feature. The reconstruction error, expressed in Eq. (10), is leveraged to minimize the disparity between the output and input, enhancing the SAE's performance.

$$o_h = \sigma(\omega_{h,e} o_{h-1}, b_{h,e}) \tag{8}$$

$$\overline{o}_{h-1} = \sigma(\omega_{h,d} o_h, b_{h,d}) \tag{9}$$

$$\xi = \sum_{j=1}^{N} \frac{||\overline{o}_{h-1}^{j} - o_{h-1}^{j}||^2}{2N} \tag{10}$$

where $\omega_{h,e}$ and $b_{h,e}$ represent the weight matrix and bias matrix of the encoder respectively. $\omega_{h,d}$ & $b_{h,d}$ denotes weights and bias matrix of the decoder, respectively, and σ denotes the activation function.

To prevent overfitting, the traditional method incorporates a sparse term KL divergence into the loss function [41]. The specific expressions for this regularization are depicted in Eq. (11) and (12).

$$KL(\psi||\overline{\psi}) = \psi log\frac{\psi}{\overline{\psi}_h} + (1-\psi)log\frac{(1-\psi)}{(1-\overline{\psi}_h)} \tag{11}$$

$$\overline{\psi}_h = \frac{1}{N}\sum_{j=1}^{N} o_h(x^j) \tag{12}$$

where $\overline{\psi}_h$ represents the average activation value of h^{th} hidden layer, and ψ is sparse parameter. Considering a layer of AE as an anecdote, the parameter $\omega_{h,e}$ could be calculated using below:

$$\frac{\partial(\xi)}{\partial(\omega_{h,e})} = \frac{\partial(\xi)}{\partial(\sigma(\omega_{h,d}o_h + b_{h,d})} * \frac{\partial(\sigma(\omega_{h,d}+b_{h,d})}{\partial(\omega_{h,d}o_h + b_{h,d})} * \frac{\partial(\omega_{h,d}o_h + b_{h,d})}{\partial(\sigma(\omega_{h,d}x + b_{h,e}} * \frac{\partial(\sigma(\omega_{h,e}x + b_{h,e})}{\partial(\omega_{h,e}x + b_{h,e})} * \frac{\partial(\omega_{h,e}x + b_{h,e})}{\partial(\omega_{h,e})}$$

Algorithm 2 Local Model Training

Input: η, $\oint_1$, $\oint_2$, $\prec$, ξ, κ, $\omega_{i-1}^{(c)}$, $c, {}_{tr}^{(c)}$
Output: Trained ℓ_c

Initialize
1: if (i == 1):
Initialize ω using suitable initialization methods
2: Initialize first & second moment variables p = 0 & q = 0 respectively
3: Create batches of $D_{tr}^{(c)}$ of batch size size_of(κ)
4: Set the model parameters $\omega_c^{(j)} \leftarrow \omega_c^{(j-1)}$
where j= round
Train
5: do{
6: for b in κ
Compute $\delta \leftarrow \Delta_{\omega_c^{(j)}}\xi$
Update biased first & second moment estimates respectively
$p \leftarrow \oint_1 .p + (1 - \oint_1)\delta$
$q \leftarrow \oint_2 .q + (1 - \oint_2)\delta^2$
Compute the bias-corrected first & second moment respectively
$\overline{p} \leftarrow \frac{p}{1-\oint_1^e}$
$\overline{q} \leftarrow \frac{q}{1-\oint_2^e}$
Update model parameters
$\omega_c^{(j)} \leftarrow \omega_c^{(j)} - \eta\frac{\overline{p}}{\sqrt{\overline{q}+\prec}}$
end
}while(until ξ converges)
7: return trained ℓ_c

The above observation indicates that as the layers increase, updated values will decay exponentially when several gradients, each less than 1, are compounded [42]. The sigmoid is

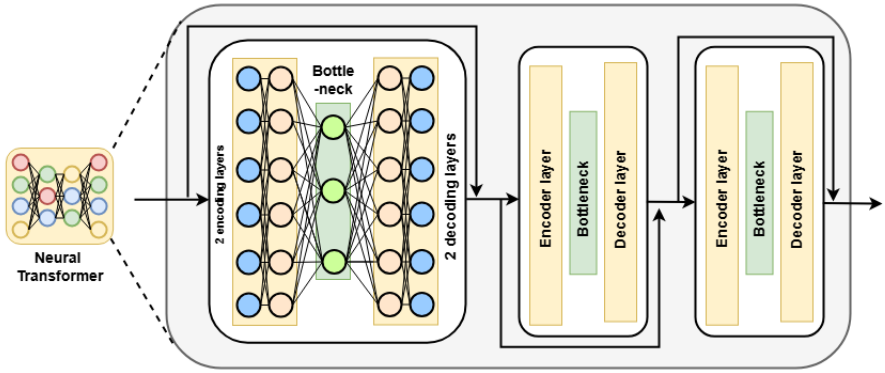

FIGURE 3. Architecture of neural transformer system.

TABLE 2. Training data distribution of clients.

Client	2 Clients	5 Clients	10 Clients	15 Clients	20 Clients
1	63204	71248	58607	16088	98253
2	5745	25281	12640	92699	57465
3	-	94231	1149	54393	40795
4	-	48264	81590	62054	35049
5	-	2298	47115	69716	6320
6	-	-	93081	39071	92507
7	-	-	35623	8427	17811
8	-	-	70098	31410	86761
9	-	-	104573	766	81015
10	-	-	24132	46732	46540
11	-	-	-	100360	52286
12	-	-	-	85038	63778
13	-	-	-	76613	29303
14	-	-	-	23749	12066
15	-	-	-	77377	75269
16	-	-	-	-	574
17	-	-	-	-	23557
18	-	-	-	-	103998
19	-	-	-	-	69524
20	-	-	-	-	58032

TABLE 3. DL model description.

Layer	Description
Input	Input(shape=(None, shape, 1)
Conv1D	TimeDistributed(Conv1D(128,3, activation='relu'))
BatchNormalization	TimeDistributed(BatchNormalization())
MaxPool1D	TimeDistributed(MaxPool1D(2,2))
Conv1D	TimeDistributed(Conv1D(64, 3, activation='relu'))
BatchNormalization	TimeDistributed(BatchNormalization())
MaxPool1D	TimeDistributed(MaxPool1D(2, 2))
Flatten	TimeDistributed(Flatten())
LSTM	LSTM(128, activation='tanh', return sequences=True)
LSTM	LSTM(128, activation='tanh')
Dense	Dense(50, activation='tanh')
Dense	Dense(100, activation='tanh')
Dropout	Dropout(rate=0.2)
Dense	Dense(1, activation='sigmoid')

a typical saturation activation function, having a maximum derivative value of 0.25. Consequently, as number of layers grows, the gradient of $\omega_{h,e}$ approaches 0, leading to a halt in network parameter updates. This vanishing gradient problem results in limited classification performance as gradients diminish after successive derivatives, causing the loss to plateau during training without further reduction.

V. EXPERIMENTAL EVALUATION

A. EXPERIMENTAL SETUP

The proposed PPFL-DCS framework was developed in Python using TensorFlow 2.13, and the global model was designed using the Keras 2.13 API. The implementation and evaluation of the proposed framework were conducted using Python 3.10.11 on a MacBook Pro equipped with an Apple M1 Pro chip, featuring an 8-core CPU and an 8-core GPU, a 16GB-core Neural engine, and a 1TB SSD. Additionally, the FL model is run with a setup of lr as 0.01, Momentum (0.9), decay (0.01), and loss function (binary cross-entropy) with XIIoTID dataset [43]. Table 2 describes the dataset distribution for multiple client systems.

B. SOLUTION EVALUATION CONSIDERATION

To validate our solution, we conducted tests on an IDS, with a primary focus on network intrusion detection. The evaluation compares the performance of our PPFL-DCS against state-of-the-art techniques and various proposed scenarios. The aim is to assess the efficacy of our approach to identify and detect network intrusions and minimize the impact of MIA, ultimately contributing to the improvement of intrusion detection capabilities in the case of multiple models with variable performance.

Table 3 describes the hybrid DL global model architecture design. These parameters were chosen by using multiple random testing values and selecting the best out of all, giving the highest global model accuracy. The approach employed in this study utilizes a combination of CNNs & LSTMs to formulate a global intrusion detection model. The model consists of a CNN unit that generates features, which are then fed sequentially into an LSTM layer to capture time series patterns over different time steps. By leveraging the temporal patterns in past connection records, the present network connection record can be categorized effectively.

Finally, an MLP layer is employed to obtain the desired output.

C. PERFORMANCE METRICS

The performance is measured upon various factors like accuracy, precision, recall, and F1-score. Here:
p: Number of attack requests correctly identified as attack
q: Number of normal samples correctly identified as normal
r: Number of normal requests incorrectly identified as attack
s: Number of attack requests incorrectly identified as normal

$$Accuracy = \frac{(p+q)}{(p+q+r+s)} \tag{13}$$

$$Recall = \frac{p}{(p+s)} \tag{14}$$

$$Precision = \frac{p}{(p+r)} \tag{15}$$

$$F1-score = \frac{2p^2}{2p^2+pr+ps} \tag{16}$$

D. RESULTS ANALYSIS

This section analyzes and compares the efficacy of the PPFL-DCS framework in different scenarios using metrics such as accuracy, precision, recall,f1-score, AUC, and loss [2].

1) NTS ANALYSIS

Table 4 describes the NTS with various configurations & their performances. The models are evaluated using metrics like learning rate, regularization method, batch size, and accuracy.

It consists of five unique models, each described by its architectural features. The first model, with "2 encoder layers + 1 bottleneck + 2 decoder layers," serves as the baseline, having the highest accuracy of 87.9%. It employs a learning rate of 0.001 and applies L1 regularization with a batch size of 16. Results show that models with more encoder layers perform relatively poorer than the baseline. L1 regularization boosts performance compared to L2. Batch sizes and learning rates also influence accuracy.

TABLE 4. Comparison of SAEs in NTS system.

Description	LR	Regularization	Batch size	Accuracy
2 encoder layers + 1 bottleneck + 2 decoder layers	**0.001**	**l1**	**16**	**87.900**
3 encoder layers + 1 bottleneck + 2 decoder layers	0.001	l2	32	74.910
4 encoder layers + 1 bottleneck + 2 decoder layers	0.001	l1	32	73.680
2 encoder layers + 1 bottleneck + 2 decoder layers	0.010	l2	32	73.611
2 encoder layers + 1 bottleneck + 2 decoder layers	0.001	l2	16	80.792

2) PPFL-DCS MODEL ANALYSIS

This section presents the overall analysis of the proposed PPFL-DCS model assessed for different clients (2, 5, 10, 15, 20) over a range of communication rounds (2, 5, 10, 15, 20, 25, 30) as described in Table 5.

It is observed that with 2 clients as the communication rounds increase, the accuracy steadily improves, reaching 97.1930% in just 2 rounds and 97.5170% after 25 rounds. Similarly, the loss decreases from 0.5286 to 0.5207 over the same rounds. The F1-score, precision, recall, and AUC also exhibit similar trends, showing consistent improvement as the communication rounds progress. Analysis of subsequent sections of the table with 5, 10, 15, and 20 clients demonstrate that the metrics tend to improve as more communication rounds are executed, indicating that additional rounds contribute to better convergence and model refinement. The table's detailed breakdown allows for a granular understanding of how different client configurations and communication rounds impact the model's performance.

This table also aims to provide details of the time requirements of FL process under different configurations, considering the number of clients and communication rounds. It is observed that the total time increases as the communication rounds progress. For instance, with just 2 communication rounds, the total time taken is 246.2918 seconds, while with 30 communication rounds, the total time extends to 3050.4598 seconds. This indicates that the FL process requires more time to converge and refine the model as the rounds increase. A similar trend of increasing time with more communication rounds is observed in This observation is in line with the distributed nature of FL, where multiple clients participate in training, leading to increased communication overhead and computation time.

3) IMPACT OF EPOCHS PER CLIENT ON GLOBAL MODEL

This section presents a comparison of model performance in an FL scenario with 2 clients with varying numbers of epochs. Table 6 evaluates different combinations of epochs per client and communication rounds to study their impact on various performance metrics indicated in it and the time taken for various epochs in Figure 4. The experiment indicated that the proposed model achieves high accuracy across all scenarios, ranging from 97.3% to 97.3810%. The loss values are relatively low, with the minimum being 0.5217 whereas

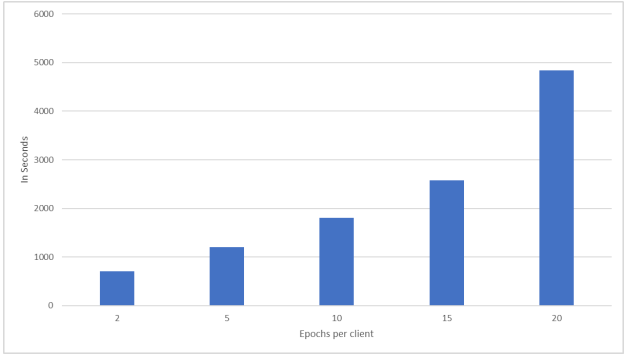

FIGURE 4. Time consumption of epochs per client in PPFL-DCS model for 2 clients.

TABLE 5. PPFL-DCS model analysis using various parameters.

Clients	Communication Rounds	Accuracy	Loss	F1-score	Precision	recall	auc	Total Time taken for federated (sec)
2	2	97.193	0.529	97.073	98.536	95.652	99.424	246.292
	5	97.360	0.525	97.253	98.522	96.016	99.475	624.506
	10	97.479	0.523	97.382	98.420	96.365	99.504	1233.542
	15	97.509	0.522	97.414	98.417	96.432	**99.505**	1809.981
	20	97.331	0.523	97.254	97.391	**97.116**	99.502	2247.805
	25	**97.517**	**0.521**	**97.428**	**98.192**	96.675	99.491	2661.847
	30	97.454	0.521	97.367	98.001	96.742	99.494	3050.460
5	2	96.798	0.528	96.627	**99.118**	94.258	99.266	211.061
	5	97.399	0.526	97.280	99.018	95.603	99.480	524.004
	10	97.681	0.525	97.585	98.915	96.290	99.568	1041.039
	15	97.760	0.524	97.671	98.868	96.503	99.612	1551.777
	20	97.783	0.524	97.700	98.645	96.773	99.631	2058.976
	25	97.815	0.524	97.733	98.700	96.784	99.639	2568.053
	30	**97.865**	**0.523**	**97.784**	98.767	**96.820**	**99.657**	3077.185
10	2	96.812	0.529	96.65	98.893	94.505	99.222	351.609
	5	97.378	0.527	97.261	98.908	95.668	99.453	867.196
	10	97.601	0.525	97.500	98.904	96.136	99.545	1713.124
	15	97.696	0.525	97.600	98.957	96.279	99.582	2570.165
	20	97.757	0.530	97.664	**98.998**	96.365	99.603	3342.217
	25	97.794	0.524	97.705	98.913	96.527	99.620	3987.541
	30	**97.832**	**0.524**	**97.746**	98.950	**96.570**	**99.630**	4631.299
15	2	96.888	0.529	96.735	98.890	94.746	99.281	1133.467
	5	97.189	0.528	97.060	98.820	95.362	99..404	2469.432
	10	97.354	0.527	97.236	98.854	95.667	99.479	4273.388
	15	97.456	0.527	97.348	98.795	95.942	99.510	5508.367
	20	97.516	0.526	97.413	98.763	96.096	99.534	6704.167
	25	97.563	0.526	97.460	**98.902**	96.059	99.549	7908.346
	30	**97.608**	**0.526**	**97.508**	98.849	**96.203**	**99.560**	9081.564
20	2	96.670	0.533	96.504	98.656	94.443	99.096	1638.945
	5	97.032	0.529	96.888	**98.909**	94.947	99.315	3372.377
	10	97.261	0.528	97.138	98.832	95.501	99.416	6235.911
	15	97.352	**0.527**	97.233	98.896	95.626	**99.458**	9265.926
	20	**97.440**	0.527	**97.331**	98.788	**95.917**	**99.483**	12196.590

TABLE 6. Performance evaluation of PPFL-DCS with multiple epochs per client using 2 clients.

Metric (2 clients 10 rounds)	Number of Epochs			
	2	5	10	15
Accuracy	97.376	97.381	97.314	97.300
Loss	0.523	0.521	0.522	0.522
Fscore	97.288	97.289	97.221	97.215
Precision	97.855	98.014	97.870	97.586
Recall	96.728	96.575	96.581	96.846
AUC	99.486	99.495	99.480	99.468

TABLE 7. Comparison of PPFL-DCS with existing state-of-the-art techniques.

Metric	Proposed	WEIGHTED [44]	DEEPFED [28]
Accuracy	97.601	92.270	95.976
Loss	0.525	0.566	0.540
F1-score	97.500	91.841	5.758
Precision	98.904	94.417	98.296
Recall	96.136	89.402	93.348
AUC	99.545	96.253	98.633

F1-score ranges from 97.2148 to 97.2891 and the precision ranges from 97.5861% to 98.0142% with recall varying from 96.5746% to 96.8462%. AUC, which represents the area under the receiver operating characteristic (ROC) curve, is exceptionally high for all scenarios, with values ranging from 99.4675% to 99.4948%. It highlights that increasing epochs per client does not necessarily leads significant improvements in the performance metrics.

4) COMPARISON OF PROPOSED PPFL-DCS WITH OTHER DL TECHNIQUES IN FL SCENARIO

Figure 5 presents a comprehensive comparison of the PPFL-DCS with several other DL models used in FL-based scenarios along with Figure 6 describing the loss metrics of each model. The models used for comparison in the FL scenario are MLP, CNN, GRU, GRU + MLP, CNN + MLP, and CNN GRU + MLP (Parallel Combination of Convolutional Neural Network and Gated Recurrent Unit with Multi-Layer Perceptron). Results indicate that the proposed work outperforms all other models across all performance metrics. It achieves the highest accuracy of 97.424%, indicating the model's ability to make accurate predictions. The loss value is significantly low at 0.525667, demonstrating the model's efficiency in minimizing errors during training and validation. The precision of the proposed work is also the highest at 98.8014% with F1-score at 97.31357% showcasing the model's balanced performance in terms of true positives and false positives. Moreover, the

recall of the proposed work is commendable at 95.8698%, indicating its ability to correctly identify positive instances from the actual ones. The AUC value is the highest among all models at 99.38936%. In comparison, the other models, including MLP, CNN, GRU, and their various combinations, achieve relatively lower performance across the metrics. While some models like CNN and CNN + MLP demonstrate reasonable performance, they still fall short compared to the proposed work.

It establishes the superiority of the proposed solution design for intrusion detection. Its exceptional performance positions it as a promising choice for enhancing cybersecurity in smart manufacturing systems and other industrial applications. The proposed model's ability to achieve high accuracy, precision, recall, and AUC makes it a robust solution for detecting cyber threats and ensuring the security of critical systems within the industry.

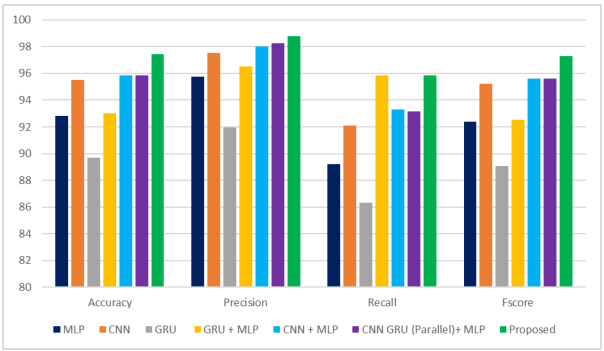

FIGURE 5. Comparison of proposed PPFL-DCS with other DL techniques in FL scenario.

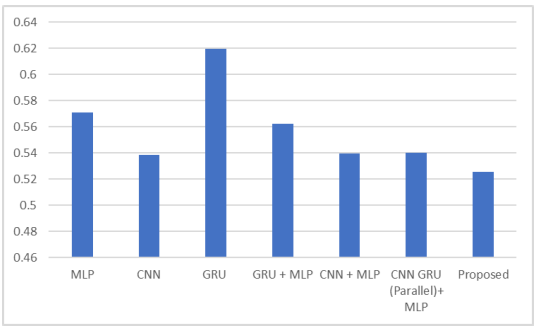

FIGURE 6. Loss comparison of proposed PPFL-DCS with other DL techniques in FL scenario.

5) COMPARISON OF PROPOSED PPFL-DCS WITH EXISTING STATE-OF-THE-ART TECHNIQUES

Table 7 presents a comparative analysis of PPFL-DCS with two other techniques, namely WEIGHTED [44] and DEEPFED [28]. The proposed technique achieves the highest accuracy of 97.6010%, showcasing its ability to make accurate predictions in intrusion detection. Moreover, the precision of the proposed technique is remarkably high at 98.9036%, implying its ability to correctly identify positive instances from the total instances predicted as positive. The recall, or sensitivity, is also excellent at 96.1361%, indicating the model's ability to correctly identify positive instances from the actual instances. In comparison, the other techniques, WEIGHTED and DEEPFED, exhibit lower performance across various metrics. WEIGHTED achieves an accuracy of 92.2700%, while DEEPFED attains an accuracy of 95.9760%. Both techniques have higher loss values, indicating less efficient error minimization during training. Furthermore, DEEPFED's F1-score of 5.7580% raises concerns about its ability to maintain a balance between precision and recall. This suggests that DEEPFED may be biased towards certain classes and might struggle with distinguishing between positive and negative instances effectively. The precision and recall values for WEIGHTED and DEEPFED are also lower compared to the proposed technique, indicating their limitations in correctly identifying positive instances and their sensitivity to true positive rates. Overall, the results from Table 7 emphasize the superiority of the PPFL-DCS for intrusion detection over WEIGHTED and DEEPFED. The proposed technique's exceptional performance in terms of accuracy, precision, recall, and AUC makes it a promising and reliable choice for detecting cyber threats in smart manufacturing and other critical systems making it a robust solution for ensuring the security and integrity of industrial networks and safeguarding against potential cyberattacks.

6) COMPARISON OF PROPOSED WORK WITH ML TECHNIQUES IN CENTRALIZED SCENARIOS

Figure 7 compares the accuracy of different models, including SVM, LR, KNN, and Decision Tree (DT) in centralized settings with the proposed model. Among the models, DT achieved the highest accuracy of 99.04%, followed by KNN with an accuracy of 98.47%. The SVM and LR models achieved lower accuracies of 92.27% and 91.48%, respectively. However, the proposed model achieves an accuracy of 97.42%, lower than the DT's and KNN's achieved accuracy, it still stands as a better IDS when compared in terms of privacy and data security. In comparison to the FL, the centralized setting may achieve better results but it falls

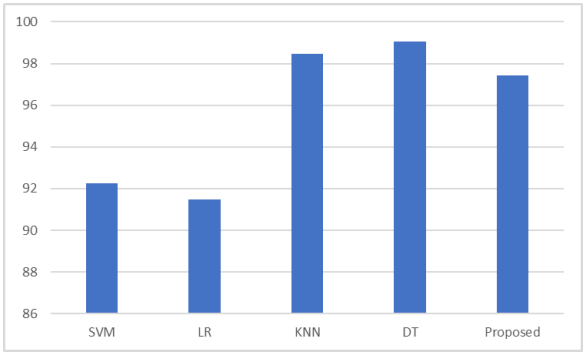

FIGURE 7. Comparison of PPFL-DCS model with ML classifiers in centralized scenario.

short of providing privacy, facing high computation overhead. Thus, PPFL-DCS could be held accountable for providing more secure IDS than any centralized setting.

VI. CONCLUSION AND FUTURE WORK

This research introduces PPFL-DCS federated deep learning scheme to address cybersecurity challenges in consumer electronics. Combining a federated weighted framework with Neural Transformer, the proposed model provides data privacy while enabling distributed collaborative model creation for industrial clients and additionally protecting them from MIAs. Extensive trials show PPFL-DCS's high accuracy 97.424% in identifying cyber threats to industrial CPSs, surpassing other techniques like WEIGHTED 92.270% and DEEPFED 95.976%. The weight enhancement technique improves security and performance by dynamically updating model weights, mitigating the risk of including deficient models. PPFL-DCS represents a significant advancement in secure FL for combating evolving cyberattacks in industries while also ensuring data privacy and accuracy. In future work, we plan to explore the integration of self-learning capabilities into the global model to enhance its robustness against poisoning and adversarial attacks. Additionally, we aim to conduct a comprehensive quantitative evaluation of the NTS to assess its impact on training time and resource usage, thereby gaining deeper insights into its computational efficiency. Furthermore, we intend to extend our security evaluation to encompass a broader range of adversarial threats, including model poisoning attacks, backdoor attacks, and inference based attacks beyond MIA, to ensure a more holistic understanding of the system's resilience.

REFERENCES

[1] A. A. Devi, E. S. Babu, R. S. Rathore, R. H. Jhaveri, and F. Benedetto, "Blockchain-based resilient pairing and bonding of BLE devices using deep reinforcement learning," *IEEE Trans. Consum. Electron.*, early access, Dec. 13, 2024, doi: 10.1109/TCE.2024.3516756.

[2] P. Verma, J. G. Breslin, D. O'Shea, N. Mehta, N. Bharot, and A. Vidyarthi, "Leveraging gametic heredity in oversampling techniques to handle class imbalance for efficient cyberthreat detection in IIoT," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 1940–1951, Feb. 2024.

[3] R. Mendes and J. P. Vilela, "Privacy-preserving data mining: Methods, metrics, and applications," *IEEE Access*, vol. 5, pp. 10562–10582, 2017.

[4] Z. Qu, J. Ding, R. H. Jhaveri, Y. Djenouri, X. Ning, and P. Tiwari, "Fed-Sarah: A novel low-latency federated learning algorithm for consumer-centric personalized recommendation systems," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 2675–2686, Feb. 2024.

[5] L. Li, Y. Fan, Y. K. Tse, and K. Lin, "A review of applications in federated learning," *Comput. Ind. Eng.*, vol. 149, Sep. 2020, Art. no. 106854.

[6] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Syst.*, vol. 216, Jan. 2021, Art. no. 106775.

[7] N. Chaurasia, M. Ram, P. Verma, N. Mehta, and N. Bharot, "A federated learning approach to network intrusion detection using residual networks in industrial IoT networks," *J. Supercomput.*, vol. 80, no. 13, pp. 18325–18346, Sep. 2024.

[8] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[9] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 3–18.

[10] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang, "Membership inference attacks on machine learning: A survey," *ACM Comput. Surveys*, vol. 54, no. 11s, pp. 1–37, Jan. 2022.

[11] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019, *arXiv:1905.10497*.

[12] M. Ge, X. Fu, N. Syed, Z. Baig, G. Teo, and A. Robles-Kelly, "Deep learning-based intrusion detection for IoT networks," in *Proc. IEEE 24th Pacific Rim Int. Symp. Dependable Comput. (PRDC)*, Dec. 2019, pp. 256–25609.

[13] L. Teng, S. Teng, F. Tang, H. Zhu, W. Zhang, D. Liu, and L. Liang, "A collaborative and adaptive intrusion detection based on SVMs and decision trees," in *Proc. IEEE Int. Conf. Data Mining Workshop*, Dec. 2014, pp. 898–905.

[14] X. K. Ren, W. B. Jiao, and D. Zhou, "Intrusion detection model of weighted navie Bayes based on particle swarm optimization algorithm," *Comput. Eng. Appl.*, vol. 52, no. 7, pp. 122–126, 2016.

[15] R.-H. Dong, X.-Y. Li, Q.-Y. Zhang, and H. Yuan, "Network intrusion detection model based on multivariate correlation analysis–long short–time memory network," *IET Inf. Secur.*, vol. 14, no. 2, pp. 166–174, Mar. 2020.

[16] F. Folino, G. Folino, M. Guarascio, F. S. Pisani, and L. Pontieri, "On learning effective ensembles of deep neural networks for intrusion detection," *Inf. Fusion*, vol. 72, pp. 48–69, Aug. 2021.

[17] M. Aamir and S. M. Ali Zaidi, "Clustering based semi-supervised machine learning for DDoS attack classification," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 33, no. 4, pp. 436–446, May 2021.

[18] K. Shiomoto, "Network intrusion detection system based on an adversarial auto-encoder with few labeled training samples," *J. Netw. Syst. Manage.*, vol. 31, no. 1, p. 5, Jan. 2023.

[19] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A. R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for iot," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 756–767.

[20] O. Friha, M. A. Ferrag, L. Shu, L. Maglaras, K.-K.-R. Choo, and M. Nafaa, "FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things," *J. Parallel Distrib. Comput.*, vol. 165, pp. 17–31, Jul. 2022.

[21] O. Aouedi, K. Piamrat, G. M´uller, and K. Singh, "Federated semisupervised learning for attack detection in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 286–295, Jan. 2023.

[22] J. Guo, Z. Liu, S. Tian, F. Huang, J. Li, X. Li, K. K. Igorevich, and J. Ma, "TFL-DT: A trust evaluation scheme for federated learning in digital twin for mobile networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 11, pp. 3548–3560, Nov. 2023.

[23] J. Guo, J. Li, Z. Liu, Y. Xiong, Y. Ma, A. V. Vasilakos, X. Li, and J. Ma, "LCEFL: A lightweight contribution evaluation approach for federated learning," *IEEE Trans. Mobile Comput.*, early access, Feb. 25, 2023, doi: 10.1109/TMC.2025.3545140.

[24] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, "Federated-learning-based anomaly detection for IoT security attacks," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2545–2554, Feb. 2022.

[25] D. C. Attota, V. Mothukuri, R. M. Parizi, and S. Pouriyeh, "An ensemble multi-view federated learning intrusion detection for IoT," *IEEE Access*, vol. 9, pp. 117734–117745, 2021.

[26] N. A. Al-Athba Al-Marri, B. S. Ciftler, and M. M. Abdallah, "Federated mimic learning for privacy preserving intrusion detection," in *Proc. IEEE Int. Black Sea Conf. Commun. Netw. (BlackSeaCom)*, May 2020, pp. 1–6.

[27] Y. Zhao, J. Chen, D. Wu, J. Teng, and S. Yu, "Multi-task network anomaly detection using federated learning," in *Proc. 10th Int. Symp. Inf. Commun. Technol.*, 2019, pp. 273–279.

[28] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "DeepFed: Federated deep learning for intrusion detection in industrial Cyber–Physical systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5615–5624, Aug. 2021.

[29] Y. Sun, H. Esaki, and H. Ochiai, "Adaptive intrusion detection in the networking of large-scale LANs with segmented federated learning," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 102–112, 2021.

[30] R. Zhao, Y. Yin, Y. Shi, and Z. Xue, "Intelligent intrusion detection based on federated learning aided long short-term memory," *Phys. Commun.*, vol. 42, Oct. 2020, Art. no. 101157.

[31] D. Man, F. Zeng, W. Yang, M. Yu, J. Lv, and Y. Wang, "Intelligent intrusion detection based on federated learning for edge-assisted Internet of Things," *Secur. Commun. Netw.*, vol. 2021, pp. 1–11, Oct. 2021.

[32] P. Verma, M. P. De Leon, J. G. Breslin, and D. O'Shea, "FedTIU: Securing virtualized PLCs against DDoS attacks using a federated learning enabled threat intelligence unit," in *Proc. IEEE Int. Conf. Smart Comput. (SMARTCOMP)*, Jun. 2023, pp. 233–236.
[33] P. Verma, N. Bharot, J. G. Breslin, D. O'Shea, A. Vidyarthi, and D. Gupta, "Zero-day guardian: A dual model enabled federated learning framework for handling Zero-day attacks in 5G enabled IIoT," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 3856–3866, Feb. 2024.
[34] P. Verma, J. G. Breslin, and D. O'Shea, "FLDID: Federated learning enabled deep intrusion detection in smart manufacturing industries," *Sensors*, vol. 22, no. 22, p. 8974, Nov. 2022.
[35] M. I. Ibrahem, M. Mahmoud, M. M. Fouda, B. M. ElHalawany, and W. Alasmary, "Privacy-preserving and efficient decentralized federated learning-based energy theft detector," in *Proc. IEEE Global Commun. Conf.*, Dec. 2022, pp. 287–292.
[36] M. I. Ibrahem, M. M. Fouda, and Z. M. Fadlullah, "FedSafe-no KDC needed: Decentralized federated learning with enhanced security and efficiency," in *Proc. IEEE 21st Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2024, pp. 969–975.
[37] O. Aouedi and K. Piamrat, "F-BIDS: Federated-blending based intrusion detection system," *Pervas. Mobile Comput.*, vol. 89, Feb. 2023, Art. no. 101750.
[38] Z. Zhang, Z. Tianqing, W. Ren, P. Xiong, and K.-K.-R. Choo, "Preserving data privacy in federated learning through large gradient pruning," *Comput. Secur.*, vol. 125, Feb. 2023, Art. no. 103039.
[39] C. G. Turhan and H. S. Bilge, "Recent trends in deep generative models: A review," in *Proc. 3rd Int. Conf. Comput. Sci. Eng. (UBMK)*, Sep. 2018, pp. 574–579.
[40] M. Cui, Y. Wang, X. Lin, and M. Zhong, "Fault diagnosis of rolling bearings based on an improved stack autoencoder and support vector machine," *IEEE Sensors J.*, vol. 21, no. 4, pp. 4927–4937, Feb. 2021.
[41] W. Li, Z. Shang, M. Gao, S. Qian, B. Zhang, and J. Zhang, "A novel deep autoencoder and hyperparametric adaptive learning for imbalance intelligent fault diagnosis of rotating machinery," *Eng. Appl. Artif. Intell.*, vol. 102, Jun. 2021, Art. no. 104279.
[42] X. Wang, Y. Qin, Y. Wang, S. Xiang, and H. Chen, "ReLTanh: An activation function with vanishing gradient resistance for SAE-based DNNs and its application to rotating machinery fault diagnosis," *Neurocomputing*, vol. 363, pp. 88–98, Oct. 2019.
[43] M. Al-Hawawreh, E. Sitnikova, and N. Aboutorab, "X-IIoTID: A connectivity-agnostic and device-agnostic intrusion data set for industrial Internet of Things," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3962–3977, Mar. 2022.
[44] A. Chaudhuri, A. Nandi, and B. Pradhan, "A dynamic weighted federated learning for Android malware classification," in *Proc. Soft Computing: Theories Appl.*, Jan. 2023, pp. 147–159.

NAKUL MEHTA received the bachelor's degree in information technology from the Dr. B. R. Ambedkar National Institute of Technology Jalandhar (NITJ), Nakul. He is currently a Research Ireland funded Ph.D. Researcher under Centre's for Research Training in Artificial Intelligence (CRT-AI) at Data Science Institute, University of Galway, Ireland. He also has experience as a Data Scientist at Aramex and also completed a research internship at the University of Galway, in the past.

NITESH BHAROT (Senior Member, IEEE) received the Ph.D. degree in cloud security from Rabindranath Tagore University (RNTU), India (a NIRF-ranked university). He is currently the Research Lead and a Senior Postdoctoral Researcher with the Insight SFI Research Centre for Data Analytics, University of Galway. He has secured numerous grants from prominent funding agencies, including the EU, GEANT, SFI, Insight, MHRD, and MPCST. Additionally, he has contributed as a technical program committee member, a speaker, and a reviewer of various IEEE conferences and journals. His research interests include cyber security, AI/ML, healthcare, and industry 5.0.

JOHN G. BRESLIN (Senior Member, IEEE) received the bachelor's degree in electronic engineering and the Ph.D. degree in electronic engineering from the University of Galway, in 1994 and 2002, respectively. He is currently a Professor in electronic engineering with the University of Galway, where he is also the Director of the TechInnovate and AgInnovate Entrepreneurship Programs. He associated with two SFI Research Centres, he is also a Co-Principal Investigator at Insight (Data Analytics) and a Funded Investigator at VistaMilk (AgTech). He has co-authored around 300 publications, including the books *The Social Semantic Web*, *Social Semantic Web Mining*, and the *Old Ireland in Colour* trilogy. He co-created the SIOC framework, implemented in hundreds of applications (by Yahoo, Boeing, and Vodafone) on at least 65 000 websites with 35 million data instances. He is also the Co-Founder of the PorterShed, boards.ie and adverts.ie.

PRIYANKA VERMA (Senior Member, IEEE) received the Ph.D. degree from the Atal Bihari Vajpayee Indian Institute of Information Technology and Management, India. She completed the Leadership and Innovation Program, Massachusetts Institute of Technology, USA. She was formerly an Assistant Professor with the Maulana Azad National Institute of Technology Bhopal. She held a Marie Skłodowska-Curie Action Fellowship. Currently, she is an Assistant Professor with the School of Computer Science, University of Galway, Ireland. She has also been a Visiting Research Scholar with Anglia Ruskin University, Chelmsford, U.K. She has co-authored numerous publications in leading journals and conferences. Her research interests include cybersecurity, the IIoT, smart manufacturing, AI/ML, cloud, and edge computing. She has received many awards at both national and international levels. She is a conference speaker and has delivered expert lectures in many countries and a reviewer of many reputed journal.