Federated Deep Learning for Intrusion Detection in IoT Networks

Othmane Belarbi
Cardiff University
Cardiff, UK
belarbio@cardiff.ac.uk

Theodoros Spyridopoulos
Cardiff University
Cardiff, UK
spyridopoulost@cardiff.ac.uk

Eirini Anthi
Cardiff University
Cardiff, UK
anthies@cardiff.ac.uk

Ioannis Mavromatis
BRIL, Toshiba Europe Ltd.
Bristol, UK
ioannis.mavromatis@toshiba-bril.com

Pietro Carnelli
BRIL, Toshiba Europe Ltd.
Bristol, UK
pietro.carnelli@toshiba-bril.com

Aftab Khan
BRIL, Toshiba Europe Ltd.
Bristol, UK
aftab.khan@toshiba-bril.com

Abstract—The vast increase of Internet of Things (IoT) technologies and the ever-evolving attack vectors have increased cyber-security risks dramatically. A common approach to implementing AI-based Intrusion Detection Systems (IDSs) in distributed IoT systems is in a centralised manner. However, this approach may violate data privacy and prohibit IDS scalability. Therefore, intrusion detection solutions in IoT ecosystems need to move towards a decentralised direction. Federated Learning (FL) has attracted significant interest in recent years due to its ability to perform collaborative learning while preserving data confidentiality and locality. Nevertheless, most FL-based IDS for IoT systems are designed under unrealistic data distribution conditions. To that end, we design an experiment representative of the real world and evaluate the performance of an FL-based IDS. For our experiments, we rely on TON-IoT, a realistic IoT network traffic dataset, associating each IP address with a single FL client. Additionally, we explore pre-training and investigate various aggregation methods to mitigate the impact of data heterogeneity. Lastly, we benchmark our approach against a centralised solution. The comparison shows that the heterogeneous nature of the data has a considerable negative impact on the model's performance when trained in a distributed manner. However, in the case of a pre-trained initial global FL model, we demonstrate a performance improvement of over 20% (F1-score) compared to a randomly initiated global model.

Index Terms—Federated Learning, Intrusion Detection System, Internet of Things, Deep Learning, Deep Belief Networks

I. INTRODUCTION

The increasing adoption of IoT devices in industrial applications and public infrastructure as well as the ever-evolving attack vectors and threat actors have increased cyber-security risks dramatically. The complexity and heterogeneity of these systems exacerbate the impact of an attack, ranging from data theft to service disruption and human injury, rendering the security of IoT devices a significant task [1].

In an attempt to enhance the security of IoT devices, several studies have focused on AI to build more sophisticated threat detection models. Traditional IDS solutions base their operations on Machine Learning (ML) models trained centrally in the Cloud and then deployed across multiple devices. Never-

theless, this centralised approach often suffers from increased network overheads and high latency [2], thereby resulting in slow detection of malicious traffic, lack of scalability and unresponsiveness to attacks in the worst case. In addition, centralised data collection can lead to data privacy and secrecy violations. These challenges urge the need to develop effective IDSs tailored to the requirements of modern IoT systems.

In 2017, Researchers at Google introduced FL, a distributed learning paradigm that preserves user privacy and data confidentiality [3]. In FL, multiple clients collaboratively train a global model under the orchestration of a central server without sharing any raw data. FL can be used to implement a distributed IDS addressing the aforementioned requirements [4]. However, the authors in [5] note that most intrusion detection research using FL utilises unrealistic data distribution among the participants and inappropriate datasets.

To address these issues, this paper presents an FL-based IDS tailored to IoT systems, considering the impact of non-Independent and Identically Distributed (non-IID) data. In particular, we use the recent TON-IoT [6] dataset and partition it according to the destination IP address in order to simulate a real-world scenario. This enables the performance evaluation of our FL-based IDS with realistic non-IID data. We implement two FL-based IDS flavours, one based on Deep Neural Networks (DNNs) and one based on our previous work on Deep Belief Networks (DBNs) and evaluate their performance. To address the impact of the non-IID dataset on the detection performance, we evaluate the effect of three aggregation methods (FedAvg, FedProx and FedYogi) and also compare the use of pre-trained models against randomly initialised models. The experimental results show that starting FL from a pre-trained model alleviates the effect of data heterogeneity and improves the stability of global aggregation in FL. In addition, FedProx increases the detection performance, followed closely by FedYogi. Nevertheless, the centralised IDS performs significantly better than the FL approaches.

The contributions of this work are summarised as follows:
  – We design realistic FL-based IDS experiments, relying

on real-world network traffic split based on the device IP address. Our set-up results in a non-IID distributed dataset that reflects the heterogeneity of IoT networks.

– We implement and evaluate the performance of a DBN-based IDS and compare it against a DNN-based IDS and a centralised IDS approach.

– We evaluate the impact of pre-training in FL on the detection performance given our realistic non-IID set-up.

– We evaluate the effect of three aggregation methods (FedAvg, FedProx and FedYogi) on the performance of FL-based IDSs in non-IID set-ups.

The remaining part of the paper proceeds as follows: In Section II, we highlight recent relevant work on FL-based IDS in IoT systems. Section III describes our methodology, including data partitioning and pre-processing. In Section IV, we report the results of our experiments. Finally, we conclude the paper and present pathways for future work in Section V.

II. RELATED WORK

In this section, we introduce the technical background related to FL and analyse the latest research on intrusion detection in IoT using FL.

A. Federated Learning for IoT systems

FL is a distributed learning paradigm where multiple clients collaboratively train a global model under the orchestration of a central server without sharing raw data with a third party [3]. Instead of exchanging their local data with the central server, the clients share only their model updates, hence preserving the data locality and privacy [3]. In principle, the training process consists of two major phases: local training and global aggregation. Initially, the clients download an initial global model from the central server. Upon retrieving it, each client trains an updated model based on their local data. Then, these selected clients send their model updates back to the central server. Finally, the central server aggregates these updates (typically by averaging) and generates a new global model. This process is repeated for multiple iterations until the desired performance is achieved. Since the raw data is not shared for the training in FL settings, the leakage of sensitive information is minimised, and data privacy/secrecy is enhanced. Also, the communication overhead is significantly reduced as there is no need for the clients to transmit their raw data to the central server over the network [3]. Hence, FL enables the deployment of IoT IDS more securely and efficiently [7].

B. FL-based IDS for IoT

In recent years, the application of FL in IoT networks has attracted increasing interest among researchers. The characteristics of FL seem to offer various benefits for IoT security systems. This section presents works that research the use of FL for the specific task of intrusion detection in IoT networks.

Various intrusion detection datasets have been used to develop FL-based IDS. The NSL-KDD dataset is used by [8], to train a multi-class classification DNN. The authors consider three scenarios with different data distributions. The

experimental results of the FL-based IDS showed that the FL approach outperforms the distributed learning setting and can reach comparable accuracy with respect to the centralised learning setting. Also, based on the NSL-KDD dataset, [4] uses Multi-Layer Perceptron (MLP) to implement an FL-based IDS. The proposed approach is based on mimic learning where a teacher model eliminates training data noise/error and soft labels are passed to the student model as regularization to avoid overfitting. In [9], the authors propose a new aggregation method named FedAGRU. The latter uses an attention mechanism to prevent the uploading of unimportant model updates that do not benefit the global model. The experiments conducted on the CICIDS2017 dataset indicate that FedAGRU has higher detection accuracy while reducing communication overhead. Finally, [10] evaluated an FL-based Cloud intrusion detection scheme using the KDDCup'99 dataset. The authors explored the utilisation of blockchain technology to secure the aggregation of the model updates. However, such an approach requires resources that are typically not available in the resource-constrained environment of an IoT system.

While the majority of the works mentioned above have been designed for IoT scenarios, a key limitation is that they are all based on intrusion datasets which do not fully reflect the IoT-specific network traffic. Moreover, the majority of the IDS datasets were not designed to be used in an FL environment. To that end, recent efforts consider IoT-specific datasets to develop FL-based IDS. For example, [11] evaluated an FL-based anomaly detection using the Modbus dataset. The authors used the Gated Recurrent Units (GRUs) models for identifying the attacks, and the FedAvg aggregation method. The evaluation results show that the suggested approach outperforms the non-FL versions in terms of accuracy in attack detection as well as privacy of user data. Furthermore, [12] presented a federated deep learning method for zero-day botnet attack detection in IoT-edge devices. The adopted method used the N-BaIoT and Bot-IoT datasets to simulate the zero-day botnet attack and the DNN model to classify the network traffic. One network traffic class was not included in each partition to reproduce a real-life scenario. The FL method was compared with the centralised, distributed and localised methods. The results demonstrated that the FL-based IDS is most efficient for the detection of zero-day attacks in IoT devices against all other methods.

However, the aforementioned studies either do not provide data distributions among the clients, or they simulate scenarios where the client's data distribution does not fluctuate, and all data samples follow the same probability distribution. In realistic FL scenarios, non-IID data is common when data is generated by the end devices. Additionally, recent studies [13], [14] have established that training on non-IID data introduces challenges that could deteriorate the performance of the IDS, and lead to convergence issues. On this question, there is a relatively small body of literature that considers the aspect of non-IID in their evaluation of FL-based IDS.

The researchers in [15] proposed LocKedge, a multi-attack detection mechanism for deployment at the edge. The LocKedge system is based on a DNN model together with

TABLE I: Statistics of the original TON-IoT dataset.

N°	Label	Total Data Record	Ratio(%)
1	Scanning	7,140,161	31.96
2	DDoS	6,165,008	27.6
3	DoS	3,375,328	15.11
4	XSS	2,108,944	9.44
5	Password	1,718,568	7.69
6	Normal	796,380	3.56
7	Backdoor	508,116	2.27
8	Injection	452,659	2.03
9	Ransomware	72,805	0.33
10	MITM	1,052	0.0047

Principal Component Analysis (PCA) for feature engineering. The suggested method split the dataset into 4 partitions according to the attackers' IP address. Similarly, the authors in [2], [16] evaluated the impact of non-IID data on the development of FL-based IDS for IoT networks. These studies confirmed the finding about the performance reduction of FL in non-IID scenarios. However, they do not provide information on the implementation being used and data distribution among the clients. To this end, we designed an experimental set-up with the necessary non-IID characteristics presented in Section III.

In most of the aforementioned works, the model's parameters are initialised with random weights. However, a growing body of research [17], [18] has explored the impact of pre-training and initialisation on FL, especially in non-IID scenarios, suggesting that starting from a pre-trained model could reduce both the effect of data heterogeneity and the training time required to reach the final model. Our work contributes to the body of knowledge by exploring the impact of pre-training in FL-based IDS. In addition, we experiment with various aggregation methods to address the impact of non-IID distributed data.

III. METHODOLOGY

In this section, we describe the key processes of our FL-based IDS for IoT. These include the data distribution among multiple FL clients, the data preparation, and learning process.

A. Dataset

The selection of an appropriate dataset is a critical component to consider while designing our FL-based IDS for IoT devices. Most IDS datasets are unsuitable for FL because they cannot be properly disseminated among the clients in a non-IID fashion [16]. As a result, our suggested approach focuses on IoT datasets for IDS that can be partitioned based on the destination IP address, namely MedBIoT [19], BoT-IoT [20] and TON-IoT [6]. In particular, we opted for the TON-IoT dataset from the University of New South Wales, Canberra [6], as it presents the best ratio between benign and attack traffic compared to other IoT-focused datasets. The TON-IoT dataset is based on a realistic testbed that includes heterogeneous data sources and a wide range of attack types, allowing it to give a comprehensive evaluation of IoT security systems. Table I describes the TON-IoT dataset data samples.
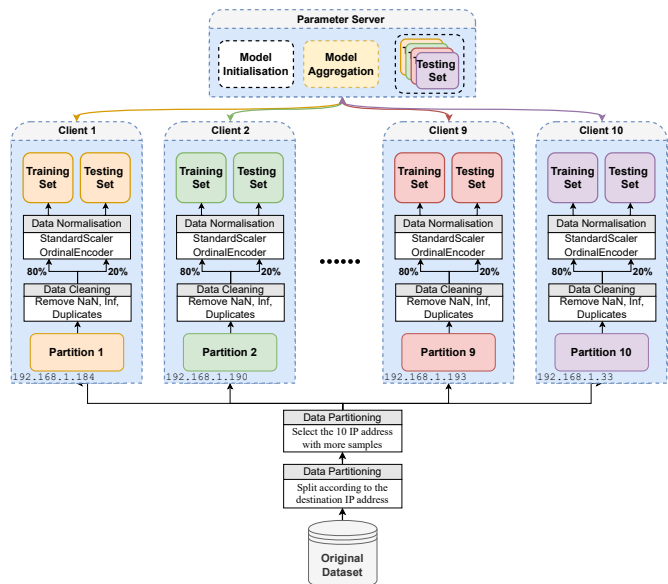

Fig. 1: High-level architecture of our FL-based IDS.

B. On-the-device Data Preparation

In our data partitioning strategy, we group the data samples by destination IP address, select the top ten IPs with the most samples, and assign each IP's samples to a single FL client. We used only the top ten IP addresses to avoid having clients with minimal data samples. Table II describes the data distribution among the ten clients after partitioning and preprocessing. Such distribution allows us to have the most realistic scenario with a high degree of non-IID-ness. Moreover, to simulate a realistic scenario where data are generated independently on each client, each FL client has to go through data pre-processing individually and independently from the other clients before local model training, as shown in Figure 1.

1) Data Cleaning: One of the major imperfections in intrusion datasets is missing data in collected samples. To handle missing values, we remove the rows containing NaN values since the dataset is large enough and the fraction of rows containing missing values is small. Hence, it will have minimal effect on performance. Moreover, we eliminate duplicate instances and samples with infinite values for the same reasons. After dropping the unwanted values, we removed unnecessary flow features such as "*ts*", "*src_ip*", "*src_port*", "*dst_ip*", and "*dst_port*" in order to ensure that the model learns the complexity of the flow properties instead of the specific endpoints responsible for each traffic, and avoid overfitting. We also encoded the categorical features with ordinal encoding and standardised the numerical ones. We then split the local on-the-device dataset into training and testing sets in the proportions of 80% and 20%, respectively. A stratified split is crucial in datasets with high-class imbalance since it permits a more accurate evaluation of the model's performance [21]. By maintaining the number of samples for each class, a stratified data split effectively ensures enough samples of the minority class in the training set.

TABLE II: Description of the class distribution for each client after pre-processing.

Client	# Samples	Scan	DDoS	XSS	Pwd	DoS	Normal	Back	Injection	Rans	MITM
1	3,786,885	815	3,502,650	576	26,460	192,130	16,202	-	48,052	-	-
2	2,740,330	636,963	993,069	285,436	278,833	303,583	149,315	-	93,129	-	2
3	2,535,874	1,167,320	6,906	344,136	864,611	13,108	48,046	-	91,740	-	7
4	1,984,149	568,501	465,525	280,915	95,029	399,568	79,285	-	95,323	-	3
5	1,378,579	13,382	630,127	604,691	2,883	86,721	27,304	18	12,889	-	564
6	1,165,842	1,161,124	210	-	-	-	4,508	-	-	-	-
7	1,008,113	412,493	452,559	-	4,444	116,463	22,154	-	-	-	-
8	680,480	680,446	-	-	-	-	22	-	-	-	12
9	603,465	336,770	3,642	-	37,835	181,710	21,745	133	-	21,629	1
10	425,249	384	-	-	-	-	-	423,122	3	1,737	-

2) Data Normalisation: Most detection algorithms only accept numerical input values. Therefore, we convert the categorical features into numerical features. To increase model performance, we normalise the whole set of numerical features without distorting disparities in the ranges of values. This stage is vital for removing any bias from raw data and optimising model training. It is important to note that in order to preserve privacy, we perform the normalisation on each FL client independently from the other clients. Table II describes the final dataset on each client after partitioning and preprocessing.

C. Deep Belief Networks

DBN is a generative model, composed of stacked layers of Restricted Boltzmann Machine (RBM). An RBM is an undirected energy-based model that consists of two layers, namely visible and hidden layer [22]. The training process of a DBN model requires two phases: unsupervised layer-by-layer training and supervised fine-tuning. During unsupervised pre-training, except for the first and last layer, each RBM layer is trained using the contrastive divergence algorithm [23], and the output of each RBM is used as input for the subsequent RBM layer. In the second phase, all weights are fine-tuned using back-propagation. DBNs can establish a meaningful pattern for the attacks which outperform traditional models. For more details on DBNs, please refer to our previous work in [24].

IV. EXPERIMENTS

All the experiments were conducted using an 8-core CPU with 16GB RAM in a macOS environment. The building, training, and evaluation of the Deep Learning (DL) model were performed using PyTorch and the Flower framework.

A. Model Training

In this study, we employ a DBN and DNN for multi-class classification. We train the global DL models across 50 rounds and two epochs for each training round, using ten clients with highly skewed data distributions. Since non-IID distributed datasets may deteriorate the performance of FedAvg [13], [14], we experiment with the FedProx [25] and FedYogi [26] aggregation methods, which are known for their robust convergence in heterogeneous settings [25], [26]. Moreover, we initialise the model with pre-trained weights on the server side and compare it against random initialisation. The pre-trained

TABLE III: Model architecture and hyperparameters.

(a) DBN

Attribute	Value
Input/Output Layer	38:10
Hidden Layer	100:150:200:50
Hidden Layers Act. Fct.	ReLu
Optimization Algorithm	Adam
Weight init.	Xavier initialiser
Bias init.	Zeros (0)
Gibbs step	1 step

(b) DNN

Attribute	Value
Input/Output Layer	38:10
Hidden Layer	128:128:64
Hidden Layers Act. Fct.	ReLu
Optimization Algorithm	SGD

weights are obtained using the data samples from the rest of the destination IP addresses excluded from the experimentation set-up, as described in Section III-B. The architecture and hyperparameters of the models are shown in Table III.

B. Evaluation Metrics

Depending on the type of dataset and its class distribution, a suitable set of metrics is required to assess a given model's performance reliably. In the case of intrusion datasets that mainly suffer from high-class imbalance, the accuracy metric cannot be relied upon [27]. Therefore, we chose the F1-score, Precision and Recall to evaluate the model. Furthermore, given the unequal distribution of the data based on the label, the weighted average is used to consider the size of each class. It is an important consideration, as it ensures that the distribution of instances in each class is considered.

C. Comparative Results

This section presents the experimental results based on the evaluation metrics defined in the previous section. Upon reviewing the literature, no study of FL-based IDS on non-IID data and reproducible methodology was found. Therefore, we could not compare our results against other works. Figure 2 shows the confusion matrices of the DNN and DBN with three different scenarios, namely centralised learning, FL and
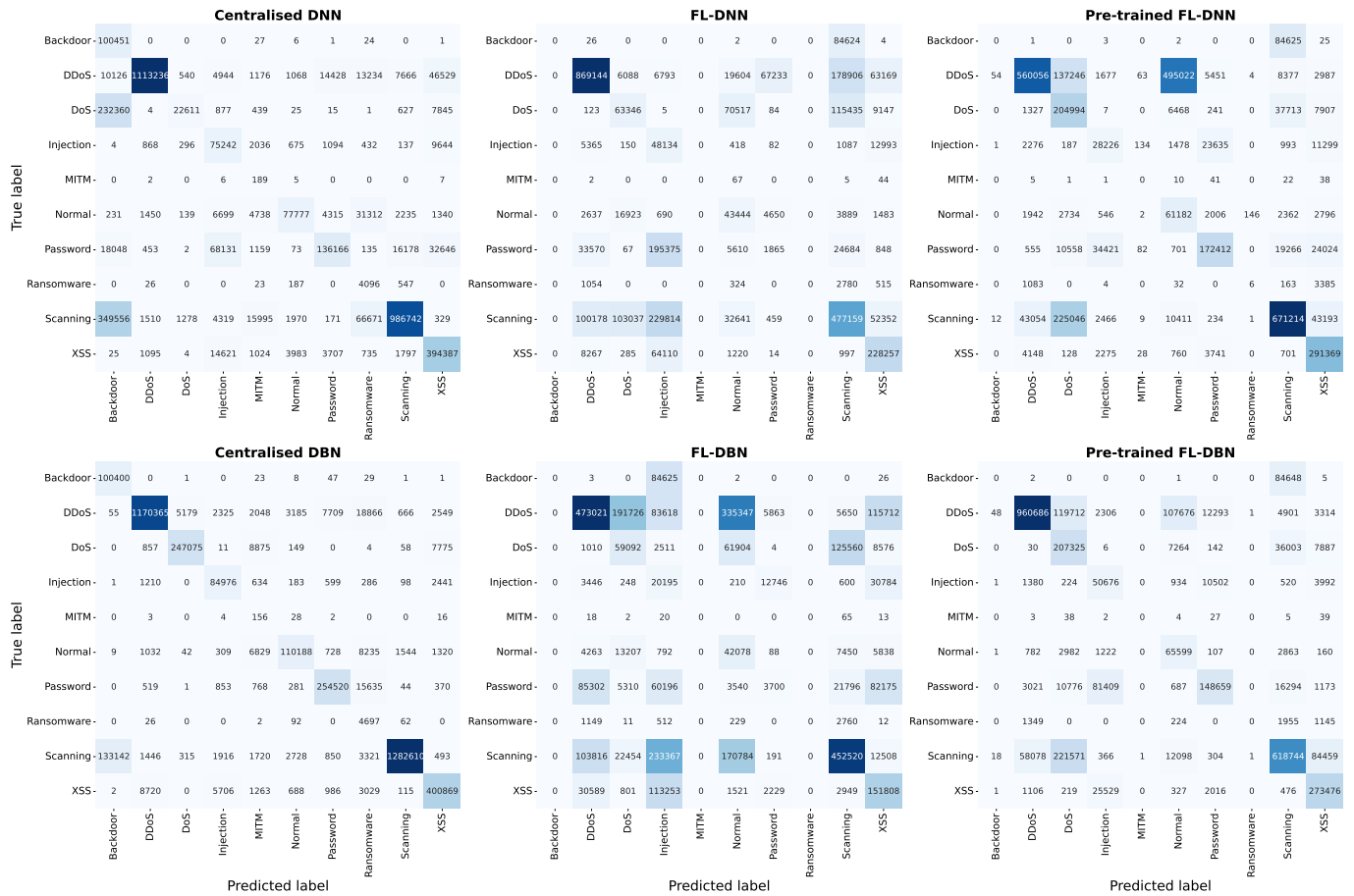
Fig. 2: Confusion Matrices.

pre-trained FL. The correct classifications are represented on the main diagonal of the confusion matrix whereas any off-diagonals denote the erroneously categorised traffic. As a result, the best model will consist of a confusion matrix with only diagonal elements and the rest of the elements near zero.

The results of the simulations are summarised in Figure 3. What stands out in these figures is the clear improvement in performance for the FedYogi and FedProx algorithms compared to FedAvg. These differences can be explained in part by their ability to tackle data heterogeneity. Furthermore, the FedProx simulation presents better convergence stability and prevents divergence through its regularisation hyperparameter.

Figure 4 depicts the averaged Precision, Recall, and F1-score achieved for the different approaches. On the one hand, we can observe that the centralised approach can detect intrusions with high Precision, Recall and F1-score. It achieves the best results with 97% Precision, 93% Recall and 94% F1-score. However, as mentioned earlier, the centralised approach requires the centralisation of a large amount of data, introducing significant delay at the pre-processing stage and making it easier for intruders to leak sensitive information.

On the other hand, the performance of the FL-DBN model with random initialisation correctly classified the network traffic with only 54% Precision, 37% Recall and 42% F1-
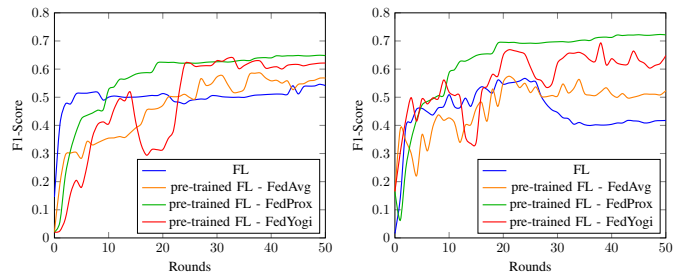
(a) DNN (b) DBN

Fig. 3: F1-score performance for 50 rounds and evaluation of three FL strategies (FedAvg, FedYogi, FedProx).

score, while the FL-DNN model achieved 57% Precision, 53% Recall and 54% F1-score. The model performance reduces significantly when the data is non-IID by up to 50%. As described by [14], training on non-IID data introduces challenges that deteriorate the performance of the IDS and lead to convergence issues. The performance reduction can be explained by the weight divergence of each client. To mitigate this issue, we propose to start FL from a pre-trained model
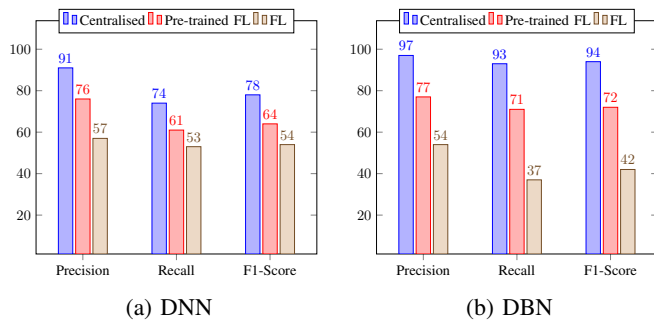
(a) DNN	(b) DBN

Fig. 4: Precision, Recall and F1-score achieved by the proposed multi-attack classification for the different approaches.

to reduce the effect of data heterogeneity and increase the stability of global aggregation in FL. We can see that the performance is significantly improved over FL with random initialisation. Indeed, the pre-trained FL-DBN and FL-DNN models have an F1-score of 72% and 62% respectively. Thus, it demonstrates that the pre-trained model enables the training of more accurate FL-based IDS (up to 30%) than is possible when starting with random initialisation.

V. CONCLUSION AND FUTURE WORK

The purpose of the current study was to design a multi-class FL-based IDS for IoT and evaluate it under realistic non-IID conditions. Our experimental results confirmed that training an FL-based IDS on heterogeneous data, either relying on DNNs or DBNs, can significantly reduce the model performance by up to 50%. As a mitigation, we investigated the utilisation of a pre-trained Global model. We demonstrated that pre-training the Global model in FL can address data heterogeneity to some extent. In addition, we demonstrated that FedProx and FedYogi increase detection performance since they are designed to address highly non-IID datasets.

A limitation of this work is that we consider a pre-trained model available. However, it may not always be possible to get public data at the server and pre-train the initial model. As such, exploring pre-training with synthetic data would be a fruitful area for further work. Finally, it is important to note that normalising the data on each client individually can deteriorate the performance of the model. This could be improved by calculating the normalisation parameters centrally, relying on the clients' statistical characteristics. However, sharing the statistical parameters of the clients can disclose sensitive information. Hence, we plan to investigate how we can achieve better normalisation whilst preserving data privacy.

REFERENCES

[1] V. Kumar, S. Gunner, T. Spyridopoulos, A. Vafeas, J. Pope, P. Yadav, G. Oikonomou, and T. Tryfonas, "Challenges in the design and implementation of iot testbeds in smart-cities: A systematic review," 2023.

[2] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke, and L. Shu, "Federated deep learning for cyber security in the internet of things: Concepts, applications, and experimental analysis," *IEEE Access*, 2021.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

[4] N. A. A.-A. Al-Marri, B. S. Ciftler, and M. M. Abdallah, "Federated mimic learning for privacy preserving intrusion detection," *2020 IEEE International BlackSeaCom*, 2020.

[5] E. M. Campos, P. F. Saura, A. González-Vidal, J. L. Hernández-Ramos, J. B. Bernabé, G. Baldini, and A. Skarmeta, "Evaluating federated learning for intrusion detection in internet of things: Review and challenges," *Computer Networks*, vol. 203, p. 108661, 2022.

[6] N. Moustafa, "A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets," *Sustainable Cities and Society*, vol. 72, p. 102994, 2021.

[7] S. Agrawal, S. Sarkar, O. Aouedi, G. Yenduri, K. Piamrat, M. Alazab, S. Bhattacharya, P. K. R. Maddikunta, and T. R. Gadekallu, "Federated learning for intrusion detection system: Concepts, challenges and future directions," *Computer Communications*, vol. 195, pp. 346–361, 2022.

[8] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, "Internet of things intrusion detection: Centralized, on-device, or federated learning?" *IEEE Network*, vol. 34, no. 6, pp. 310–317, 2020.

[9] Z. Chen, N. Lv, P. Liu, Y. Fang, K. Chen, and W. Pan, "Intrusion detection for wireless edge networks based on federated learning," *IEEE Access*, vol. 8, pp. 217 463–217 472, 2020.

[10] X. Hei, X. Yin, Y. Wang, J. Ren, and L. Zhu, "A trusted feature aggregator federated learning for distributed malicious attack detection," *Computers & Security*, vol. 99, p. 102033, 2020.

[11] 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 of Things Journal*, 2022.

[12] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in iot-edge devices," *IEEE Internet of Things Journal*, 2022.

[13] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," 2019.

[14] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *ArXiv*, 2018.

[15] T. T. Huong, T. P. Bac, D. M. Long, B. D. Thang, N. T. Binh, T. D. Luong, and T. K. Phuc, "Lockedge: Low-complexity cyberattack detection in iot edge computing," *IEEE Access*, vol. 9, 2021.

[16] V. Rey, P. M. Sánchez Sánchez, A. Huertas Celdrán, and G. Bovet, "Federated learning for malware detection in iot devices," *Computer Networks*, vol. 204, p. 108693, 2022.

[17] H.-Y. Chen, C.-H. Tu, Z. Li, H. W. Shen, and W.-L. Chao, "On the importance and applicability of pre-training for federated learning," in *International Conference on Learning Representations*, 2023.

[18] J. Nguyen, K. Malik, M. Sanjabi, and M. Rabbat, "Where to begin? exploring the impact of pre-training and initialization in federated learning," 2022.

[19] A. Guerra-Manzanares, J. Medina-Galindo, H. Bahsi, and S. Nõmm, "Medbiot: Generation of an iot botnet dataset in a medium-sized iot network," in *Proceedings of the 6th International Conference on Information Systems Security and Privacy*, 2020.

[20] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. P. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Gener. Comput. Syst.*, vol. 100, pp. 779–796, 2018.

[21] N. Y. Hammerla and T. Plötz, "Let's (not) stick together: Pairwise similarity biases cross-validation in activity recognition," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2015.

[22] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07, 2007.

[23] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, p. 1771–1800, Aug. 2002.

[24] O. Belarbi, A. Khan, P. Carnelli, and T. Spyridopoulos, "An intrusion detection system based on deep belief networks," in *SciSec*, 2022.

[25] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems*, 2020.

[26] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," in *International Conference on Learning Representations*, 2021.

[27] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: Bagging, boosting, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics*, 2012.