Applied Intelligence (2024) 54:8132–8153
https://doi.org/10.1007/s10489-024-05589-6

Random forest with differential privacy in federated learning framework for network attack detection and classification

Tijana Markovic[1] · Miguel Leon[1] · David Buffoni[2] · Sasikumar Punnekkat[1]

Accepted: 2 June 2024 / Published online: 21 June 2024

Abstract
Communication networks are crucial components of the underlying digital infrastructure in any smart city setup. The increasing usage of computer networks brings additional cyber security concerns, and every organization has to implement preventive measures to protect valuable data and business processes. Due to the inherent distributed nature of the city infrastructures as well as the critical nature of its resources and data, any solution to the attack detection calls for distributed, efficient and privacy preserving solutions. In this paper, we extend the evaluation of our federated learning framework for network attacks detection and classification based on random forest. Previously the framework was evaluated only for attack detection using four well-known intrusion detection datasets (KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017). In this paper, we extend the evaluation for attack classification. We also evaluate how adding differential privacy into random forest, as an additional protective mechanism, affects the framework performances. The results show that the framework outperforms the average performance of independent random forests on clients for both attack detection and classification. Adding differential privacy penalizes the performance of random forest, as expected, but the use of the proposed framework still brings benefits in comparison to the use of independent local models. The code used in this paper is publicly available, to enable transparency and facilitate reproducibility within the research community.

Keywords Attack detection · Attack classification · Random forest · Federated learning · Differential privacy

1 Introduction

Communication networks are crucial components of the underlying digital infrastructure in any smart city setup, and the identification of anomalies and intrusions in networks is of paramount importance for providing the intended services to various stakeholders, such as public departments, enterprises, and citizens. These networks are heterogeneous, with large numbers of diverse sensing nodes collecting periodic data and transmitting them to various coordination and decision-making services. Integrating the data created in those networks into diverse platforms, providing stakeholder specific views based on their access rights as well as arriving at intelligent conclusions based on data are the key activities of any smart city framework.

The widespread usage of computer networks also brings many cyber security concerns, and every organization has to implement preventive measures to avoid compromising their valuable data and assets. In the growing landscape of cyber security threats, both organized and amateur attempts to access and jeopardize smart city infrastructure become a serious concern to public authorities [1, 2]. One of the necessary protective mechanisms is Intrusion Detection System (IDS), and this research area has received a lot of attention during the past decade [3]. IDS is a software or hardware

Tijana Markovic and Miguel Leon contributed equally to this work.

✉ Miguel Leon
miguel.leonortiz@mdu.se

Tijana Markovic
tijana.markovic@mdu.se

David Buffoni
david.buffoni@molnlycke.com

Sasikumar Punnekkat
sasikumar.punnekkat@mdu.se

[1] School of Innovation, Design and Engineering, Mälardalen University, Universitetsplan 1, Västerås 72220, Vastmanland, Sweden

[2] Mölnlycke Healthcare AB, Gothenburg, Sweden

system that monitors the events occurring in a computer system or network and analyzes those events for signs of intrusion or violations of security policies [4].

In recent years, artificial intelligence techniques are being widely used in the field of network security, especially for Intrusion Detection (ID). Machine learning (ML) algorithms can learn from data how to distinguish between normal and abnormal activities, and this ability has been proven to be very effective for the development of reliable IDSs [5–7]. We have been exploring efficient ML techniques for ID as part of two large EU projects InSecTT[1] and DAIS,[2] where the former provided us with a strong understanding of the applicability of various ML methods for ID, and in the latter we plan to apply them in a smart city application. During the discussions with our industrial partners, we have identified the following aspects that are of primary importance for any chosen approach:

- Accuracy
- Efficiency, in terms of the memory, computation and communication requirements
- Privacy-preserving ability.

Many studies that compared the performances of different ML algorithms on different benchmark datasets concluded that the Random Forest (RF) algorithm has the highest accuracy [8–12]. RF requires a lot of data for training purposes, as any other ML algorithm, so one of the main obstacles is the security of the provided data. Centralizing the locally collected data can raise various privacy and security concerns that can be overcome by implementing a collaborative learning approach, without the need of data sharing, and this approach is called federated learning (FL) [13, 14]. FL is a decentralized learning technique that trains models locally on clients and transfers them to the centralized server [15, 16]. FL has three main alternatives, where various approaches are used to distribute data among different clients:

- Vertical Federated Learning (VFL): Each client uses the same instances but has access to different features.
- Horizontal Federated Learning (HFL): Each client uses the same features but has access to different instances.
- Transfer Federated Learning (TFL): A combination of VFL and HFL where each client has limited access to both features and instances.

where, in the context of the paper, one instance is one network reading (e.g. a packet sent through the network), while one feature is specific information about the instance (e.g. protocol, duration, etc.).

Another concern is that the model itself can be attacked and vulnerable data can be extracted from the model. This can be solved by using Differential Privacy (DP), which is a mechanism that provides a quantifiable measure of data anonymization by adding random noise during the training process [17]. In this way, an attacker cannot derive any data by accessing the information of the model.

In this paper, we are using a FL framework based on RF that was previously proposed in [18]. The framework employs HFL approach and its main idea is to train independent RFs on clients using the local data, merge independent models into a global one on the server and send it back to the clients for further use. The developed framework was evaluated for attack detection on the most commonly used ID datasets (KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017). The novelty of this framework lies in the provision of different alternatives to create the global RF in the server, for subsequent distribution to the different clients. This paper extends the framework by including DP into the different RFs. Additionally, the evaluation of the framework is extended by:

- Evaluating different data division approaches.
- Evaluating the framework performance for attack classification.
- Evaluating the framework performance when using RF with DP for both attack detection and attack classification.

This paper is organized as follows. Section 2 presents the state of the art in federated learning, random forest in a federated learning setup, and differential privacy. Section 3 presents the main components of the used framework. The details of the datasets and preprocessing techniques used, as well as the experimental setup, are given in Section 4. In Section 5, we present the results of the conducted experiments, followed by the conclusion and plans for future work in Section 6.

2 Related work

2.1 Federated learning

Federated Learning (FL) [19] is a ML setting where computer devices learn a task in a collaborative way without sharing data with a centralised server. For example, ML algorithms can be trained across multiple devices and servers with decentralised data over multiple iterations [20]. FL is an iterative process of training a global ML model by aggregating

[1] https://www.insectt.eu/

[2] https://dais-project.eu/

a set of local ML models trained on multiple devices. In each training round, a set of devices is selected to receive the current version of the global model from the server. Then, each device trains a local version of the model on the locally present data and sends back the updated version to the server. The server aggregates all the local versions and repeats the process for another round until the target performance is attained. Various ML algorithms have been adapted to FL setting [19, 21–23] in various areas, including industry engineering, healthcare, computer vision, finances, etc.

FL setting is a natural candidate for training and deploying network intrusion detection systems, as it reduces the computational load on the central server, reduces the communication bandwidth (data remains locally), and enables data privacy. Surveys of the various ML methods using FL for related network intrusion detection tasks are presented in [15, 24]. Most of the presented works in these two surveys use deep learning techniques such as Neural Networks [25], Convolutional Neural Networks [26, 27], Generative Adversarial Networks [28] or Recurrent Neural Network [29].

Contrary to gradient-based methods, the implementation and the efficiency of Gradient Boosting Decision Trees in a FL setting is shown in [30]. In the same vein, [31] used a FL Gradient Boosted Decision Tree (GBDT) approach to solve a network intrusion detection task. According to them, one advantage of their algorithm compared to the Deep Learning approach is that GBDT is more interpretable while reaching similar predictive performances. Due to the same reasons, interpretability and scalability, we also opted for a tree-based approach in this work. However, because of their great parallelism and their high prediction performances, we consider Random Forests as a perfect candidate for FL in solving a network intrusion detection task.

In [32] and [33], federated version of RF is applied for healthcare related applications. In [32], their RF performs a weighted combination of the forests trained locally. They used the Matthews correlation coefficient (MCC) for boosting local models with high classification performance in the final combination. In [33], authors focused on the comparison between local models trained on incomplete information with a Federated RF. Both approaches showed a better performance of the Federated RF compared to local RF models.

2.2 Federated learning and privacy-preserving techniques

FL enables the preservation of data privacy by design from adversarial exposition, but private information can be reconstructed from the local models that are sent to the server [34]. Several approaches have been proposed to handle this scenario in FL, by adding privacy techniques such as k-anonymity [35], data encryption [36] or differential privacy [37].

The goal of privacy techniques is to monitor what can be learned from the data and several works enhanced RF with a privacy layer. For example, [38] proposed a version of RF by using anonymization methods on the data to preserve data privacy. Furthermore, [39] opted for the k-anonymity approach in RF for FL settings. Using data encryption techniques offers a security layer by controlling and protecting access to the data. In [40], authors incorporated a homomorphic encryption mechanism on the data in their implementation of a RF in FL ensuring a data security property. In [41], authors proposed a way to create a decentralized federated forest in a ID scenario based on the blockchain. Similar to these approaches, a Homomorphic Encryption and Secure Multi-Party Computation mechanisms are used for privacy by [42] in the context of Federated version of Gradient Boosted Decision Trees. However, homomorphic encryption techniques increase the algorithm complexity and are computationally time-consuming.

Differential privacy offers a rigorous definition of privacy. Assuming we consider an algorithm that queries or analyzes data and computes statistics about them, then Differential Privacy would be applied on the algorithm's output. If by looking at the output, one cannot tell whether any private data of an individual was included in the original data set or not, then the algorithm is differentially private [37]. Thanks to that definition, we can guarantee that private information about individuals in a dataset will not be leaked.

Differential Privacy provides a formal notion of privacy by adding calibrated noise to the parameters of the ML algorithm [43]. In [44], authors surveyed how the addition of differential privacy affects ML algorithms such as their inputs, outputs, and objective function. In our work, we are focused on using Differential Privacy on tree-based methods and more especially to Random Forests in the FL setting.

2.3 Random forest with differential privacy

The survey by Fletcher [45] provides an overall understanding of the tree-based approaches such as Random Forest with Differential Privacy property. This work mainly focuses on how to design a Decision Tree to preserve privacy without decreasing their classification capabilities. In Random Forest and more especially when a Decision Tree is built, data is queried by the algorithm to either split the node (to partition the data based on the best attribute) or for predicting the class label of the data records in the leaves. Patil et al. [46] were the first to adapt the definition of Differential Privacy from a greedy Decision tree to a Random Forest. Adding noise to Decision tree outputs, generally reduces the algorithm's

accuracy. To overcome that, the authors proposed a hybrid Decision Tree algorithm that balances the privacy and the classification accuracy of the Random Forest algorithm based on Differential Privacy. Fletcher and Islam [47] focused on the Gini index which is used while building Decision Trees. Based on it, they defined the quantity of noise added to make the forest differentially private. Having precise control of the added noise allows them to limit the accuracy performance loss by the Differential Privacy definition. A similar approach was used in [48], where the goal of controlling the quantity of noise on the outputs and the algorithm's parameters tuning depends on the theory of Signal-to-Noise Ratios. All these approaches use the Laplacian method as a Differential Privacy mechanism.

An alternative way of applying Differential Privacy is to use the Exponential mechanism as in [17]. Fletcher and Islam [17] proposed that the Random Forest's leaves return the majority class label instead of the class counts. A negative aspect of such approach is that it reduces the learning ability, however, it makes the algorithm more private by design. The authors experimentally validated this approach showing that Random Forest was accurate even after adding Differential Privacy.

Other approaches have been designed for making decision trees private such as [49–52]. In [49], authors proposed the use of permute-and-flip which randomly chooses a value from a set of options given a weight and a privacy parameter. This approach never performs worse than the Exponential mechanism in expectation. Sun et al. [50] combined several mechanisms for building the trees. The Exponential mechanism selects the split nodes and the Laplace one adds noise to leaf nodes which results in a tighter use of the privacy budget. In the same vein, in [52] the authors proposed to combine an Exponential mechanism and a Laplace one during the tree construction. The first mechanism, Exponential, is used to protect the sensitive features which are given as inputs to the second mechanism, Laplace, which ensure the protection of the leaf nodes. However, they opted for a Gradient Boosting Decision Trees approach instead of Random Forests as preferred in our work. Li et al. [51] used the Out-of-Bag Estimation which perturbs the true number of data for building the tree.

Among the presented related works the closest contributions to this paper can be found in [32] and [33], which are focused on healthcare applications and perform different merging approaches. In our work, we evaluate various approaches for combining local RF models for cybersecurity applications. In addition, differential privacy property is added to our model making it private by design and providing a countermeasure to potential data poisoning attacks [53]. To achieve that, we decided to follow the work from [17] where differential privacy is done with the Exponential mechanism giving strong privacy guarantees and high accuracy performances in practice. We extend their work from the centralised setting to a federated learning one.

3 Random Forest with differential privacy in a federated learning framework

3.1 Random forest

Random Forest (RF) combines the predictions of different Decision Tree (DT) algorithms into a final prediction [7]. DT is ML algorithm used for classification [54–56] and/or regression [57]. In this paper, we will focus only on classification, since that is the main objective of the intrusion detection research area. An example of a DT is presented in Fig. 1. As we can see, DT is formed by decision nodes and leaf nodes. A decision node takes the most relevant feature

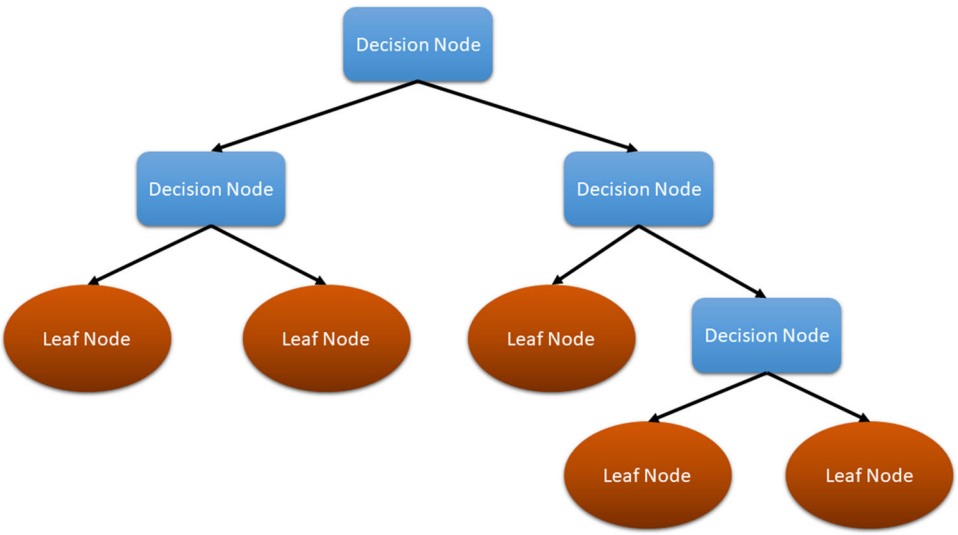

Fig. 1 Overview of Decision Tree. The decision nodes perform a choice according to a data attribute. The leaf nodes output the number of data samples belonging to each class

from the dataset that has not been used before and uses it as a condition to divide the dataset into subsets. If a node does not undergo further divisions, it is a leaf node that contains a final prediction. There are different methods to select the most relevant feature [58], and we are using the following two methods:

- *gini* - attempts to find and isolate the largest homogeneous class from the rest of the data. For this purpose, the Gini Index (GI) is calculated for all the different features. The GI for a feature F (denoted by $GI(D|F)$) is calculated as follows:

$$GI(D|F) = \sum_{f \in F} \left(\frac{|D|F = f|}{|D|} \times GI(D|F = f) \right) \tag{1}$$

$$GI(D|F = f) = 1 - \sum_{c \in C} P(C = c, F = f)^2 \tag{2}$$

  where D is the entire dataset, F is a certain feature, and f is the value that the feature takes. $|D|F = f|$ is equal to the number of instances within the dataset that takes f as the value for the feature F, while $|D|$ is the number of instances of the entire data set. Finally, $P(C = c, F = f)$ is the probability of selecting the class c out of all classes $|C|$ within the dataset $D|F$ when selecting f as the value for F. After calculating the $GiniIndex$ of all the features, the one with the lowest value is selected as the parent node. Then, further divisions are performed following the same principle.
- *entropy* - attempts to minimize the within-group diversity. For this reason, this method calculates the information gain (IG) in order to split the dataset into subsets using a certain feature F. This is done by using entropy (E) and it is calculated for all the different features. The information gain for a specific feature F (denoted by IG(D,F)) is calculated as follows:

$$IG(D, F) = E(D) - E(D|F) \tag{3}$$

$$E(D|F) = \sum_{f \in F} \left(\frac{|D|F = f|}{|D|} \times E(D|F = f) \right) \tag{4}$$

$$E(D|F = f) = \sum_{c \in C} (P(C = c, F = f) \times \log_2(P(C = c, F = f))) \tag{5}$$

  where D is the entire dataset, $D|F$ stands for the dataset after splitting it by a certain feature F, f is the value that F takes, c is a class of all possible classes (C) and $P(C = c, F = f)$ is the probability of selecting the class c out of all classes (C) within the dataset $D|F$ when selecting f as the value for F. Finally, $|D|F = f|$ represents the number of cases left after assigning the value f to the selected feature, while $|D|$ is the size of the entire dataset. The feature with a higher IG is then selected as the parent node and further divisions are made.

Multiple DTs are used to build RF as shown in Fig. 2. The number of DTs that are used is one of the hyper-parameters of RF. In order to create those DTs different subsets of data must be used, since the usage of the same data will produce the exact same DT. The division on the subsets is performed randomly, using a certain percentage of the entire dataset. In addition, the different DTs in RF would normally use

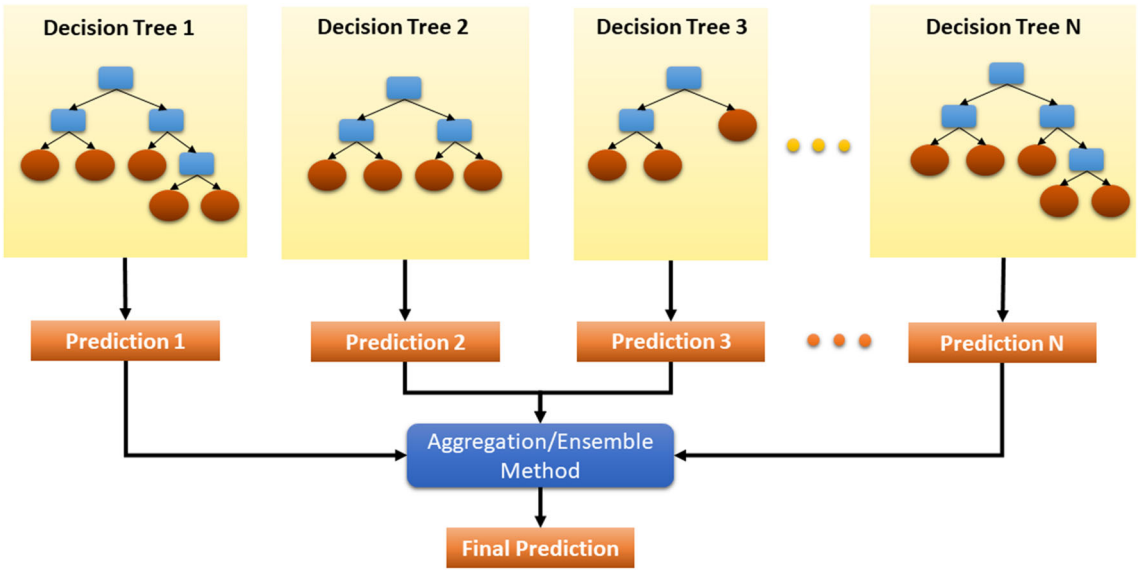

Fig. 2 Overview of Random Forest

different features. However, we decided to keep all the features on all the trees.

The final step is to ensemble or to aggregate the predictions of the different DTs into the final prediction given by RF. In this paper, we are using two different ensemble methods:

- *Simple Voting (SV)* - takes a majority vote as a predicted class.
- *Weighted Voting (WV)* - takes a majority vote as a predicted class, but weights the accuracy of each DT for the predicted class by multiplying it with the average of the accuracy of all classes for that DT.

3.2 Differential privacy

Differential Privacy (DP) guarantees the privacy of individuals in datasets and prevents unauthorized extraction of private and sensitive information [37]. There exist many different ways of applying DP on the ML models, often called mechanisms. Usually, a mechanism adds probabilistic noise to the output to make it differentially private. Several mechanisms focus on adding noise to numerical predictions of the algorithms. The Laplacian mechanism, proposed by [59], is one of these cases. For example in a decision tree, the leaf node predicts the class label of the data by returning class counts. The Laplacian mechanism ensures DP by altering these counts by adding a noise sampled from the Laplace distribution, which is the case for differentially private decision trees proposed in [46–48, 60].

An alternative is to use the Exponential mechanism [61], which provides an approximation of the best elements from the set. For example, in tree-based algorithms, instead of returning class counts and adding noise to them, the goal is to return the approximate majority class, i.e. one of the classes with the highest number of counts. The probability of selecting one of the outputs (z), represented by $Pr(f(x) = z)$ is given in (6).

$$Pr(f(x) = z) \propto exp\left(\frac{\epsilon \times u(z, x)}{2 \times \Delta(u)}\right) \tag{6}$$

where $u(z, x)$ represents the scoring function of the output z with respect to the data (x), $\Delta(u)$ refers to the sensitivity of u [59], which show the deviation when different inputs are used, and ϵ is a parameter that allows adapting the strength of privacy.

In this paper, we followed the work done in [17], where authors integrated the differential privacy property into RF using the Exponential mechanism with a smooth sensitivity function from [62]. Different values of ϵ were evaluated and it was concluded that a smaller value of ϵ ensures high privacy but drastically impacts the prediction performances. The same approach is applied in this paper, but using an FL-based setting. In Fig. 3, we can see DT with DP.

3.3 Random Forest with differential privacy in a federated learning framework

In this paper, RF is used in a FL setup where each client receives data that are not available to others. Independent RFs are trained on these clients and sent to a server, where a global RF is created as a combination of DTs from the clients. The decisions made by individual DTs are preserved and integrated into the global RF. Each DT which is included in the federation contributes to the final decision based on its own classification outcome. With this setup, we can say that we are using a horizontal approach [63] since the data from different clients have the same structure (same number of features). The novelty of this framework lies in the inclusion of DP into the different RFs, as well as the provision of

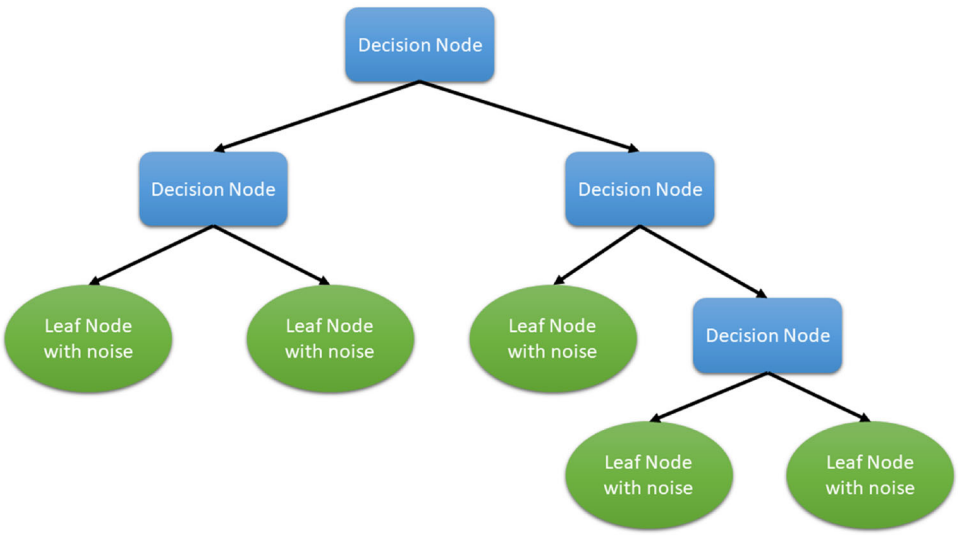

Fig. 3 Overview of Decision Tree with Differential Privacy where the leaf nodes output the majority class instead of the class counts. In that case, looking at the output, one cannot tell whether any private data of an individual was included in the original dataset or not, making the algorithm private

different alternatives to create the global RF in the server to later be distributed to the different clients. An overview of the proposed framework can be found in Fig. 4.

In order to select the DTs to be included in the global RF, the performance of each DT is evaluated using two different methods:

- *Accuracy (A)* - general accuracy of the DT in the validation set
- *Weighed accuracy (WA)* - general accuracy of the DT (in the validation set) multiplied by the average accuracy of the same DT for all different classes in the validation set. In this way, DTs that perform well in a larger number of classes are prioritized.

To perform this combination, different approaches were used to decide which DTs will be merged to create the global RF:

- *Global RF created by Sorting DTs per RF based on Accuracy (RF_S_DTs_A)* - DTs per RF are sorted based on the accuracy and the best ones from each RF are selected
- *Global RF created by Sorting DTs per RF based on Weighed Accuracy (RF_S_DTs_WA)* - DTs per RF are sorted based on the weighted accuracy, and the best ones from each RF are selected
- *Global RF created by Sorting All DTs based on Accuracy (RF_S_DTs_A_All)* - all DTs are assembled, sorted based on the accuracy, and the best ones are selected
- *Global RF created by Sorting All DTs based on Weighed Accuracy (RF_S_DTs_WA_All)* - all DTs are assembled, sorted based on the weighted accuracy, and the best ones are selected

The maximum number of DTs (MaxDTs) that can be used for generating the global RF is the number of DTs per RF multiplied by the number of clients. The number of the best DTs that will be included in the global RF is a hyper-parameter that may vary from 1 to MaxDTs for RF_S_DTs_A_All and RF_S_DTs_WA_All, and from the number of clients to MaxDTs for RF_S_DTs_A and RF_S_DTs_WA.

The global RF that is created on the server is returned to the clients to be used in the future, which provides clients more knowledge without the need of sharing data.

4 Experiments

4.1 Datasets

The experiments were conducted on four publicly available datasets, that are among the most frequently used datasets in the ID research area [5]. In order to create those datasets, the network traffic was recorded during normal behavior and during different network attacks that were simulated. The traffic was recorded in the form of network packets and preprocessed to create the features. Each packet is characterized

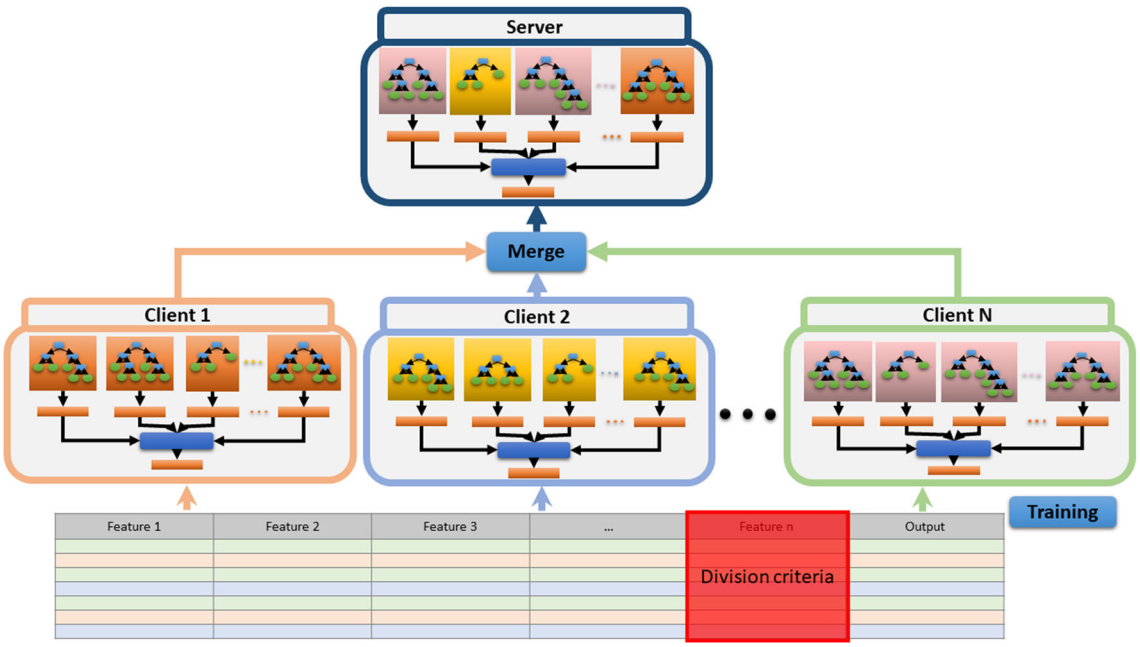

Fig. 4 Architecture of a Random Forest in a Federated Learning setup with differential privacy. There are N clients which train RFs locally and transfer them to the centralized server. The independent RFs are merged on the server to create a global RF which is returned to the clients for further use. The green leaf nodes in DTs represent leaf nodes with Differential Privacy, where the output is the majority class

Table 1 Basic information about KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 datasets

Dataset	No. of features	No. of instances	File used for the experiments
KDD [64]	41	4 898 431	kddcup.txt
NSL-KDD [65]	41	148 517	KDDTrain+.txt
UNSW-NB15 [66]	42	2 540 044	UNSW_NB15_trainingset.csv
CIC-IDS-2017 [67]	78	2 830 743	{Tu.-WorkingHours W.-workingHours Th.-WorkingHours-Morning-WebAttacks F.-WorkingHours-Afternoon-DDos F.-WorkingHours-Afternoon-PortScan F.-WorkingHours-Morning} .pcap_ISCX.csv

by certain features and labeled as normal or as some type of network attack. More information about each dataset is provided in Table 1.

4.2 Datasets pre-processing

From each of the datasets that were described in the previous section, we selected a certain part to use for the experiments (the last column in Table 1) and we removed the instances that belong to classes with less than 800 cases in CIC-IDS-2017. All features from the original datasets were used, except for CIC-IDS-2017, where two features were removed (flow of bytes and flow of packets per second). The original features were pre-processed depending on their type. Numeric features were normalized to a range between 0 and 1 using min/max approach, categorical features were one-hot encoded, while binary features were not changed. The output label was encoded into the numerical values for attack classification, while for attack detection normal instances were labeled with 0 and all the others with 1.

The datasets were divided into the training set, validation set, and testing set with a 70%-10%-20% distribution, and then split into subsets. One feature was used as a division criteria for an HFL setup: "protocol" was used for KDD and NSL-KDD, "service" was used for UNSW-NB15, and "destination port" was used for CIC-IDS-2017. Subsets that had less than 50 normal or malicious instances were not used. This process resulted in 3, 3, 6, and 14 subsets for KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-201 dataset, respectively.

Summary information after performing HFL division and pre-processing is given in Table 2.

4.3 Experimental setup

Four different experiments have been performed on the pre-processed data, for two different problems: Attack Detection (AD) and Attack Classification (AC). Python programming language is used to implement the algorithms. The sklearn[3] [68] ML library implementation of DT classifier was used. The differential privacy was implemented using the IBM differential privacy library (diffprivlib[4]) [69]. The code is publicly available on GitHub.[5]

An explanation of each experiment (EXP) is given below.

- *EXP 1 - Selection of RF hyper-parameters*: The experiment was conducted before splitting the datasets into subsets, with the goal of finding the best combination of RF hyper-parameters for a specific dataset and specific problem. Hyper-parameters that were tested include the number of DTs (odd numbers between 1 and 100), splitting rule (gini or entropy), and ensemble method (SV or WV). The best combination of hyper-parameters that was discovered in this experiment was used as the RF setup for all subsequent experiments for the specific problem on a specific dataset.
- *EXP 2 - Evaluation of independent RFs on different clients*: For each client an independent RF was trained on data from its subset, using the best combination of hyper-parameters from EXP1. The number of clients corresponds to number of subsets in each dataset, which can be seen in column S. of Table 2. Different methods of obtaining subsets were tested in this experiment:
  - EXP 2.1 - Subsets obtained using a specific feature as a division criteria, as explained in Section 4.2.
  - EXP 2.2 - Subsets obtained using random division of data among clients, such that each client gets the same amount of data.
  - EXP 2.3 - Subsets obtained using random division of data among clients, such that each client gets the same amount of data as in the EXP 2.1.

[3] https://scikit-learn.org/stable/

[4] https://github.com/IBM/differential-privacy-library

[5] https://github.com/vujicictijana/RF_FL

Table 2 Information and distribution of the used instances for KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 datasets after performing HFL division and pre-processing

Dataset	Inst.	F.	C.	S.	Subsets			
					Feature Value	No. of Inst.	% of Inst.	% of Attacks
KDD	493 347	115	4	3	icmp	283 235	57.41%	99.5%
					tcp	189 786	38.47%	59.5%
					udp	20 326	4.12%	5.7%
NSL-KDD	125 597	119	4	3	icmp	8 090	6.44%	83.8%
					tcp	102 517	81.62%	47.7%
					udp	14 990	11.93%	17.1%
UNSW-NB15	79 924	177	9	6	–	45 516	56.95%	39.9%
					dns	21 367	26.73%	85.6%
					ftp	1 550	1.94%	51.1%
					ftp-data	1 396	1.75%	32%
					http	8 244	10.31 %	51.3%
					smtp	1 851	2.32%	65.7%
CIC-IDS-2017	1 543 535	75	7	14	21	11 781	0.76%	69.4%
					22	13 498	0.87%	45.5%
					53	643 986	41.72%	0.03%
					80	541 594	35.09%	70.6%
					88	3 920	0.25%	4.1%
					135	749	0.05%	21.4%
					139	1 921	0.12%	10.3%
					389	4 477	0.29%	3.6%
					443	312 986	20.28%	0.08%
					445	1 195	0.08%	15%
					465	2 656	0.17%	6%
					1124	259	0.02%	61.8%
					3268	1 903	0.12%	8.4%
					8080	2 610	0.17%	54.4%

Inst. , F. , C. , and S. stand for number of instances, features, classes, and subsets, respectively

For EXP 2.1 two different options were considered for testing: RFs were tested on the data from their own subsets and RFs were tested on the entire testing set. For EXP 2.2 and EXP 2.3 RFs were tested on the entire testing set.

- *EXP 3 - Global RF based on Federated Learning*: Independent RFs were combined into a global one using four different merging methods (RF_S_DTs_A, RF_S_DTs_WA, RF_S_DTs_A_All, RF_S_DTs_WA_All) and varying number of DTs. The global RF was tested on the entire testing set and the performances of global RF were compared with the performances of independent RFs on the entire testing set.
- *EXP 4 - Global RF with differential privacy based on Federated Learning*: Independent RF with differential privacy was trained for each client on data from its subset (with respect to the division criteria) and tested on the entire testing set. Four different values of ϵ parameter were tested: 0.1, 0.5, 1 and 5. After that, the independent RFs were combined into a global one using the combination of the merging method and the number of DTs that had the best performance in EXP3 for the specific problem in the specific data set. The global RF was tested on the entire testing set and the results and performances of global RF were compared with the performances of independent RFs with differential privacy.

The performance of the ML algorithms was measured using different metrics: accuracy and F1 score [70].

5 Results

As explained in Section 4.3, results will be divided into four sections where the selection of the hyper-parameters is given in Section 5.1. Section 5.2 explores different division of the datasets into different clients and the performance of

independent RFs. In Section 5.3, a global RF is formed by combining different trees from the RF trained on the clients and compared with the individual RFs. Lastly, in Section 5.4, we have performed the same experiments as in Sections 5.2 and 5.3 but using DP into the different DTs.

5.1 EXP 1 - Selection of RF hyper-parameters

As stated in Section 3, there is one important hyper-parameter in DT (splitting rule) and two in RF (number of trees and ensemble method). The results of the combination of the three can be found in Fig. 5 for AD and in Fig. 6 for AC. The first thing to mention is that the differences between the methods are minimal. The biggest difference is 0.7 percentage points in the case of UNSW-NB15 between the worse combination and the best independent of the problem that we are solving.

In the case of AD (Fig. 5), we can observe in the curves that entropy is the best splitting method for KDD, UNSW-NB15 and CIC-IDS-2017, while gini is better in NSL-KDD. With respect to the ensemble method, there are no big differences as the curves with the same splitting rule are crossing all the time. Just two exceptions, gini_WV in KDD and entropy_WV where the results are worse. If AC is considered (Fig. 6), we can see how entropy is clearly better in all the datasets. With respect to the ensemble method, the same as in AD is happening, there is no clear advantage of one method over the other. Finally, with respect to the number of trees, we can mention that there is a clear improvement from 0 to 15-30 (depending on the dataset and problem type) trees, but the performance afterwards does not improve much.

A summary of the best combination of hyper-parameters that is selected for each dataset and problem type is given in Table 3. These values will be used over the rest of the experiments.

5.2 EXP 2 - Evaluation of independent RFs on different clients

In this subsection, we divided the data into the different clients in three different ways: dividing the data according to a specific feature as explained in Table 2 (EXP 2.1), and dividing the data randomly between the different clients with the same number of instances between the clients (EXP 2.2) or dividing it randomly but with the same number of instances as in EXP 2.1 (EXP 2.3).

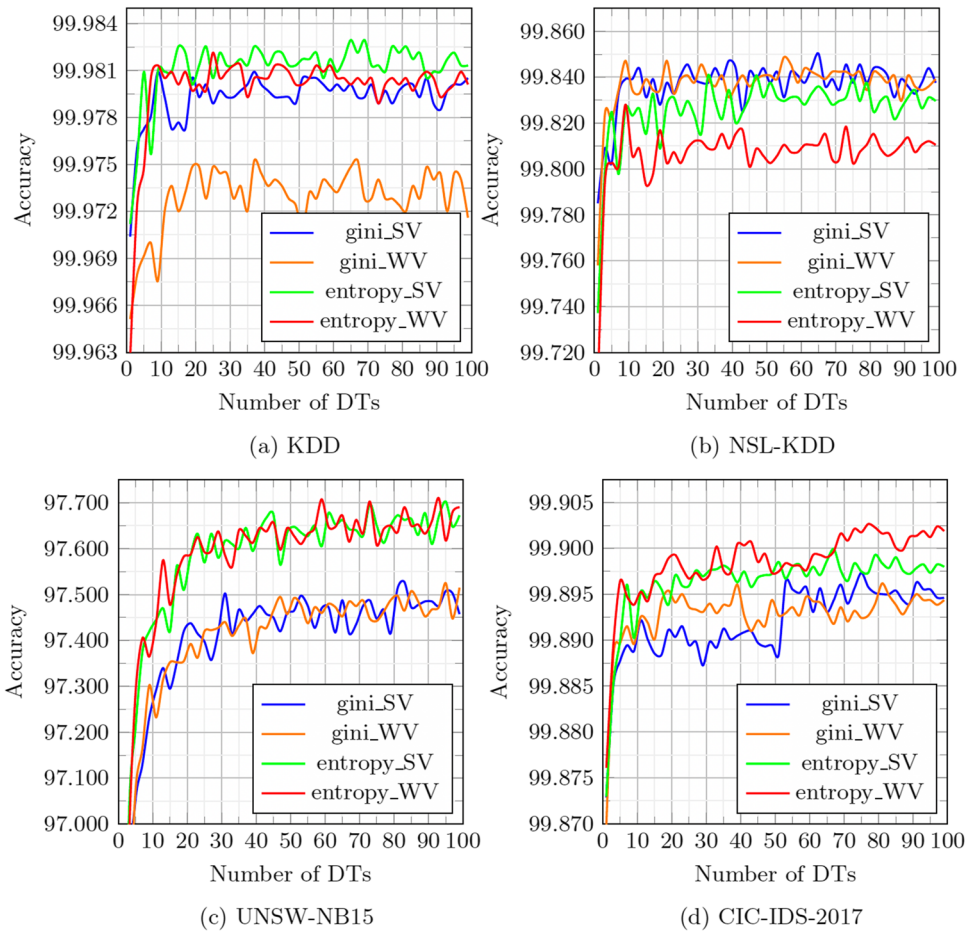

Fig. 5 EXP 1 - Selection of RF hyper-parameters: Accuracy of RF for AD on the validation set in (a) KDD, (b) NSL-KDD, (c) UNSW-NB15, and (d) CIC-IDS-2017, for different combinations of hyper-parameters. Notice that Y-axis range is from minimum to maximum accuracy on the specific dataset

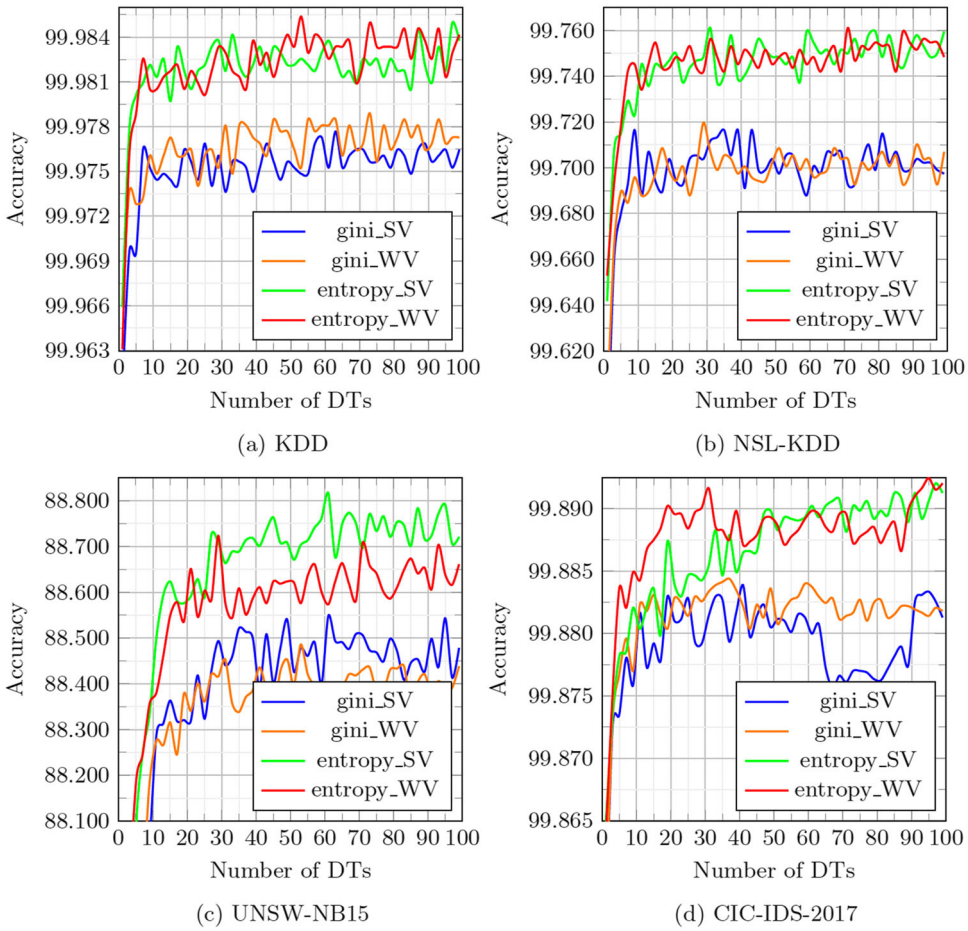

Fig. 6 EXP 1 - Selection of RF hyper-parameters: Accuracy of RF for AC on the validation set in (a) KDD, (b) NSL-KDD, (c) UNSW-NB15, and (d) CIC-IDS-2017, for different combinations of hyper-parameters. Notice that Y-axis range is from minimum to maximum accuracy on the specific dataset

In EXP 2.1, one RF was trained per client using different data as explained above. Then, these RFs were tested into two different types of testing sets: (1) the testing set only contains information from the specific subset, or (2) the testing set contains information independently of the subset. The results of RF in EXP 2.1 can be found in Table 4. It can be observed that, independently of whether it is AD or AC, and independently of which dataset it is, the accuracy of the different RFs is higher if they are tested on the testing data that belong to the same subset as trained than when we test on the entire testing set. This means that there is information that RF is missing and that it will not be able to classify.

The above statement is corroborated by EXP 2.2 and EXP 2.3. In this example, the data is divided randomly, which means that no specific value for a feature is followed to divide the dataset for the specific clients. The performance of independent RFs is shown in Table 5. We can observe how the performance of RF is really high in both experiments for AC and AD and independently of the dataset. This happens because the different clients had access to the whole range of data and did not miss any information. This strengthens our point of creating a global RF in the server, where the information of the different clients is shared without compromising the information by send it to the server through the network.

Table 3 EXP 1 - Selection of RF hyper-parameters: Best combination of hyper-parameters in RF, per dataset

Hyper-parameters		KDD	NSL-KDD	UNSW-NB15	CIC-IDS-2017
AD	No. of trees	65	65	93	77
	Split rule	entropy	gini	entropy	entropy
	Ensemble method	SV	SV	WV	WV
AC	No. of trees	53	31	61	95
	Split rule	entropy	entropy	entropy	entropy
	Ensemble method	WV	SV	SV	WV

Table 4 EXP 2.1 - Evaluation of independent RFs on different clients using a specific feature as a division criteria: Performance of independent RFs for AD and AC on different subsets from KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset

Dataset	Subset	AD				AC			
		On subset		On entire set		On subset		On entire set	
		Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
KDD	icmp	100	100	68.50	71.01	99.98	99.98	72.05	66.97
	tcp	99.97	99.97	42.59	43.70	99.84	99.84	42.35	30.71
	udp	99.96	99.96	25.88	18.23	99.81	99.79	20.00	6.78
NSL-KDD	icmp	99.29	99.29	27.37	26.89	98.75	98.75	48.03	37.12
	tcp	99.89	99.89	92.29	92.21	99.86	99.86	92.36	89.76
	udp	99.83	99.83	83.59	82.65	99.79	99.79	56.85	49.44
UNSW-NB15	–	96.76	96.76	80.63	80.50	83.38	83.48	62.68	55.25
	dns	100	100	76.95	75.22	99.87	99.88	52.89	53.66
	ftp	94.79	94.78	52.69	52.07	92.23	91.60	26.10	25.83
	ftp-data	100	100	44.27	43.59	99.79	99.79	32.76	28.84
	http	98.00	98.00	76.20	75.88	85.72	84.63	36.89	38.08
	smtp	99.95	99.95	46.51	45.47	94.16	93.61	36.38	30.53
CIC-IDS-2017	21	100	100	31.29	35.05	99.77	99.77	71.29	61.95
	22	99.96	99.96	43.02	46.17	99.74	99.74	65.72	59.15
	53	100	100	73.35	64.71	100	100	74.23	63.28
	80	99.88	99.88	94.1	94.32	99.59	99.59	98.70	98.25
	88	100	100	61.81	59.3	99.92	99.92	74.22	63.28
	135	100	100	29.58	28.99	100	100	74.24	63.29
	139	99.94	99.94	60.77	54.83	100	100	67.98	60.36
	389	100	100	65.6	59.45	100	100	72.42	62.42
	443	100	100	74.1	63.21	100	100	74.21	63.26
	445	100	100	68.5	63.1	99.34	99.34	73.78	63.02
	465	99.96	99.96	62.44	56.99	99.96	99.96	74.23	63.29
	1124	99.62	99.62	58.41	58.08	100	100	74.23	63.29
	3268	100	100	74.06	63.19	100	100	74.22	63.28
	8080	79.55	79.58	33.22	37.16	79.84	79.00	73.81	63.11

Table 5 EXP 2.2 and EXP 2.3 - Evaluation of independent RFs on different clients using random generated subsets: Performance of independent RFs for AC on different subsets from KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset

Dataset	EXP 2.2				EXP 2.3			
	AD		AC		AD		AC	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
KDD	99.97	99.97	99.96	99.96	99.97	99.97	99.97	99.97
	99.97	99.97	99.96	99.96	99.97	99.97	99.97	99.97
	99.97	99.97	99.96	99.96	99.93	99.93	99.85	99.85
NSL-KDD	99.70	99.70	99.58	99.58	99.44	99.44	99.25	99.24
	99.71	99.71	99.63	99.63	99.81	99.81	99.74	99.74
	99.70	99.70	99.63	99.63	99.55	99.55	99.42	99.42
UNSW-NB15	96.44	96.44	87.71	87.60	97.30	97.30	88.64	88.73
	96.42	96.42	87.53	87.45	96.73	96.73	88.02	88.01
	96.47	96.47	87.52	87.46	93.71	93.71	84.95	84.71
	96.40	96.40	87.50	87.46	92.99	92.99	84.55	84.18
	96.46	96.46	87.41	87.34	95.88	95.88	87.39	87.28
	96.52	96.52	87.68	87.63	94.01	94.01	85.01	84.66

Table 5 continued

Dataset	EXP 2.2				EXP 2.3			
	AD		AC		AD		AC	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
CIC-IDS-2017	99.81	99.81	99.80	99.79	99.48	99.48	98.14	97.93
	99.81	99.81	99.79	99.79	99.55	99.55	98.94	98.86
	99.81	99.81	99.79	99.79	99.88	99.88	99.86	99.86
	99.81	99.81	99.80	99.79	99.88	99.88	99.86	99.86
	99.81	99.80	99.79	99.79	98.91	98.91	96.62	96.10
	99.81	99.81	99.79	99.78	96.88	96.88	91.68	90.90
	99.81	99.81	99.79	99.79	98.46	98.46	94.83	94.01
	99.81	99.81	99.78	99.78	98.95	98.95	96.24	95.65
	99.82	99.82	99.78	99.78	99.86	99.86	99.84	99.84
	99.81	99.81	99.79	99.78	97.67	97.67	94.06	93.16
	99.81	99.81	99.78	99.78	98.73	98.73	95.54	94.71
	99.81	99.81	99.80	99.79	95.42	95.39	85.75	83.37
	99.81	99.81	99.79	99.78	98.54	98.54	94.88	94.20
	99.81	99.81	99.80	99.79	98.74	98.74	95.17	94.47

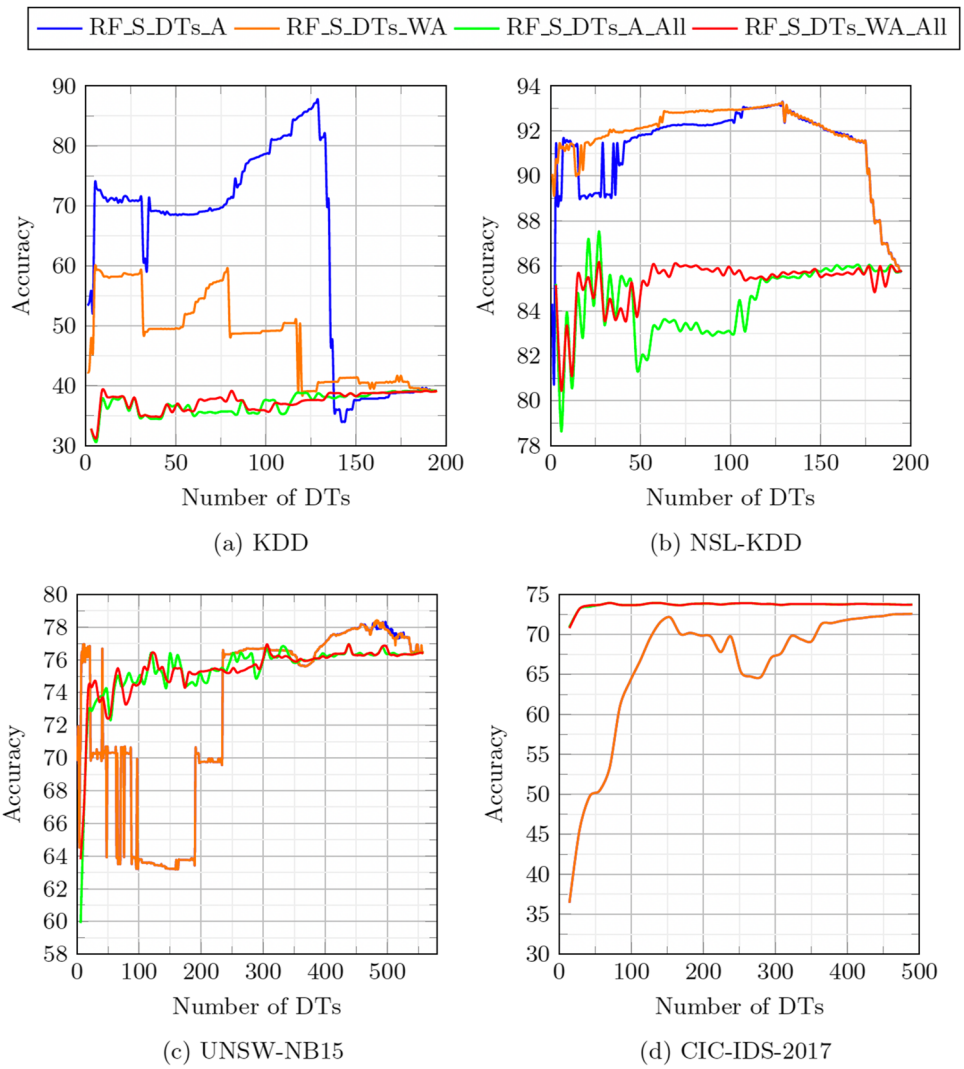

Fig. 7 EXP 3 - Global RF based on Federated Learning: AD accuracy of the global RF on the testing set of (a) KDD, (b) NSL-KDD, (c) UNSW-NB15, and (d) CIC-IDS-2017 using four different merging methods. Notice that Y-axis range is from minimum to maximum accuracy on the specific dataset

5.3 Experiment 3 - Global RF based on federated learning

The goal of this experiment was to find the best combination of the hyper-parameters for the global RF that is built on the server. We evaluated four different merging methods that are explained in Section 3.3 (RF_S_DTs_A, RF_S_DTs_WA, RF_S_DTs_A_All, RF_S_DTs_WA_All) and different number of DTs. The number of DTs that were evaluated includes every number from 1 to MaxDTs for the first two methods. For the remaining two methods we used multiplication of number of clients until we reach the MaxDTs. The only exception is CIC-IDS-2017 datasets were the maximum number of DTs that was evaluated is 500. The performances of global RF were tested for AD and AC in the entire testing set for all four datasets.

Figure 7 presents a comparison of the accuracy of global RF for AD in all four datasets. We can see that the methods that combine all DTs trees together before selecting the best ones for global RF have higher accuracy for three datasets (KDD, NSL-KDD, UNSW-NB15). Only for CIC-IDS-2017 dataset the methods which select the best DTs from each RF have better performance. When the sorting measurement is considered, there is not a big difference between A and WA, except for KDD, where using A gives a considerable improvement (around 30 percentage points).

Figures 8 and 9 show the comparison of the accuracy and F1 score of the global RF for AC in all four datasets. For KDD we can notice that RF_S_DTs_WA_All has considerably better results than other tree methods if we are using lower number of DTs. An interesting observation for NSL-KDD is that RF_S_DTs_A_All has in general better performances than RF_S_DTs_WA_All, but RF_S_DTs_WA_All has one peek when it outperforms all the others and it is better than RF_S_DTs_A_All for around 5 percentage points. For UNSW-NB15, RF_S_DTs_WA_All outperforms all the others, while for CIC-IDS-2017 all the methods have the similar accuracy. When it comes to F1 score, we can notice that it is

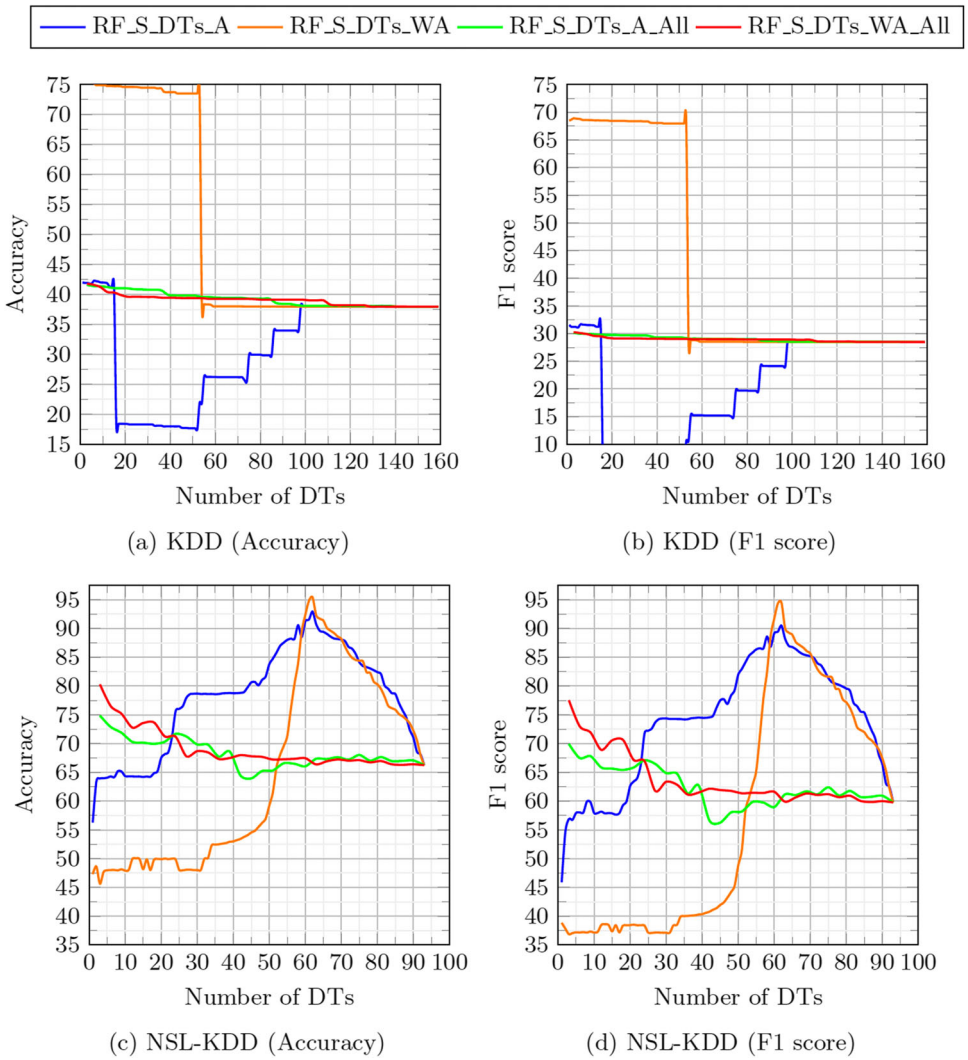

Fig. 8 EXP 3 - Global RF based on Federated Learning: AC accuracy and F1 score of the global RF on the testing set of (a)(b) KDD and (c)(d) NSL-KDD using four different merging methods. Notice that Y-axis range is from minimum to maximum value of both accuracy and F1 score on the specific dataset

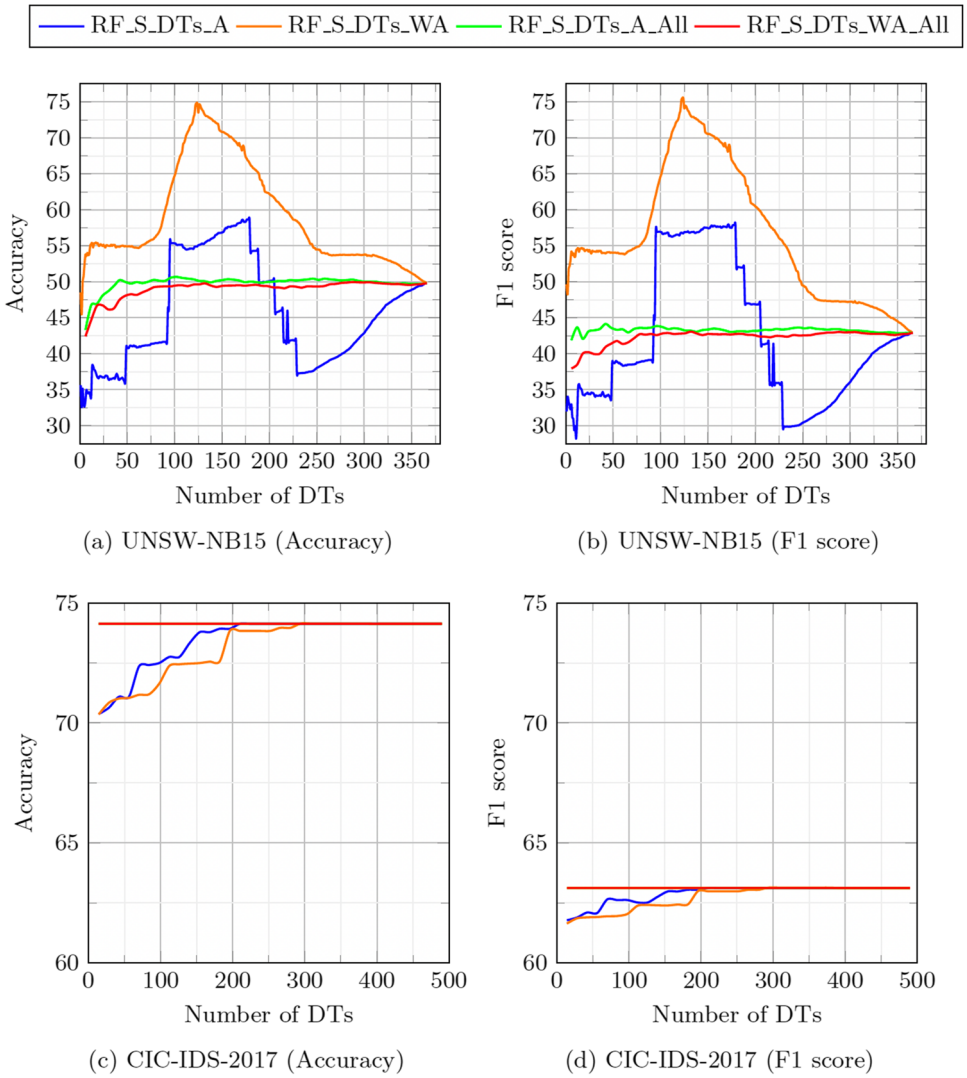

Fig. 9 EXP 3 - Global RF based on Federated Learning: AC accuracy and F1 score of the global RF on the testing set of (a)(b) UNSW-NB15 and (c)(d) CIC-IDS-2017 using four different merging methods. Notice that Y-axis range is from minimum to maximum value of both accuracy and F1 score on the specific dataset

not considerably lower than accuracy in any dataset, except CIC-IDS-2017 where the difference is around 10 percentage points.

The best combination of the number of DTs in global RF and the merging method per dataset, as well as the accuracy and F1 score that were achieved using this combination, are given in Table 6 for AD and Table 7 for AC. If more than one combination resulted with the same accuracy, the following criteria were applied to select the best one:

1. the one with the highest F1 score was selected
2. if F1 score is also the same the one that achieved those performances with the least number of DTs is selected
3. if the number of DTs is also the same, the fastest method is selected.

The global RFs were compared with the performances of independent RFs on the entire testing set and the results are presented in Table 8 for AD and in Table 9 for AC. As a performance measure for the independent RFs we use the maximum, average, and minimum accuracy of all independent RFs. For AD can see how the global RF improves the

Table 6 EXP 3 - Global RF based on Federated Learning: The best combination of parameters for global RF for AD for KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 dataset

Dataset	No. of DTs	Merging method	Accuracy	F1 score
KDD	129	RF_S_DTs_A_All	87.511	87.861
NSL-KDD	129	RF_S_DTs_A_All	93.28	93.216
UNSW-NB15	483	RF_S_DTs_WA_All	78.426	77.301
CIC-IDS-2017	70	RF_S_DTs_A	73.543	63.486

Table 7 EXP 3 - Global RF based on Federated Learning: The best combination of parameters for global RF for AC for KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 dataset

Dataset	No. of DTs	Merging method	Accuracy	F1 score
KDD	3	RF_S_DTs_WA_All	75.847	68.919
NSL-KDD	62	RF_S_DTs_WA_All	95.433	94.481
UNSW-NB15	123	RF_S_DTs_WA_All	74.811	75.522
CIC-IDS-2017	14	RF_S_DTs_A	74.133	63.121

Table 8 EXP 3 - Global RF based on Federated Learning: Comparison of maximum, minimum, and average accuracy of independent RFs against the accuracy of global RF on the entire testing set of KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset for AD

Dataset	No. of Clients	Independent RFs			Global RF
		Max	Min	Avg	
KDD	3	68.497	25.885	45.658	**87.511**
NSL-KDD	3	92.289	27.367	67.750	**93.28**
UNSW-NB15	6	**80.629**	44.273	62.876	78.426
CIC-IDS-2017	14	**94.103**	29.583	59.305	73.543

The best option for each dataset is shown in boldface

Table 9 EXP 3 - Global RF based on Federated Learning: Comparison of maximum, minimum, and average accuracy of independent RFs against the accuracy of global RF on the entire testing set of KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset for AC

Dataset	No. of Clients	Independent RFs			Global RF
		Max	Min	Avg	
KDD	3	72.049	20.003	44.801	**75.847**
NSL-KDD	3	92.364	48.026	65.746	**95.433**
UNSW-NB15	6	62.679	26.102	41.284	**74.811**
CIC-IDS-2017	14	**98.701**	65.722	74.521	74.133

The best option for each dataset is shown in boldface

Table 10 EXP 3 - Global RF based on Federated Learning: Comparison of maximum, minimum, and average F1 score of independent RFs against the F1 score of global RF on the entire testing set of KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset for AC

Dataset	No. of Clients	Independent RFs			Global RF
		Max	Min	Avg	
KDD	3	66.97	6.78	34.82	**68.92**
NSL-KDD	3	89.76	37.12	58.77	**94.48**
UNSW-NB15	6	55.25	25.83	38.70	**75.52**
CIC-IDS-2017	14	**98.25**	59.15	65.09	63.12

The best option for each dataset is shown in boldface

Table 11 EXP 4 - Global RF with differential privacy based on Federated Learning: Comparison of maximum, minimum, and average accuracy of independent RFs against the accuracy of global RF, both options with DP, on the entire testing set of KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset for AD

Dataset	No. of clients	ϵ	Independent RFs with DP			Global RF with DP
			Max	Min	Avg	
KDD	3	0.1	80.335	19.710	60.112	**92.402**
		0.5	82.134	19.676	60.711	**93.272**
		1	84.762	19.759	61.587	**93.205**
		5	82.109	19.733	60.703	**90.002**
NSL-KDD	3	0.1	62.676	46.392	54.225	**85.698**
		0.5	65.176	46.339	55.059	**79.444**
		1	61.932	46.406	53.977	**85.477**
		5	76.926	46.459	58.975	**79.135**
UNSW-NB15	6	0.1	53.938	40.593	50.233	**70.281**
		0.5	53.922	46.078	50.596	**71.140**
		1	53.937	46.063	51.083	**70.672**
		5	53.742	46.258	50.257	**70.079**
CIC-IDS-2017	14	0.1	**74.131**	25.869	61.921	**74.131**
		0.5	**74.134**	25.932	63.086	74.068
		1	**74.106**	25.894	62.459	**74.106**
		5	74.104	25.896	60.646	**74.109**

The best option for each combination of dataset and hyper-parameter ϵ is shown in boldface

maximum accuracy of individual RFs for KDD and NSL-KDD, and it is very close for UNSW-NB15. For CIC-IDS-2017 it fell behind the maximum, but it performed better than the average accuracy. Also, for AC can see how the global RF improves the maximum accuracy of individual RFs for three out of four datasets, only for CIC-IDS-2017 it fell behind the maximum, but it is very close to the average. The same can be concluded if F1 score is considered (Table 10).

5.4 Experiment 4 - Global RF with differential privacy based on federated learning

In this section, we test how our proposed setup is affected by adding DP to the different DTs. The results can be found in Tables 11 and 12, for AD and AC, respectively. Firstly, we test how the hyper-parameter ϵ affects the performance of the algorithm. We can say that with the tested values, ϵ does not have a big impact on the results, except for NSL-KDD where higher differences can be noticed for both AD and AC.

Secondly, if we combine the trees into a global RF, we can see how the performance of the global RF is better than the performance of the best RF in the clients, except for CIC-IDS-2017 and NSL-KDD in AC. However, the performance is close. In addition, for CIC-IDS-2017 in AD the results of the best independent RF and the global one are the same or very similar. If F1 score is considered for AC (Table 13) the same conclusions can be found except for KDD where the performance of global RF with DP is not better than the maximum, but it is close.

If instead of comparing to the best performance, we compare it with the average, in all cases the global RF is better than the individual ones except for CIC-IDS-2017 in AC where the results are very close to each other.

Lastly, if we compare the results without DP (Tables 8 and 9) and with DP (Tables 11 and 12), we can see how adding DP decreases the results of RF for all the datasets in AC. On the case of AD, we can see how the same is happening in NSL-KDD and UNSW-NB15, while in KDD and CIC-IDS-2017, RF improves the performance when DP is added. These two datasets are the ones with more instances which can be an indication that adding noise will result in a more general tree, which can be useful in this case.

6 Conclusion

This paper extends the evaluation of the previously proposed federated learning framework based on random forest by adding differential privacy into random forest, as well as performing experiments for both attack detection and attack classification. The experiments were conducted on four well-know intrusion detection datasets: KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017.

The results have shown that combining independent RFs into a global one on the server outperforms the average accuracy of the RFs on the clients for both AD and AC. Additionally, it is concluded that adding differential privacy to random forest penalizes the performance to a major extent in some cases. However, if we compare a global random

Table 12 EXP 4 - Global RF with differential privacy based on Federated Learning: Comparison of maximum, minimum, and average accuracy of independent RFs against the accuracy of global RF, both options with DP, on the entire testing set of KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset for AC

Dataset	No. of clients	ϵ	Independent RFs with DP			Global RF with DP
			Max	Min	Avg	
KDD	3	0.1	57.240	4.452	27.706	**62.161**
		0.5	57.505	8.228	29.053	**61.508**
		1	57.318	4.425	27.761	**70.689**
		5	**59.682**	4.505	28.653	57.409
NSL-KDD	3	0.1	**59.610**	3.539	38.964	55.438
		0.5	**60.526**	3.452	39.205	57.346
		1	**62.348**	3.428	39.817	55.295
		5	**61.229**	3.155	39.326	54.607
UNSW-NB15	6	0.1	46.251	13.219	35.290	**65.301**
		0.5	46.021	13.312	36.839	**64.994**
		1	45.908	13.673	36.794	**63.225**
		5	45.824	13.433	36.755	**65.584**
CIC-IDS-2017	14	0.1	**74.728**	74.092	74.138	74.092
		0.5	**75.149**	74.109	74.186	74.109
		1	**74.939**	74.011	74.081	74.011
		5	**74.453**	74.133	74.162	74.137

The best option for each combination of dataset and hyper-parameter ϵ is shown in boldface

forest on the server with the independent random forests on the clients the accuracy can be improved even when using differential privacy.

The proposed framework is recommended in the applications where the data cannot be centralized and the goal is to apply AI, while protecting the data as much as possible. It is also proved that the proposed framework is beneficial in the cases where the model can be attacked or an unauthorized access to the model can happen, and differential privacy has to be implemented as an additional protection mechanism to prevent the extraction of the data from the model. An example of such application is AI-based healthcare solutions that use patients' personal medical data to identify global outbreaks of emerging pandemics. If anonymity of local models can

Table 13 EXP 4 - Global RF with differential privacy based on Federated Learning: Comparison of maximum, minimum, and average F1 score of independent RFs against the F1 score of global RF, both options with DP, on the entire testing set of KDD, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 dataset for AC

Dataset	No. of clients	ϵ	Independent RFs with DP			Global RF with DP
			Max	Min	Avg	
KDD	3	0.1	**56.95**	6.57	25.712	56.33
		0.5	**56.98**	9.15	25.40	54.06
		1	56.89	6.64	25.17	**64.83**
		5	**59.75**	5.69	25.36	46.76
NSL-KDD	3	0.1	**47.29**	0.62	28.49	43.36
		0.5	**48.44**	0.83	28.91	44.34
		1	**50.78**	0.73	29.66	44.26
		5	**49.75**	0.51	29.22	42.55
UNSW-NB15	6	0.1	29.35	3.12	19.58	**53.75**
		0.5	29.01	3.13	19.84	**53.46**
		1	28.89	3.29	19.77	**51.90**
		5	28.80	3.18	19.75	**54.13**
CIC-IDS-2017	14	0.1	**64.84**	63.07	63.19	63.07
		0.5	**65.64**	63.09	63.28	63.09
		1	**65.38**	62.96	63.14	62.96
		5	**65.06**	63.12	63.27	63.13

The best option for each combination of dataset and hyper-parameter ϵ is shown in boldface

be ensured, more individuals and regions might be willing to share their data, which can greatly support faster diagnosis, detection, and controlling the spread of such diseases. Collaborative manufacturing, smart cities and intelligent system of systems from multiple (even mutually competing) vendors also demand privacy preservation and selective sharing of local models.

The main challenge in practical implementations lies in minimizing the overheads associated with differential privacy. Communication overhead and scalability issues may also arise, particularly in scenarios involving a large number of participating entities with complex internal organisation and demanding data privacy policies. Additionally, limitation of computational resources of the entities can influence the selection of hyperparameters, which in turn affects the model performance and hence the feasibility and applicability in a given context.

As the future work we plan to evaluate the proposed framework in different real-world scenarios where decentralized learning is required. This evaluation will provide insights into its practical applicability and scalability. Additionally, we aim to extend the framework to include support for vertical federated learning. This extension will enhance framework's applicability in scenarios where different feature spaces are used across different entities. Furthermore, we plan to add a combination of vertical and horizontal approach to address data access limitations, ensuring its applicability in scenarios with diverse data privacy concerns. Additionally, we aim to evaluate our framework using actual devices to measure memory requirements, response time, and other performance metrics, providing insights into its practical effectiveness and areas for optimisation.

Acknowledgements This work has been partially supported by the H2020 ECSEL EU Project Intelligent Secure Trustable Things (InSecTT) and Distributed Artificial Intelligent System (DAIS). InSecTT (www.insectt.eu) has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876038 and DAIS (https://dais-project.eu/) has received funding from the ECSEL JU under grant agreement No 101007273. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, Netherlands, Turkey.

The document reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains.

Author Contributions Tijana Markovic and Miguel Leon have equal contributions to the paper. They provided the main idea of the paper, wrote most of the text, wrote the code for most of the experiments, and performed all the experiments. David Buffoni participated in the paper idea, contributed with part of the code, and wrote small parts of the text. Sasikumar Punnekkat participated in the paper idea, wrote small parts of the text, and revised the whole manuscript.

Funding Open access funding provided by Mälardalen University. This work has been partially supported by the H2020 ECSEL EU Project Intelligent Secure Trustable Things (InSecTT) and Distributed Artificial Intelligent System (DAIS). InSecTT (www.insectt.eu) has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876038 and DAIS (https://dais-project.eu/) has received funding from the ECSEL JU under grant agreement No 101007273.

Availability of data and materials All data is publicly available and the corresponding references are given in Table 1.

Code Availability The code is publicly available.

Declarations

Consent to participate I, Tijana Markovic consent to participate in this study. I, Miguel Leon consent to participate in this study. I, David Buffoni consent to participate in this study. I, Sasikumar Punnekkat consent to participate in this study.

Consent for publication I, Tijana Markovic consent to share my data and image for publication. I, Miguel Leon consent to share my data and image for publication. I, David Buffoni consent to share my data and image for publication. I, Sasikumar Punnekkat consent to share my data and image for publication.

References

1. AlDairi A, Tawalbeh L (2017) Cyber Security Attacks on Smart Cities and Associated Mobile Technologies. Proc Comput Sci 109:1086–1091. https://doi.org/10.1016/j.procs.2017.05.391. 8th International conference on ambient systems, networks and technologies, ANT-2017 and the 7th International conference on sustainable energy information technology, SEIT 2017, 16-19 May 2017, Madeira, Portugal
2. Ma C (2021) Smart city and cyber-security; technologies used, leading challenges and future recommendations. Energy Rep 7:7999–8012. https://doi.org/10.1016/j.egyr.2021.08.124
3. Liao H-J, Lin C-HR, Lin Y-C, Tung K-Y (2013) Intrusion detection system: A comprehensive review. J Netw Comput Appl 36(1):16–24
4. Bace R, Mell P (2001) Intrusion detection systems. National Institute of Standards and Technology (NIST), Technical Report 800-31
5. Ahmad Z, Shahid Khan A, Wai Shiang C, Abdullah J, Ahmad F (2021) Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Trans Emerg Telecommun Technol 32(1):4150
6. Buczak AL, Guven E (2015) A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun Surv Tutor 18(2):1153–1176

7. Resende PAA, Drummond AC (2018) A survey of random forest based methods for intrusion detection systems. ACM Comput Surv (CSUR) 51(3):1–36
8. Revathi S, Malathi A (2013) A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. Int J Eng Res Technol (IJERT) 2(12):1848–1853
9. Abedin M, Siddiquee KNEA, Bhuyan M, Karim R, Hossain MS, Andersson K et al (2018) Performance analysis of anomaly based network intrusion detection systems. In: 43nd IEEE conference on local computer networks workshops (LCN Workshops), Chicago, October 1-4, 2018, pp 1–7. IEEE Computer Society
10. Farnaaz N, Jabbar M (2016) Random forest modeling for network intrusion detection system. Proc Comput Sci 89:213–217
11. Hautsalo J (2021) Using supervised learning and data fusion to detect network attacks. https://mdh.diva-portal.org/smash/record.jsf?pid=diva2:1569348
12. Leon M, Markovic T, Punnekkat S (2022) Comparative evaluation of machine learning algorithms for network intrusion detection and attack classification. In: International joint conference on neural networks (IJCNN). IEEE
13. Li Q, Wen Z, Wu Z, Hu S, Wang N, Li Y, Liu X, He B (2021) A survey on federated learning systems: vision, hype and reality for data privacy and protection. IEEE Trans Knowl Data Eng 1–1
14. Kairouz P, McMahan HB, Avent B, Bellet A, Bennis M, Bhagoji AN, Bonawitz K, Charles Z, Cormode G, Cummings R et al (2021) Advances and open problems in federated learning. Found Trends Mach Learn 14(1–2):1–210
15. Agrawal S, Sarkar S, Aouedi O, Yenduri G, Piamrat K, Alazab M, Bhattacharya S, Maddikunta PKR, Gadekallu TR (2022) Federated learning for intrusion detection system: Concepts, challenges and future directions. Comput Commun
16. Campos EM, Saura PF, González-Vidal A, Hernández-Ramos JL, Bernabé JB, Baldini G, Skarmeta A (2022) Evaluating federated learning for intrusion detection in internet of things: Review and challenges. Comput Netw 203:108661
17. Fletcher S, Islam MZ (2017) Differentially private random decision forests using smooth sensitivity. Expert Syst Appl 78:16–31
18. Markovic T, Leon M, Buffoni D, Punnekkat S (2022) Random forest based on federated learning for intrusion detection. In: IFIP international conference on artificial intelligence applications and Innovations, pp 132–144. Springer
19. McMahan HB, Moore E, Ramage D, Hampson S, Arcas BA (2016) Communication-efficient learning of deep networks from decentralized data. In: International conference on artificial intelligence and statistics
20. Taheri R, Shojafar M, Alazab M, Tafazolli R (2020) Fed-IIoT: A robust federated malware detection architecture in industrial IoT. IEEE Trans Ind Inform 17:8442–8452
21. Li L, Fan Y, Tse M, Lin K-Y (2020) A review of applications in federated learning. Comput Ind Eng 149:106854
22. Wen J, Zhang Z, Lan Y, Cui Z, Cai J, Zhang W (2023) A survey on federated learning: challenges and applications. Int J Mach Learn Cybern 14(2):513–535
23. Zhang C, Xie Y, Bai H, Yu B, Li W, Gao Y (2021) A survey on federated learning. Knowl-Based Syst 216:106775
24. Lavaur L, Pahl M-O, Busnel Y, Autrel F (2022) The Evolution of Federated Learning-Based Intrusion Detection and Mitigation: A Survey. IEEE Trans Netw Service Manag 19:2309–2332
25. Qin Y, Kondo M (2021) Federated learning-based network intrusion detection with a feature Selection Approach. 2021 International conference on electrical, communication, and computer engineering (ICECCE), pp 1–6
26. Fu Y, Du Y, Cao Z, Li Q, Xiang W (2022) A deep learning model for network intrusion detection with imbalanced data. electronics
27. Man D, Zeng F, Yang W, Yu M, Lv J, Wang Y (2021) Intelligent Intrusion Detection Based on Federated Learning for Edge-Assisted Internet of Things. Secur Commun Netw 2021:9361348. https://doi.org/10.1155/2021/9361348
28. Chen J, Zhao Y, Li Q, Feng X, Xu K (2022) FedDef: defense against gradient leakage in federated learning-based network intrusion detection systems
29. Chen Z, Lv N, Liu P, Fang Y, Chen K-S, Pan W (2020) Intrusion Detection for Wireless Edge Networks Based on Federated Learning. IEEE Access 8:217463–217472
30. Li Q, Wen Z, He B (2020) Practical federated gradient boosting decision trees. Proc AAAI Conf Artif Intell 34:4642–4649
31. Dong T, Li S, Qiu H, Lu J (2022) An interpretable federated learning-based network intrusion detection framework
32. Gencturk M, Sinaci AA, Cicekli NK (2022) BOFRF: A Novel Boosting-Based Federated Random Forest Algorithm on Horizontally Partitioned Data. IEEE Access 10:89835–89851. https://doi.org/10.1109/ACCESS.2022.3202008
33. Hauschild A-C, Lemanczyk M, Matschinske J, Frisch T, Zolotareva O, Holzinger A, Baumbach J, Heider D (2022) Federated Random Forests can improve local performance of predictive models for various healthcare applications. Bioinformatics 38(8):2278–2286. https://doi.org/10.1093/bioinformatics/btac065. https://academic.oup.com/bioinformatics/article-pdf/38/8/2278/49009424/btac065.pdf
34. Wei K, Li J, Ding M, Ma C, Yang HH, Farokhi F, Jin S, Quek TQS, Vincent Poor H (2020) Federated Learning With Differential Privacy: Algorithms and Performance Analysis. IEEE Trans Inf Forensics Secur 15:3454–3469. https://doi.org/10.1109/TIFS.2020.2988575
35. Sweeney L (2002) k-Anonymity: A Model for Protecting Privacy. Int J Uncertain Fuzziness Knowl Based Syst 10:557–570
36. Gentry C (2009) A fully homomorphic encryption scheme. Stanford University, ???
37. Dwork C (2006) Differential privacy. In: Encyclopedia of cryptography and security
38. Szücs G (2013) Random Response Forest for Privacy-Preserving Classification. J Comput Eng 2013:397096–13970966
39. Kwatra S, Torra V (2022) A k-anonymised federated learning framework with decision trees. In: Garcia-Alfaro J, Muñoz-Tapia JL, Navarro-Arribas G, Soriano M (eds) Data Privacy Management, Cryptocurrencies and Blockchain Technology. Springer, Cham, pp 106–120
40. Liu Y, Liu Y, Liu Z, Liang Y, Meng C, Zhang J, Zheng Y (2020) Federated forest. IEEE Trans Big Data 8(3):843–854
41. Souza LAC, Antonio F. Rebello G, Camilo GF, Guimarães LCB, Duarte OCMB (2020) Dfedforest: Decentralized federated forest. In: 2020 IEEE international conference on blockchain (Blockchain), pp 90–97
42. Maddock S, Cormode G, Wang T, Maple C, Jha S (2022) Federated boosted decision trees with differential privacy. Proceedings of the 2022 ACM SIGSAC conference on computer and communications security
43. Geyer RC, Klein T, Nabi M (2017) Differentially private federated learning: a client level perspective. arXiv:1712.07557
44. Sarwate AD, Chaudhuri K (2013) Signal processing and machine learning with differential privacy. IEEE Signal Process Mag 30(5)
45. Fletcher S, Islam MZ (2019) Decision tree classification with differential privacy: A survey. ACM Comput Surv (CSUR) 52(4):1-33
46. Patil A, Singh S (2014) Differential private random forest. 2014 International conference on advances in computing, communications and informatics (ICACCI), pp 2623–2630
47. Fletcher S, Islam MZ (2015) A differentially private decision forest. In: Australasian data mining conference
48. Fletcher S, Islam MZ (2015) A differentially private random decision forest using reliable signal-to-noise ratios. In: Australasian conference on artificial intelligence

49. Vos D, Vos J, Li T, Erkin Z, Verwer S (2023) Differentially-private decision trees and provable robustness to data poisoning
50. Sun D, Li N, Yang S, Du Q (2021) A decision tree based on differential privacy. In: 2021 IEEE 5th Information technology,networking,electronic and automation control conference (ITNEC), vol 5, pp 445–453. https://doi.org/10.1109/ITNEC52019.2021.9587254
51. Li X, Qin B, Luo Y, Zheng D (2022) A differential privacy budget allocation algorithm based on out-of-bag estimation in random forest. Mathematics 10(22)
52. Li Y, Feng Y, Qian Q (2023) Fdpboost: Federated differential privacy gradient boosting decision trees. J Inf Secur Appl 74:103468. https://doi.org/10.1016/j.jisa.2023.103468
53. Xia G, Chen J, Yu C, Ma J (2023) Poisoning Attacks in Federated Learning: A Survey. IEEE Access 11:10708–10722. https://doi.org/10.1109/ACCESS.2023.3238823
54. Kingsford C, Salzberg SL (2008) What are decision trees? Nat Biotechnol 26(9):1011–1013
55. Quinlan JR (1990) Decision trees and decision-making. IEEE Tran Syst Man Cybern 20(2):339–346
56. Charbuty B, Abdulazeez A (2021) Classification based on decision tree algorithm for machine learning. J Appl Sci Technol Trends 2(01):20–28
57. Quinlan JR et al (1992) Learning with continuous classes. In: 5th Australian joint conference on artificial intelligence, vol 92, pp 343–348. World Scientific
58. Zambon M, Lawrence R, Bunn A, Powell S (2006) Effect of alternative splitting rules on image processing using classification tree analysis. Photogramm Eng Remote Sens 72(1):25–30
59. Dwork C, McSherry F, Nissim K, Smith AD (2006) Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography conference
60. Rana S, Gupta S, Venkatesh S (2015) Differentially private random forest with high utility. 2015 IEEE international conference on data mining, pp 955–960
61. McSherry F, Talwar K (2007) Mechanism design via differential privacy. 48th Annual IEEE symposium on foundations of computer science (FOCS'07), pp 94–103
62. Nissim K, Raskhodnikova S, Smith AD (2007) Smooth sensitivity and sampling in private data analysis. In: Symposium on the theory of computing
63. Yang Q, Liu Y, Cheng Y, Kang Y, Chen T, Yu H (2019) Federated learning. Synth Lect Artif Intell Mach Learn 13(3):1–207
64. Hettich S, Bay SD (1999) The UCI KDD archive. [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science
65. (2009) NSL-KDD. [https://www.unb.ca/cic/datasets/nsl.html]
66. Moustafa N, Slay J (2015) UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: Military communications and information systems conference (MilCIS). IEEE
67. Sharafaldin I, Lashkari AH, Ghorbani AA (2018) Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp 1:108–116
68. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine Learning in Python. J Mach Learn Res 12:2825–2830
69. Holohan N, Braghin S, Mac Aonghusa P, Levacher K (2019) Diffprivlib: the IBM differential privacy library. arXiv:1907.02444. [cs.CR]
70. Hossin M, Sulaiman MN (2015) A review on evaluation metrics for data classification evaluations. Int J Data Mining Knowl Manag Process 5(2):1

Tijana Markovic received the B.S. degree in software engineering from University Mediterranean (Montenegro) in 2012, and the M.S. and the Ph.D. in software engineering from University of Belgrade (Serbia), in 2015 and 2018, respectively. Since 2021, she is employed as postdoctoral researcher at Mälardalen University (Sweden). Her research covers machine learning and its applications in different domains.

Miguel Leon received the B.S. and M.S. degrees in computer science from Granada University (Spain) in 2011 and 2013 respectively, and the Ph.D. with distinction in computer science from Mälardalen University (Sweden), in 2019. Since 2020, he is employed as senior lecturer in Artificial Intelligence at Mälardalen University (Sweden). His research covers various aspects of computational intelligence, including machine learning and data analytics, evolutionary algorithms, multi-sensor data fusion, as well as their applications in the industrial and medical domains. He has been program committee member for a number of conferences and invited referee for many leading international journals.

David Buffoni received the M.S. and Ph.D. degrees in computer science from Pierre et Marie Curie Sorbonne University (France) in 2007, and 2012 respectively. In 2014, he started a position of Data Scientist at Tietoevry where he was involved in several European Research projects such as DAIS and InSecTT. In 2023, he joined Mölnlycke Healthcare AB as an AI/ML Applied Research where his research focus are on Artificial Intelligence and Machine Learning applied to the Healthcare industry.

Sasikumar Punnekkat received M.Tech (Hons) in Computer Science from the Indian Statistical Institute in 1984 and started his career as a Scientist Engineer at the Indian Space Research a Organisation and made significant contributions to the software development and testing of Satellite Launch Vehicles. He was recipient of the Commonwealth scholarship of UK and received D.Phil in computer Science from the University of York in 1997 for his thesis on Fault-tolerant scheduling of real-time systems. He continued at ISRO and was the head of software testing and reliability till 2004 when he joined Mälardalen University, Sweden where he holds a chair in dependable software engineering since 2007. He was also the Director of BITS-Goa campus in India during 2015-16. His research interests include multiple aspects of Real-time Systems, Dependability, and Software Engineering. Dr Sasikumar Punnekkat has over 160 research publications in international conferences and journals (including 5 best paper awards). He had been member of several Program committees and has played a lead role in several EU and national projects such as DAIS, InSecTT, SafeCer, SafeCoP, SUCCESS, EuroWeb, EURECA, FORA, Retnet, Progress and Synopsis.