# Federated Learning with Differential Privacy for Predicting Personal Information and Ad-Request


**Patel Jaykumar Girishbhai**
**202051136**

Faculty Advisor: Dr. Antriksh Goswami
NIT Patna
Mentor Signature :

# Declaration

I, **Patel Jaykumar**, declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referred to the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated any idea/data/fact/source in my submission. I fully understand that any violation of the above will invite disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained.

**Date:14th May,2024** **Place:NIT Patna**

# Abstract

In an era characterized by extensive data accumulation, the preservation of personally identifiable information (PII) and the mitigation of data leakage risks are of utmost importance. This research project extends the work of the FedPacket initiative, focusing on strengthening data security through the application of differential privacy. The integration of Federated Learning represents a proactive approach towards addressing data privacy concerns. By enhancing methods such as feature extraction, key filtration, and client-server model, the study aims to categorize packets, specifically those containing PII or Ad Requests, thereby promoting privacy-preserving machine learning practices. A significant advancement in this research is the incorporation of Differential Privacy, which fortifies the privacy preservation measures. Guided by ethical data handling principles and a commitment to responsible data management, this research utilizes datasets from NoMoAds [1] and AntShield [2]. This study provides practical insights and theoretical underpinnings to enhance data security in technology, thereby fostering a more secure and privacy-conscious digital ecosystem.Our simulation results using the two datasets shows that key filtration increase f1-scire and reduces resource requirement. It also shows relation between parameters of Differential privacy ($\varepsilon$, $\delta$) and F1-Score of resulting model.

# Contents

# Introduction

In this age of technology, data is the weapon, many applications used by the population regularly collect PII (personally identifiable information), sensory data, and user activity and send it to their servers[3]. This data is also used to serve ads to consumers [4][5]. This data is connected to the client's unique advertisement ID and Apps display ads using this unique ID as discussed in this work [6].

In this work, we focus on network-based approaches that trains model on packets transmitted from devices to detect PII and ad-request using Federated learning with Differential privacy. This can then be used to alter or block the packets, notify the user, or monitor the app.

Prior Solutions proposed to predict PII [3][2][7] and ad-request [1] in outgoing packets using features extracted from HTTP requests.Of which AntShield [2], ReCon [7] and No-Mo-Ads [1] trains a model using Centralized approch[3]. Centralized approach: The client sends data(that may consist of PII.) to a centralised server, and the server trains the model. This approach compromises the privacy of the Clients. To mitigate that Federated Learning(FL) can be used.

**Federated Learning** is a method to train machine learning models in a distributed manner. Local models are trained on the client's device and then collected and aggregated on the server. This way client doesn't have to send data to the server to participate in the model training process.

In FedPacket, as well as in our research, Federated learning is used with Support Vector Machine (SVM) as the primary classifier model because of its ability to easily integrate locally trained SVM models into a global model[3].

**Support Vector Machine (SVM)** is a supervised machine learning algorithm used for classification and regression. When training SVM it finds a hyperplane separating different classes in feature space. This hyperplane is then used to predict the class of data points. The dimension of the hyperplane separating classes depends upon the number of classes. For example, for 2 features a line can divide

data points into two classes,3 Features we require a 2-D hyperplane.

But even after using FL, Servers can still infer data by observing gradient updates, analyzing them, and storing the information for future use. Like in they were able to extract images used in training data from the model[8]. In this work, this deferential privacy is implemented.

**Deferential privacy** is an approach to share data about population while keeping individual's data private.It is implemented such even if a individual participates in a research or not it doesn't affect the outcome of the research.To achive this artificial noise is introduced in data before training model.

The privacy guarantee of differential privacy is formalized as follows:

$$\Pr[\mathscr{K}(D) \in S] \leq e^{\varepsilon} \times \Pr[\mathscr{K}(D') \in S] + \delta \tag{1.1}$$

where,

$\mathscr{K}$ is the randomized algorithm,

$D$ and $D'$ are datasets differing in at most one element,

$S$ is any subset of possible outputs, and

$\delta$ is a small positive number in the case of $(\varepsilon, \delta)$-differential privacy.

**Sensitivity** $\Delta s$ The sensitivity of a function $f$, denoted by $\Delta f$, is the maximum difference in the output of $f$ when any single individual's data is added or removed from the dataset.

**Privacy Budget** $\varepsilon$ The Privacy Budget controls the trade-off between privacy and accuracy. A smaller $\varepsilon$ provides stronger privacy guarantees but less accurate results.

**Delta** $\delta$ Delta represents the additional privacy parameter in ($\varepsilon$, $\delta$)-differential privacy. It introduces a small probability that the privacy guarantee might not hold. A lower delta signifies a stricter privacy assurance, reducing the likelihood of privacy breaches in differential privacy mechanisms.

# Literature Review

For this research, we have mainly referred to 3 research papers **Fedpacket[3]:** They have researched the Federated learning approach to mobile packet classification in this research. Our research is mainly based on this paper with the addition of differential privacy.

Training: This study employs Federated Learning to train SVM model, which classifies into two categories: PII exposure and requests within separate packets.

Datasets: The model is trained using 3 real-world datasets labeled for the PII exposure and Ad Request NoMoAds[1] for Ad requests, AntShield[2] for PII exposures and In-house datasets collected from real users.

Feature Extraction: In order to train the model, features are extracted while ensuring the removal of any sensitive data. This is accomplished through the implementation of the ”bag-of-words” method, as described in the research conducted by ReCon ([7]). This technique serves to identify and exclude both infrequent and common features across all packets, such as dst-ip, dst-port, and user-agent-contents.

Other than that the values after the delimiters like ”=” is also removed because it contains sensitive data like Zipcode,AndoidID,etc. however, certain values that do not follow this syntax are not removed from the feature space and those might contain sensitive information. These features are called Recon Words. Because of this both features and updates can contain sensitive data that is sent to server in order to keep FL Devices and servers on same page.To avoid that privacy risk, it purposely limits feature space to use only non-sensitive keys from HTTP packets.

In particular, it considers the structure of HTTP packets and extracts features from the URI query keys, the Cookie keys, custom HTTP headers and whether or not there is a file request in the packet.

Testing: This work Demonstrate the effectiveness of a model trained using such method in terms of Classification performance and Communication and computa-

tion cost Using three real-world datasets against two extreme ends and FL

- local training: none of the training data or model is shared but limited by the local data only.
- Centralised training: All training data shared with the server, but privacy is lost.
- Federated Learning (FL): Only model parameters are shared. Train a global model that includes data from all clients, while also preserving privacy. '

Threat Modeling and analysis for the model:

- Inference attack by "an honest-but-curious server" (receives the data and uses it for the intended purpose but also saves it for its use) and outcome of such attack are Specific Feature recovery and Predicting visited domain.
- Analysis of the effects of changes in algorithm parameter for model training on this threat.
- Discussion of solution for such an attack using secure aggregation.

FedPacket leverages federated learning to train classifier.But even after removing sensitive information before training, some features can be infered using Inference attack using model and linkage attack with other publicaly available data.To mitigate this we have implemented differenctial privacy.

**Federated Learning With Differential Privacy[9]:**

In this work, the authors introduce the "noising before model aggregation FL" (NbAFL) framework, which leverages differential privacy (DP) to enhance privacy-preserving federated learning.

The NbAFL framework prevents information leakage by injecting artificial noise into the model parameters on the client side prior to aggregation. It's primary objective is to achieve differential privacy while maintaining convergence performance.

Key Contributions and Findings:

- DP Satisfaction: The authors demonstrate that NbAFL can satisfy DP under distinct protection levels by adjusting the variances of the added noise appropriately.

- Theoretical Convergence Bound: A theoretical convergence bound is developed for the loss function of the FL model trained within NbAFL. Three essential properties emerge:

  1. Tradeoff between convergence performance and privacy protection levels (better performance implies lower protection).
  2. Improved convergence with an increased number of participating clients.
  3. Optimal aggregation rounds for a given privacy level. Calculation of values of $\sigma$ and $c$ needed for optimal noise to be added using the data size and number of clients to have ($\varepsilon$, $\delta$)-deferrentially privacy is derived in this work. which is,

$$\Delta s = \frac{2c}{|D_i|}$$
$$\sigma = \frac{c\dot{\Delta} s}{\varepsilon}$$

  where,
  $c = \sqrt{2 \cdot \log\left(\frac{1.25}{\delta}\right)}$

- K-Client Random Scheduling: The authors propose a K-client random scheduling strategy, randomly selecting $K$ clients ($1 < K < N$) for each aggregation step.

- Practical Implications: NbAFL provides a principled approach to balancing privacy and performance in federated learning.

**Comment on "Federated Learning With Differential Privacy: Algorithms and Performance Analysis"[10]:** This works discusses mistake in Previous research. Previous research (2.2) derives an equation of $\Delta s = \frac{2C}{|D_i|}$.But this research shows that it should be $\Delta s = 2C$.

# Dataset Description

Both of the data is in slight variations of the JSON format used by ReCon

### 3.0.1 NoMoAds [1]

This data was collected by [11] by manually interacting and capturing packet traces of 50 popular Android apps. They extracted and saved fields from each packet primarily from HTTP/S and IP headers.

Other than HTTP/S and IP headers the JSON for each packet includes[1]: **"ad":** Label indicating if the packet contains an ad request. **"pii_types":** Types of personally identifiable information (PII) found. Redacted for privacy. **"package_responsible":** App package fetching the ad. **"package_name":** App package responsible for the HTTP connection fetching the ad, which may differ from "package_responsible". **"package_version":** Version number of the app provided by "package_name".

### 3.0.2 AntShield [2]

The data was collected by manually interaction with 100 popular Android Apps.But unlike NoMoAds they also collected data for 400 popular apps automatically using Monkey . Each packet contains HTTP/S and IP header.Additionally,The JSON for each packet includes[2]: **"label":** Indicates if the packet contains personally identifiable information (PII). Details on label acquisition are in the paper. **"pii_types":** Shows types of PII found. We redacted actual data, retaining only the PII type. **"package_name":** App's name responsible for the TCP/UDP connection. **"package_version":** App version from "package_name". **"post_body":** For HTTP POST messages, it's the body; otherwise, it holds packet data, usually from segmented HTTP(S) connections.

# METHODOLOGY

## 4.1 Problem Setup

Our goal is to train classifier to predict PII and ad-request using features extracted from HTTP packets.To achieve this we need training data which contains packets labelled for PII and ad-request.

We assume that training data are collected and labeled on client's side to a server that aggregates them and trains classifiers and Client do not trust the server or other devices but do want to contribute to the training and use the resulting global classifier. So we aim to provide such clients with a methodology that allows them to contribute to training of global classifier with minimum exposure of their data.

We have employed federated learning to this packet classification task so that client can participate without sending any data to server. When we are working with packet extracting key posses some challenges.First of all is identifing and seprating keys is difficult because HTTP packets containes non-dictionary words (urls and api request) madeof random letters and numbers.This raises a problem because we can not use pre-trained embeddings on NLP corpus.Another problem is state-of-the-art on PII/Ad Request classification [7][11]trained DTs in fully centralized way, and features e.g., words extracted from packets were fixed a priori. We are using Fedrated learning, so alll clients must share their feature space before starting FL, Which raises privacy concers and scalability challenges. As more clients participates union of feature explodes.To solve privacy issue only HTTP keys are shared as features not values. To solve the second issue we are using bag-of-words used in [7] to filter out the keys that may not be related to PII/Ad requests.

## 4.2 Data Collection

We collected in-house data using Wireshark.Collecting http packets is easy but to collect https packet we needed to store session keys then using them we were able decrypt them.Then we need to extract HTTP headers from them and format them in Json format used by NoMoAds [1], AntShield [2] and ReCon [7].This then can be used by Feature Extracter.We were facing challanges with collecting our own data for mutiple apps automatically So DataSets from NoMoAds [1] and AntShield [2] is used in this research.

## 4.3 Feature Extraction

Feature extraction is done in the following steps:

**Iteration:** Open the JSON file in read mode and iterate through the packets in the JSON file. For each packet, iterate through its fields. **Tokenization:** For each field, list all the words separated by the delimiters (/, =,:,...) in the packet and save them in the list_of_keys packet-wise. **Counting:** Maintain a global_list/master_key_list for all packets containing all the keys and their occurrences in packets (total_count), along with occurrences in packets with PII and Ad-Request (target_count). **Filtering:** Filter the keys from the master_key_list based on the ratio of target_count/total_count and total_count.We used this heuristic to filter-out keys that aren't necessarily related to PII[7]. **Format and save:** Generate a CSV file based on the existence of keys from master_key_list in list_of_keys packet-wise, with the label pii_type in the packet, and train our model based on that.

We have shown this for a single json file containing single packet in Figure 4.1 and output csv for the same in Figure 4.2.

## 4.4 Model Training

Using this csv file every client trains their model and submits to server for aggragation.Server aggregates the model and replies back with the global model.In this menor the global model is trained. In the Beginning of the research SVM was used but because of limited support of libaries for model aggragation in later part Neural Network is used.

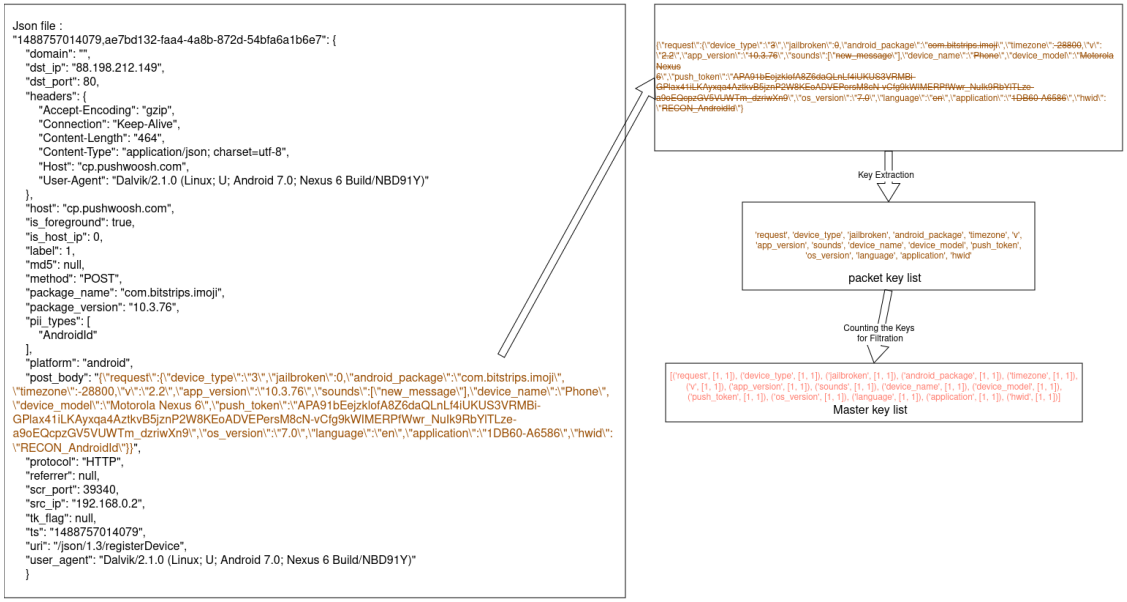
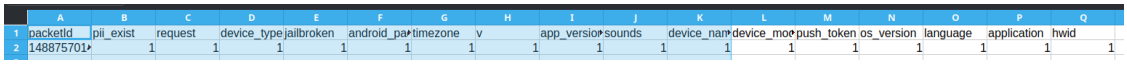



Figure 4.1: Process of extracting keys.



| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | packetId | pii_exist | request | device_type | jailbroken | android_pa | timezone | v | app_versio | sounds | device_nam | device_mo | push_token | os_version | language | application | hwid |
| 2 | 148875701 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Figure 4.2: CSV file of extracting keys.

in Our model architecture consists of a neural network, specifically a Multi-Layer Perceptron (MLP), tailored for packet classification tasks.

### 4.4.1 Neural Network Architecture

The MLP comprises an input layer, a single hidden layer, and an output layer.

- The input layer receives features extracted from HTTP packets.
- The hidden layer performs computations on the input features, introducing nonlinearity through activation functions like ReLU.
- The output layer utilizes softmax activation, producing a probability distribution over classes (PII or ad-request).
- The Methematical equation of the neural network is:

$$\mathbf{y} = \sigma(\mathbf{W}_2 \cdot \sigma(\mathbf{W}_1 \cdot \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2) \tag{4.1}$$

where:

- $\mathbf{x}$ is the input vector,

- $\mathbf{W}_1$ and $\mathbf{W}_2$ are the weight matrices of the first and second layers, respectively,
- $\mathbf{b}_1$ and $\mathbf{b}_2$ are the bias vectors of the first and second layers, respectively,
- $\sigma(\cdot)$ represents the activation function (e.g., ReLU or sigmoid), and
- $\mathbf{y}$ is the output vector.

### 4.4.2 Training Procedure

The model is trained using the Adam optimizer and binary cross-entropy loss function.

- During training, the model iteratively adjusts its parameters to minimize the loss function, optimizing its ability to classify packets accurately.
- Backpropagation is employed to compute gradients and update the model's parameters accordingly.

## 4.5 Global Model Aggregation

- In federated learning, the global model is updated by aggregating the model parameters from all participating clients.
- This process ensures that the global model reflects the collective knowledge of all clients while preserving privacy.

## 4.6 Federated Learning

- Federated learning enables multiple clients to collaboratively train a shared model while keeping their data decentralized.
- Each client trains a local model on its respective data without sharing it centrally, ensuring data privacy and security.
- After local training, the model parameters are sent to a central server, which aggregates them to update the global model.
- This collaborative learning approach allows for scalable model training while preserving privacy and data locality.

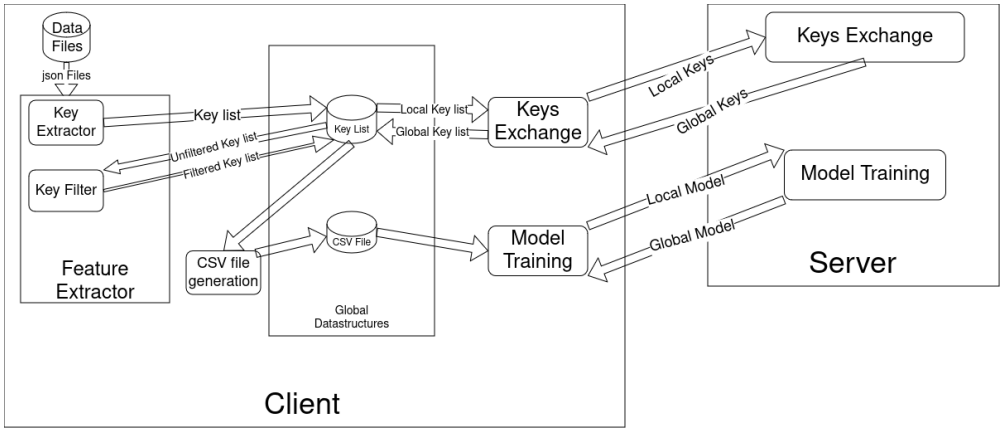



Figure 4.3: Architecture of our Program

## 4.7 Client-Sever Architecture

We have impllemented Client and server model to implement the FL training algorithm.

**Algorithm 1:** Federated Learning Server Pseudocode

**Input:** client_num, clients_list, client_models, global_key_list, k
**Output:** Model aggregation and communication functions
Wait for connections from clients;
Accept client connections and add them to clients_list;
Send key list request to each client;
**while** *not all clients have sent their keys* **do**
  receive keys from the client
**end**
Broadcast global key list to all clients;
**while** *Global model's accuracy* $<$ *trashhold* **do**
  Receive models from all clients and store them;
  Perform model aggregation;
  Broadcast global model to clients;
**end**

**Algorithm 2:** Federated Learning Client Pseudocode

**Input:** client_socket, host, port, client_data_path, outFile
**Output:** Functions for communication, key extraction, and model training
extract keys from files and prepare a key list;
Connect to the server;
send the key list to server;
**while** *server has not sent back global key list* **do**
  Wait
**end**
Receive and store the global key list received from the server;
**while** *Global model's accuracy* $<$ *trashhold* **do**
  Perform model training(with dp) using the global key list and Global model;
  **while** *Server requests for model* **do**
    wait
  **end**
  Send model to server;
  receive global model;
**end**

## 4.8 Differential Privacy:

### 4.8.1 Noise Addition Mechanism

The noise addition mechanism is designed to perturb the model's weights, thereby obfuscating the contributions of individual data points. This is crucial to preserve the privacy of the data set.

In our work we are adding noise on client side before sending model to server like discussed in [9].The values of $\varepsilon$ and $\delta$ are taken from client based on the level of privacy they prefer.Using them other parameters are calculted that generates optimal noise for ($\varepsilon$, $\delta$)-differential privacy.

$$c = \sqrt{2 \cdot \log\left(\frac{1.25}{\delta}\right)}, \Delta s = \frac{2c}{|D_i|}, \sigma = \frac{c\dot{\Delta}s}{\varepsilon}$$

# Result

When preprocessing data for ML.We used a heuristic based filter to reduce the number of columns.This are the results of with and without the filters.

Results:

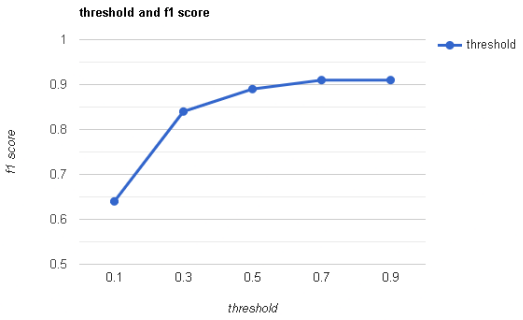
| | Non-filtered | Filtered |
|---|---|---|
| Keys/Features: | <class 'pandas.core.frame.DataFrame'><br>RangeIndex: 1807 entries, 0 to 1806<br>Columns: 10742 entries, packetId to sdk-core-v40-loader.appcache.1<br>dtypes: int64(10741), object(1)<br>memory usage: 148.1+ MB | <class 'pandas.core.frame.DataFrame'><br>RangeIndex: 1807 entries, 0 to 1806<br>Columns: 6306 entries, packetId to 52.22.199.27.1<br>dtypes: int64(6305), object(1)<br>memory usage: 86.9+ MB |
| Training Results: | precision recall f1-score support<br>0 0.99 0.99 0.99 130<br>1 1.00 1.00 1.00 23<br>2 0.93 0.96 0.95 27<br>3 0.00 0.00 0.00 1<br>accuracy 0.98 181<br>macro avg 0.73 0.74 0.73 181<br>weighted avg 0.98 0.98 0.98 181 | precision recall f1-score support<br>0 0.99 0.99 0.99 130<br>1 1.00 1.00 1.00 23<br>2 0.93 0.96 0.95 27<br>3 0.00 0.00 0.00 1<br>accuracy 0.98 181<br>macro avg 0.73 0.74 0.73 181<br>weighted avg 0.98 0.98 0.98 181 |

Table 5.1: Comparison of Keys/Features in first row and Training Results in second between Non-filtered and Filtered Data with 0.65 treshhold for heuristic for filtering key. Left column shows results for Non-filtered data, right column for Filtered data.

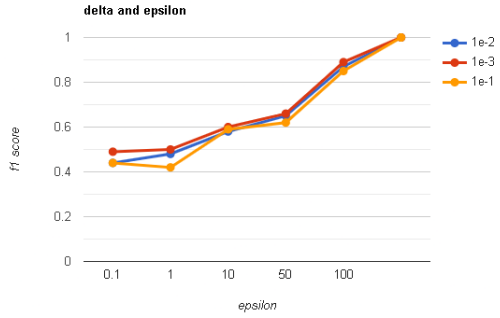
Table 5.2 contains insites about how fitering keys impact resourses utilization and model's accuracy.

left top figure in Table 5.2 shows that there are 10742 keys and 148 MB memory is used before filtering but after filtering only 6306 keys and 87 MB is used. But as we observed, even when we reduce the number of keys to half, that doesn't change the model's accuracy. However, the Time taken to train a model and the resources required is significantly reduced after filtering.

Also Figure 5.1a Shows us that as we increase the threshold F1-score also




(a) Key Filtration Threshold vs F1-score




(b) Epsilon vs F1-score for deltas = $1E^{-1}$, $1E^{-2}$, & $1E^{-3}$

increases.The reason for that is key that are not related to PII/Ad request being filter out.

Results for Diffrencial Privacy with $\varepsilon = 1000$ and $\delta = 10^{-2}$.This shows how much the noise introduced affects the accuracy of the model.We have take very large delta here to show the disparity.

Without DP F1 score of all the local model in higher that with DP resulting in global model with lower F1 score if we use DP.

| Without-DP | With-DP |
|---|---|
| 3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>Client 1 F1 Score: 0.8138845281702423 F1 Global 0.86507713884993<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>Client 2 F1 Score: 0.7783582537680899 F1 Global 0.886247352514219<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step<br>Client 3 F1 Score: 0.8254330415347364 F1 Global 0.9558876811594202<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step<br>Client 4 F1 Score: 0.79036016144809 F1 Global 0.9100514258999531<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>Client 5 F1 Score: 0.8803606649760497 F1 Global 0.9781157187199928<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step<br>Client 6 F1 Score: 0.8803606649760497 F1 Global 0.9781157187199928<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>Client 7 F1 Score: 0.8915452363728226 F1 Global 0.9558876811594202<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step<br>2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>Client 8 F1 Score: 0.9021132713440406 F1 Global 0.9558876811594202<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>...<br>Confusion Matrix:<br>[[54 6]<br>[ 3 28]]<br>F1 Score: 0.9021132713440406 | 3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step<br>Client 1 F1 Score: 0.6445251554909115 F1 Global 0.5810760667903524<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step<br>Client 2 F1 Score: 0.6567876567876568 F1 Global 0.6878068519390625<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step<br>Client 3 F1 Score: 0.6678175412623069 F1 Global 0.666811817466023<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step<br>Client 4 F1 Score: 0.6114380975062089 F1 Global 0.7060692714164485<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step<br>Client 5 F1 Score: 0.6628346012961397 F1 Global 0.6724826897240691<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 37ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step<br>Client 6 F1 Score: 0.6483516483516484 F1 Global 0.6724826897240691<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step<br>Client 7 F1 Score: 0.5781032724575877 F1 Global 0.6288137150206116<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step<br>Client 8 F1 Score: 0.5758241758241758 F1 Global 0.6605053066591529<br>3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step<br>...<br>Confusion Matrix:<br>[[43 17]<br>[18 13]]<br>F1 Score: 0.6138213500359551 |

Table 5.2: Results with $\varepsilon = 1000$ and $\delta = 10^{-2}$

In the graph in Figure 5.1b we have ploted epsilon vs f1-score for different values of deltas.We observed that as epsilon increases the f1-score increases and as delta increases f1-score decreases.

# Bibliography


[1] Athina Group, University of California, Irvine, "Nomoads data." https://athinagroup.eng.uci.edu/projects/nomoads/data/. [Accessed: 2024-02-15].

[2] Athina Group, University of California, Irvine, "Antshield dataset." https://athinagroup.eng.uci.edu/projects/antmonitor/antshield-dataset/. [Accessed: 2024-02-15].

[3] E. Bakopoulou, B. Tillman, and A. Markopoulou, "Fedpacket: A federated learning approach to mobile packet classification," *IEEE Transactions on Mobile Computing*, vol. 21, no. 10, pp. 3609–3628, 2022.

[4] Google Ads Help, "Google ads help: How google uses information from sites or apps that use our services," Accessed 2024. Paragraph: How it works.

[5] J. G. Cabanas, A. Cuevas, A. Arrate, and R. Cuevas, "Does facebook use sensitive data for advertising purposes?," *Communications of the ACM*, vol. 64, pp. 62–69, December 2020.

[6] *MobiCom '20: Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, (New York, NY, USA), Association for Computing Machinery, 2020.

[7] J. Ren, A. Rao, M. Lindorfer, A. Legout, and D. Choffnes, "Recon: Revealing and controlling pii leaks in mobile network traffic," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '16, ACM, June 2016.

[8] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, (New York, NY, USA), pp. 1322–1333, 2015.

[9] K. Wei, J. Li, M. Ding, C. Ma, H. Yang, F. Farokhi, S. Jin, T. Quek, and H. Vincent Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020. Publisher Copyright: © 2005-2012 IEEE.

[10] K. Rajkumar, A. Goswami, K. Lakshmanan, and R. Gupta, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 17, 2022.

[11] A. Shuba, A. Markopoulou, and Z. Shafiq, "NoMoAds: Effective and Efficient Cross-App Mobile Ad-Blocking," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 4, 2018.