Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

Anna Maria Dall Grimsmo Haug

A Survey of the Federated Learning and Differential Privacy Techniques

Master's thesis in Electronics Systems Design and Innovation
Supervisor: Tor Andre Myrvoll
June 2022

NTNU
Kunnskap for en bedre verden

Anna Maria Dall Grimsmo Haug

A Survey of the Federated Learning and Differential Privacy Techniques

Master's thesis in Electronics Systems Design and Innovation
Supervisor: Tor Andre Myrvoll
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

NTNU
Norwegian University of
Science and Technology

Abstract

To be able to utilize data sets containing sensitive information, they must be secure and protected. Federated learning is a technique in machine learning which aims to solve this problem. By bringing a global model to a local device, users can utilize and learn from a shared model while at the same time keeping their data local. However, due to being a relatively new concept, there are still some challenges with the approach. Hence, federated learning alone is not always strong enough to protect the data set. Therefore, it is often used in combination with differential privacy (DP). This is a definition that provides privacy by injecting noise.

The original definition of differential privacy, (ϵ)-DP, is often too strict to use in practice. Thus, many variations of the definition have been developed, and the most common is the (ϵ, δ)-DP. This is a relaxation of the definition allowing some leakage of information. Further, there are several mechanisms that provide differential privacy. The Laplace mechanism and the exponential mechanism are some of the most common approaches. Many machine learning classifiers, such as linear and logistic regression, deep neural networks, and support vector machines, can be adopted to provide differential privacy.

The privacy budget, ϵ, gives a trade-off between the performance of the model and the privacy of the data set. Experiments reveal that the amount of trade-off depends on the choice of e.g. data set, classifier, model, and approach. Some experiments indicate no significant trade-off, while others indicate a severe trade-off. In addition, there still exists some open questions about both federated learning and differential privacy requiring further research.

Sammendrag

For å kunne bruke et datasett som inneholder sensitiv informasjon, må det være sikret og beskyttet. *Federated learning* er en metode innen maskinlæring som prøver å løse dette problemet. Ved å bringe en global modell til en lokal enhet kan brukere dra nytte og lære av en felles modell, samtidig som de beholder sin data lokalt. Siden dette er et relativt nytt konsept derimot, er det fremdeles noen utfordringer med metoden. Derfor er ikke *federated learning* alene alltid sterkt nok til å beskytte datasettet. Av den grunn brukes det ofte i kombinasjon med differensielt personvern (DP). Dette er en definisjon som sikrer personvern ved bruk av støy.

Den originale definisjonen av differensielt personvern, (ϵ)-DP, er ofte for streng til at det kan brukes i praksis. Derfor har det blitt utviklet mange variasjoner av definisjonen, og den mest vanlige er (ϵ, δ)-DP. Dette er en avslappet definisjon som godtar noe lekkasje av informasjon. Videre er det flere mekanismer som sikrer differensielt personvern. Laplace mekanismen og den eksponentielle mekanismen er noen av de vanligste metodene. Mange maskinlæringsklassifiserere, slik som lineær og logistikk regresjon, dype nevrale nettverk og support vektor maskiner, kan tilpasses til å sikre differensielt personvern.

Personvernbudsjettet, ϵ, gir en avveining mellom ytelsen til modellen og personvernet til datasettet. Eksperimenter avslører at graden av avveining er påvirket av valget av for eksempel datasett, klassifiserer, modell og fremgangsmåte. Noen av eksperimentene viser ingen betydelig avveining, mens andre viser alvorlige avveininger. I tillegg eksisterer det fremdeles noen åpne spørsmål rundt både *federated learning* og differensielt personvern som krever videre undersøkelser.

Contents

Figures

Tables

Acronyms

CDP concentrated differential privacy. 24, 25

CNN convolutional neural network. 35

DL deep learning. 8–10

DP differential privacy. 1, 2, 18–27, 29, 31–33, 35–40, 44, 45, 47

DP-SGD differential private stochastic gradient descent. 35, 36

DPRF differentially private random forest. 33

FedAvg federated averaging. 7, 8, 10, 11

FedSGD federated stochastic gradient descent. 8, 10

FL federated learning. xi, 1, 3, 6–8, 10–15, 43, 47

GD gradient descent. 10

GDPR general data protection regulation. 12, 13, 47

MCC Matthews correlation coefficient. 36

ML machine learning. 1, 13

PCA principal component analysis. 39, 40

PriMIA privacy-preserving medical image analysis. 35, 36

RDP Rényi differential privacy. 25, 26

SGD stochastic gradient descent. 5, 10, 31, 39

SMO sequential minimal optimization. 32

SMPC secure multi-party computation. 36

SuLQ sub-linear queries. 18

SVM support vector machine. 32, 44

zCDP zero-concentrated differential privacy. 25

Chapter 1

Introduction

The purpose of machine learning (ML) is to gain knowledge from a data set and predict the best output [1]. However, data may be revealed to the public when it is collected and processed [2]. Today's improvements in regulations have resulted in it being more demanding to obtain data. Hence, central training approaches may not fulfill requirements regarding privacy. Individuals might seek to train ML algorithms only on their own data, however, this is often not an option since the amount of data might not be sufficient. Ideally, solely removing the names of the individuals in a study would be enough to anonymize the data [1]. Although, if some other information is already held by an attacker, individuals can be identified.

To solve these problems, the concept of federated learning (FL) was proposed [3]. With this approach, individuals can train their own data locally on e.g. their smart phones enabling users to learn predictions from a shared model. This shared model is controlled by a global server who obtains updates from all the users. Since this is a fairly new concept and thus have some unsolved challenges, it is often used in combination with the definition differential privacy (DP). This is a definition where random noise is utilized to prevent leakage of information [1]. On that account, no individual will affect the data set if it is removed, hence the attacker cannot with certainty gain knowledge about an individual.

One challenge with DP is that it might affect the performance of the training [1]. Although by using large data sets, researchers have developed models that can capture the distributions while differential privacy is guaranteed. Here, individuals or single samples cannot be heavily relied upon by the model to be able to generalize the method. This will provide both usefulness and privacy without the effects of individual samples.

By definition of differential privacy, there follows a privacy budget [4]. Every time a network is trained, a part of this budget will be used. This budget is defined by the parameter ϵ from the definition (ϵ)-differential privacy, which is the original and strictest definition of DP. Another way of improving the utility while still perceiving a small privacy budget, is to slacken the interpretation of differential privacy. Thus, a variety of definitions of DP has been proposed. The most utilized

definition in practice is (ϵ, δ)-differential privacy where a pre-defined amount of accidentally leaked information, δ, is allowed.

The purpose of this thesis is to give an overview of the federated learning and differential privacy field, as well as investigate how the privacy budget affects the performance of the training by reviewing previous research. This thesis is structured as follows. Chapter two presents the federated learning concept, promises and technical limitations, as well as different variations of the approach. Additionally, a brief overview of several types of attacks are presented. Chapter three presents the differential privacy definition as well as the most common types of mechanisms utilized to obtain DP. Furthermore, some variations of the definition are introduced as well as the promises of DP. In chapter four some of the most common machine learning classifiers are presented along with how they can adapt to provide differential privacy. Three experiments from related work are presented in chapter five. Here, their methods and choices are stated together with the results of their study. These experiments are among others used to discuss the issue of this thesis in chapter six. Lastly, a conclusion and further work are presented in chapter seven.

Chapter 2

Federated Learning

Google researchers McMahan and Ramage [3] introduced in 2017 the concept of federated learning. By using data sets who are distributed over several devices, the machine learning models can prevent data leakage. Hence, training data is kept locally on the device while multiple users can learn collaboratively from a shared global model. This is the main idea behind the concept. Instead of bringing the data to a global model, the model is brought to the data.

With FL, the data does not have to be trained on a centralized machine, as in the standard approaches in machine learning [3]. Several users can then share a predicted model without sharing their own data. A server or cloud holds the current model, which is downloaded by the users to their device. Data stored on the device is used for local training to improve the model. All the changes are summarized in an encrypted update sent back to the server. Updates from all the users are averaged and used for improvement of the shared model. This ensures that the data is kept local, and the server deletes all the updates from the devices after the shared model is improved. An illustration of this is presented in figure 2.1 [3].

2.1 Definition

Federated learning can be defined as having N data owners $\mathcal{F}_1, ..., \mathcal{F}_N$ with data $\mathcal{D}_1, ..., \mathcal{D}_N$ aiming to train a model $\mathcal{M}$ [5]. In traditional machine learning, $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_N$ when all the data is used in model training. Instead, FL exploits a shared model $\mathcal{M}_{FED}$ when an owner $\mathcal{F}_i$ seek to train their own data $\mathcal{D}_i$. The concept of FL promises that the owner's data is not exposed to any of the other participants. Additionally, $\mathcal{V}_{FED}$ and $\mathcal{V}_{SUM}$ symbolizes the accuracy of respectively the federated model $\mathcal{M}_{FED}$ and the traditional model $\mathcal{M}_{SUM}$. Ideally, these two accuracies should be equal. According to Yang *et al.* [5], the federated learning algorithm obtains a δ-accuracy loss if

$$|\mathcal{V}_{FED} - \mathcal{V}_{SUM}| < \delta \tag{2.1}$$

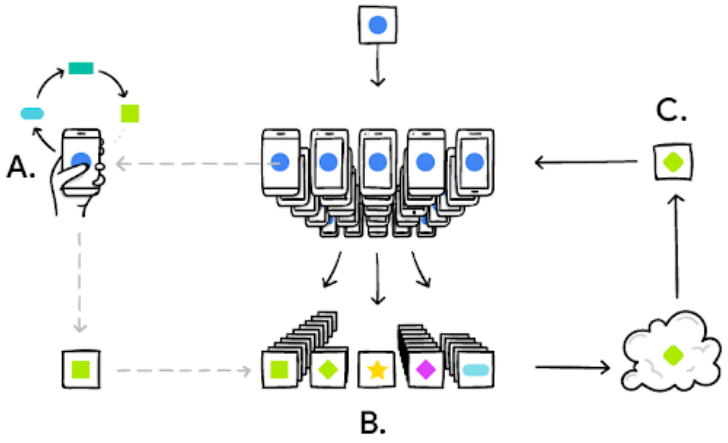


Figure 2.1: Illustration of the federated learning process retrieved from McMahan and Ramage [3].

holds. Here, δ is a real, positive number.

In a federated learning setting, the server collects and stores updates from several users to improve the current model [6]. Once these updates have been used to obtain an improvement of the model, they are no longer stored on the server. Hence, the server will only hold information from the current training. Additionally, there is no need for the server to have access to all the training data from the users. There is still a possibility that the updates contain sensitive information, which require trusting the server. Furthermore, the server's job is to coordinate the training.

2.2 Main Features

2.2.1 Iterative Learning

An engineer is typically the one who is operating the federated learning process to obtain a model from a specific task [7]. The life cycle of such a process is depicted in figure 2.2.

The typical workflow of the life cycle of a FL model can be divided into six steps [7]. The first step is for the engineer to identify the problem. Next, several clients with local training data are instrumented for the task. Optionally, a proxy data set can be used for simulation prototyping. Step four contains the federated model training, where alterations of the model are being trained by several federated tasks. After sufficient training, the model evaluation is conducted by engineers and analysts who selects good candidates. This analysis can be a federated evaluation, or it can be evaluated on standard data sets. The final step is to launch the selected

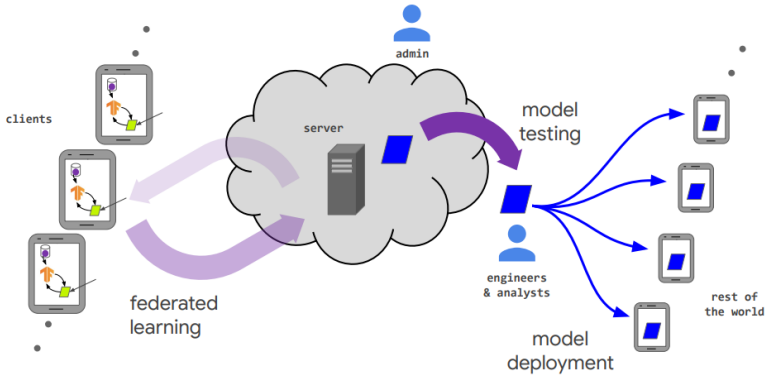


Figure 2.2: A sketch of the federated learning life cycle retrieved from Kairouz *et al.* [7].

Table 2.1: Table of the order-of-magnitude of a typical FL application retrieved from Kairouz *et al.* [7].

Total population size	$10^6 - 10^{10}$ devices
Devices selected for one round of training	50-5000
Total devices that participate in training one model	$10^5 - 10^7$
Number of rounds for model convergence	500-10000
Wall-clock training time	1-10 days

model.

As mentioned, step four of a typical federated learning workflow is to train a model. Assuming the Federated Averaging algorithm (which will be defined in section 2.4), the process of federated training can be summarized in five steps [7]. These steps are managed by the server until the training is completed. First step is to select the clients from a set. In practice, this could be e.g. different phones where only the connected ones are being picked. Second step is to send the training program and the current model weights to the clients participating in the process. Furthermore, every selected client perform training on their own data locally and sends an update back to the server. In Federated Averaging, this is done by running stochastic gradient descent (SGD) (which will be defined in section 2.4) locally. The fourth step contains aggregation. Here, all the updates from the clients are aggregated and collected by the server. At this point, if there is an acceptable number of results from clients, the server may improve efficiency by dropping clients who are struggling. This step can be done in different ways e.g. lossy compression, noise adding and update clipping, or secure aggregation. Last step is for the server to use the aggregated update from the current round to update the shared model. The order-of-magnitude in a typical application of federated learning for e.g., mobile devices, is given in table 2.1.

2.2.2 Non-IID data

In a federated learning task, sampling has to be done twice [7]. First, the model samples a client $i \sim Q$, where Q is the distribution of all the clients available. Then, from the local data distribution of that client, a sample $(x, y) \sim P_i(x, y)$ is drawn. Here, x and y corresponds to respectively the features and labels in the task.

Generally in federated learning, the data is not independent and identically distributed (non-IID) [7]. For two clients i and j, this is defined as $P_i \neq P_j$. By using the definition of joint probability, $P_i(x, y)$ can be written as $P_i(x|y)P_i(y)$ and $P_i(y|x)P_i(x)$. Some of the most common ways where the data is non-IID are given in the list below.

- **Covariate shift**: Even though $P_i(y|x)P_j(y|x)$, the marginal distributions $P_i(x)$ might differ throughout the clients. E.g. when recognizing handwriting, there will still be small variations within one single persons writing.
- **Prior probability shift**: Even though $P_i(x|y)P_j(x|y)$, the marginal distributions $P_i(y)$ might differ throughout the clients. E.g. the distribution of labels may have a different range corresponding to the geographical location of the client.
- **Concept drift**: Even though $P(y)$ is the same for all clients, the conditional distributions $P_i(x|y)$ might differ throughout the clients. A variety of features x can correspond to the same label y.
- **Concept shift**: Even though $P(x)$ is the same for all clients, the conditional distributions $P_i(y|x)$ might differ throughout the clients. Equivalent to the previous item in this list, a variaty of labels y can correspond to the same feature x, due to personal preferences.
- **Unbalancedness**: The amounts of data held by various clients may be significantly different.

2.2.3 Cross-Device vs. Cross-Silo

Federated learning can be divided into two different schemes: cross-device and cross-silo, illustrated in figure 2.3 [8]. In cross-device FL the data is distributed across mobile phones or IoT devices locally, while in cross-silo FL the data is distributed across e.g. organizations or data centers [7]. Hence, cross-device has a large number of clients (up till 10^{10}) with a smaller amount of data, while cross-silo have fewer clients (usually 2-100) with a larger amount of data.

Another difference between the two types is that in cross-device FL solely a few clients are accessible at once, while in cross-silo FL almost all clients can be accessed at the same time [7]. In cross-device FL a major bottleneck is communication. The reason for this is because the devices, or clients, often use wifi-connections when performing computations. Sometimes even slower connections are used. This might also cause a bottleneck in cross-silo FL, but not to the same extent. Additionally, in cross-device the clients cannot be identified, whereas in

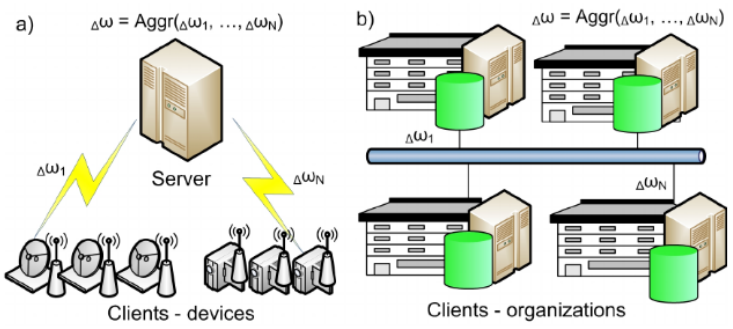


Figure 2.3: Illustration of the two federated learning types: cross-device and cross-silo, retrieved from Kholod *et al.* [8].

cross-silo the system can identify a client to specifically access it. Furthermore, in cross-silo all the clients may engage in every round of training. In cross-device, this is not the case. Here, it is more likely that a client only engages in a task once. Then, in each round of training it can be assumed that only unseen clients engage. However, the clients are highly unreliable because it is expected that at least 5% of them will drop out. This happens when a device is disqualified due to e.g. violation of network, battery, or idleness requirements.

Federated learning was originally presented as the cross-device type by Google, and is the most commonly used [7]. Therefore, most emphasis is directed towards this scheme throughout this thesis.

2.3 A Typical Federated Learning Algorithm

A typical federated learning algorithm is presented in table 2.2 given by McMahan *et al.* [9]. This is an example of the federated averaging (FedAvg) variation. The server starts by initializing a model w_0 with predefined parameters. These parameters are the number of clients K, the size of the local mini-batch B, the number of local epochs E, and the learning rate η. Every round, given by t, the server decides the amount of clients who are going to participate. Here, C defines a fraction multiplied with the total number of clients K, resulting in m amount of clients. Additionally, the server randomly collects these clients, given by S_t. For each client, the model is updated given the clients training data locally, providing w_{t+1}^k. When all the clients have updated their current model locally, the shared model w_{t+1} is updated by taking the weighted average of all the local updates. Here, n_k represents the amount of data samples that client k has available for training.

The client update is performed by taking one step of gradient descent [9]. This will be described later in section 2.4. On each client's device, their training examples, denoted by P_k, are divided into batches β of size B. A total number of local epochs, E, runs, where for each epoch and each batch, the model is trained.

Here, $l(w;b)$ corresponds to a loss function. This results in an updated model w which is returned to the server.

Table 2.2: A typical federated learning algorithm retrieved from McMahan *et al.* [9].

Algorithm Federated Averaging. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.
Server executes:
initialize w_0
for each round $t = 1, 2, ...$ **do**
$m \leftarrow max(C \cdot K, 1)$
$S_t \leftarrow$ (random set of m clients)
for each client $k \in S_t$ **in parallel do**
$w_{t+1}^k \leftarrow ClientUpdate(k, w_t)$
$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$
ClientUpdate(k, w)**:** *//Run on client k*
$\beta \leftarrow$ (split P_k into batches of B)
for each local epoch i from 1 to E **do**
for batch $b \in \beta$ **do**
$w \leftarrow w - \eta \nabla l(w; b)$
return w to server

2.4 Federated Learning Variations

Federated learning can be performed in different matters, and a selection of the variations is presented below. The federated averaging (FedAvg) and the federated stochastic gradient descent (FedSGD) are the most common ones in practice. Before defining some different federated learning variations, some background on deep learning (DL) is provided.

2.4.1 Deep Learning

In deep learning, complex features are extracted from high-dimensional data for the purpose of constructing a model with an input-output relationship [10]. This relationship can be defined by for example classes. Usually, multi-layer networks are used as architecture, and a classic neural network consisting of two hidden layers is illustrated in figure 2.4. Here, the circles are called nodes, and each of them models a neuron.

The first neurons, which are the black circles in figure 2.4, are bias nodes who emits 1 to the neurons in the next layer [10]. Further, the neurons send their output to the neurons in the next layer. When a neuron receives this as inputs, a

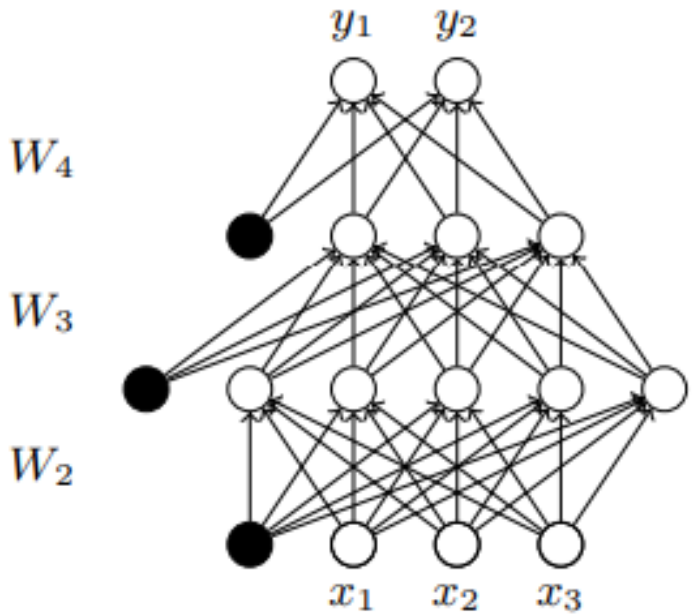


Figure 2.4: Demonstration of a typical neural network retrieved from Shokri and Shmatikov [10].

weighted average is computed and is called the total input. Using this total input, a non-linear activation function is applied to compute an output. Given a layer k with neurons, the output vector is given by

$$\mathbf{a}_k = f(W_k \mathbf{a}_{k-1}), \tag{2.2}$$

where the input signals contribution is regulated by a weighted matrix given by W_k, and the activation function is given by f. Some commonly used activation functions are listed as follows [10]:

- **Hyperbolic tangent:** $f(z) = (e^{2z} - 1)(e^{2z} + 1)^{-1}$
- **Sigmoid:** $f(z) = (1 + e^{-z})^{-1}$
- **Rectifier:** $f(z) = max(0, z)$
- **Softplus:** $f(z) = log(1 + e^z)$
- **Softmax:** $f(z_j) = e^{z_j} \cdot (\sum_k e^{z_k})^{-1}, \forall j$

The softmax function is often used in the last layer when a finite number of classes is used [10]. If so, neuron j's output correlates to the probability that its input corresponds to class j.

At each layer, features are extracted according to f and W_k [10]. From the training data, the network can learn which parameter values, also called the weighted matrices, that maximizes the purpose of the network. A major challenge is to automatically engineer this in DL. Learning parameters can be achieved by

supervised, semi-supervised or unsupervised learning.

Gradient descent, or a variation of it, is typically used to update the parameters in supervised learning [10]. First, a random set of parameters are initialized for the neural network. This is the starting point of the algorithm. The non-linear functions gradient is computed in each step and is optimized. Further, to decrease the gradient, the parameters are updated. When the algorithm converges to a local optimum, this operation is stopped.

Computation of the gradient of the parameters is done through procedures such as feed-forward and back-propagation [10]. Given the input data, the output of the network is computed by feed-forward. Here, the error is calculated, and represents the change from the true value of the function and the achieved output. This error is used by back-propagation which propagates it backwards through the network to compute the total error by inspecting every neuron's contribution. The neuron's contribution to the error and its activation values are exploited to compute the individual parameters gradients.

A simplification of gradient descent is stochastic gradient descent (SGD) [10]. This algorithm uses only a really small subset of the data set called mini-batches. Some advantages by using SGD rather than GD are that it is less computationally expensive and faster, which makes it able to performe training on larger samples [11]. Additionally, SGD is stochastic in nature, this is not the case for GD which is deterministic in nature.

2.4.2 Federated Stochastic Gradient Descent (FedSGD)

Optimization in deep learning relies almost entirely on different variations of stochastic gradient descent [9]. In federated learning, calculation of a single batch gradient can be performed in every communication round. To obtain a good model, this approach requires a high amount of training rounds. In the setting of FL, having a large batch is equivalent to involving more clients, which is not a huge cost in time. A fraction of clients, C, is selected every round, and the gradient of the loss is computed on the client's data. C also controls the global batch size, where a full-batch is used when $C = 1$. This is the federated stochastic gradient descent algorithm.

2.4.3 Federated Averaging (FedAvg)

A generalization of the federated stochastic gradient descent is the federated averaging [12]. In this variation, instead of updating the gradients, the weights are updated. This is possible because the local nodes are allowed to use the local data to utilize several batch updates, and not only one. Averaging the gradients or the weights is analogous if the initialization is the same for all the local nodes. Hence, a generalization of the FedSGD can be made by utilizing the weights instead of the gradients. The algorithm for FedAvg is presented in table 2.2 in section 2.3.

FedAvg was introduced to solve the problem of limited bandwidth and latency [3]. Deep networks can be trained with $10 - 100$ times less communication. In-

stead of using simple gradient steps to compute updates, FedAvg utilizes the mobile devices' powerful processors to obtain updates with higher quality. To produce a good model, these high-quality updates require fewer iterations. Hence, the training communication is much less. Random rotations and quantization are utilized to compress the updates, reducing the cost of upload communication.

2.4.4 Federated Stochastic Variance Reduced Gradient (FSVRG)

In federated stochastic variance reduced gradient (FSVRG), each client distributes stochastic updates, and only one computation of the expensive gradient is performed centrally [12]. This stochastic update is executed by selecting a random collection of the local data and iterating through it, resulting in each data point performing one update. This variation of FL solely relies on one hyper parameter called the step size h. The local step size h_k for client k is given by $h_k = h/n_k$. Even if n_k differ a lot between clients, h_k ensures that the amount of progress made for each client should approximately be the same.

As in most variations of federated learning, the current model w_t is sent to the clients where they use their local data to compute the loss gradients [12]. The server obtains a gradient by aggregating all the uploaded client gradients. When the clients receive this gradient from the server, their step size h_k and model w_t^k can locally be initialized. The gradient from the server, the local gradient, and h_k are all used to perform n_k SVRG updates from a random permutation of the client's data. At last, the server collects all w_{t+1}^k from all the clients and creates a new current model w_{t+1} based on them.

2.4.5 Federated Learning with Dynamic Regularization (FedDyn)

Acar *et al.* [13] introduced another variation of federated learning called federated learning with dynamic regularization (FedDyn). This method aims to solve the issue that arises when heterogeneously distributed data sets are used. Consequently, the global loss objective cannot in this case be minimized by minimizing the loss function locally by the clients. In FedDyn the total loss from all the clients converges to the global loss by dynamically regularizing the loss function at each client. This method aligns the local losses, which in turn ensures performing minimization on the client's devices properly, as well as ensuring robustness at different levels of heterogeneity.

2.4.6 Personalized Federated Learning by Pruning (Sub-FedAvg)

With the non-IID data that is often utilized in FL, good global performance can be extremely difficult to achieve [14]. Therefore, Vahidian *et al.* [14] introduced Sub-FedAvg which utilizes personalized models. This algorithm exploits both structured and unstructured pruning where the sub-networks of the clients are averaged on the intersection. This means that the weight connections are removed in the network for the purpose of decreasing the size of the storage in the model as

well as increasing the speed [15]. Simultaneously, communication efficiency and the accuracies of the personalized models are controlled [14].

2.5 Promises

Privacy of the data is the primary benefit of federated learning [3]. This approach of machine learning keeps the data local and only exchanges encrypted parameters to a central server. Hacking the information in the data set is therefore in theory more difficult. Additionally, to ensure privacy, FL also assures models that are smarter, have lower latency, and have less consumption of power.

Unlike the traditional machine learning approaches, federated learning aims to guarantee privacy of the data which fulfills the general data protection regulation (GDPR) conditions [16]. The main aspects of GDPR is illustrated in figure 2.5 [17]. Protection of personal data is the aim of this regularization, and it is fully described in 99 articles. Organizations in EU/UK then have detailed requirements for how the personal data is supposed to be handled.

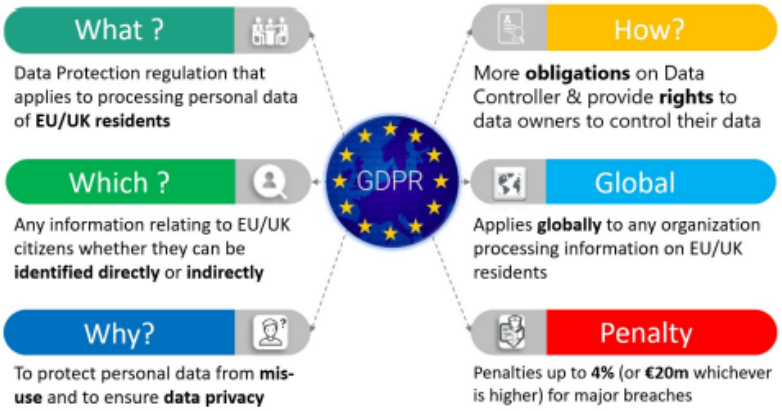


Figure 2.5: The essence of GDPR retrieved from Truong *et al.* [17].

The data protection framework can be divided into: Data Subject, Data Controller, and Data Processor [17]. To process the data, the Data Subject must give consent to the Data Controller. Full responsibility of the processing of personal data is given to the Data Controller.

Storing and processing the personal data locally, as well as only exchanging the parameters to the model, are some of the reasons why the GDPR holds in federated learning [17]. Data privacy and security are also enhanced by applying cryptography techniques to the aggregation and updating of the parameters. Clients' personal data does not have to be shared with service providers to be able to use the algorithm to implement applications or services. Therefore, the regulations of data protection are followed. With FL, models can be trained on data from different countries because there is no need to transfer the data to a central

server. Additionally, the server does not have direct access to the local training data or the local models in a FL system. Model parameters are aggregated and updates the global model in such way that the privacy of the clients is minimally impacted.

According to the GDPR, a Data Controller can only utilize data from clients that is related to the claimed purposes [17]. This is a challenge in centralized machine learning because it is often difficult to predict which data is essential for the model training. However, in FL several local models, and not the original data itself, are provided from the clients to obtain a global model. Therefore, FL can ensure that the parameters from the local models are used for the only purpose of updating the global model. In addition, since the updates from the clients are aggregated, there are no sensitive information from any individual in the server. Since the server does not hold any private information, there is nothing on the server for any adversary to exploit.

2.6 Technical Limitations

Heterogeneous and enormous networks are often utilized when training in a federated learning setting [18]. This can cause some severe challenges, and four main problems are presented below.

In federated networks a crucial bottleneck is communication [18]. Specifically, a great amount of devices may participate in federated training, causing slower network communication. Instead of waiting until the training process is finished before the entire data set is sent, it would be more efficient to implement small updates from the model during the training. By developing such communication-efficient methods, it would be more straightforward to fit a model. Further, the size of each message per round should be reduced as well as communication rounds in total should be fewer.

Each device may provide different hardware, network connection, or battery power, which may lead to providing different capacity within communication, storage, or computation [18]. Further, the amount of devices that are active at the same time are equal to only a tiny fraction due to limitations in systems or the size of the network. Additionally, in any iteration of the training, a device that is active may drop out because of connection or battery, as well as the reliability of the devices may be low. Therefore, all federated learning methods should aim to allow hardware that are heterogeneous, expect the number of participants to be low, as well as being sturdy to devices that are dropped during the training process.

Consider a task of next word prediction. Several users would write in their own fashion providing variations in the language. Therefore, the collected and generated data would be distributed non-identically over the network [18]. Additionally, there may be a significant variation within the devices amount of data points. Thus, IID data cannot be assumed in the distribution, which in turn could increase the probability of devices struggling, as well as result in modeling, ana-

lysis, and evaluation complexity. Such statistical heterogeneity can often be operated by utilizing device-specific or personalized learning.

One of the substantial concerns within federated learning is privacy [18]. By sharing information from the gradients in updates rather than the raw data, FL aims to protect the user's private information. Although, experiments reveal that the server or a third-party may pick up some sensitive information when communicating updates from the local models. An alternative to strengthen the privacy is to combine federated learning with differential privacy. However, a theory is that the performance of the model is reduced in this setting. Thus, a significant challenge is to balance and understand the trade-off between privacy and utility.

2.7 Examples of Federated Learning Applications

Smart phones can typically be utilized as devices to perform FL applications [18]. For example, multiple mobile phones can jointly learn the behavior of the users, or face and voice recognition can be applicated from statistical models. Nevertheless, most users would not allow their private information to be shared with others. Federated learning can possibly prevent private information to leak as well as provide good user experience. An application like this is depicted in figure 2.6 [18].

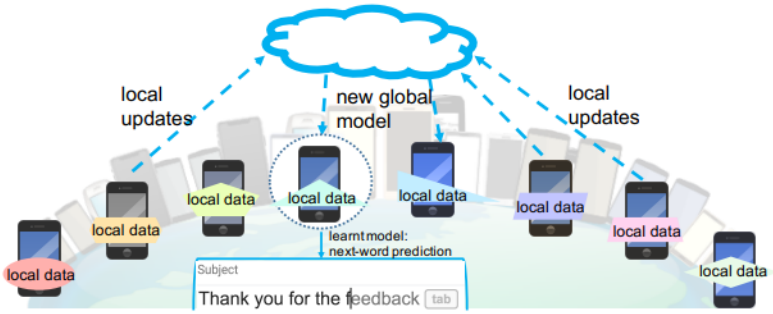


Figure 2.6: A next-word prediction example of a federated learning application on smart phones retrieved from Li *et al.* [18].

Other types of devices in FL can be institutions or organizations [18]. Predictive healthcare can for example be performed on several patients by hospitals. Since hospitals require strict privacy preservation, it is important that the private data is kept local. FL enables learning in a private manner among numerous organizations.

Nowadays, the internet of things (IoT) is all around us in smart homes, wearable devices, and self-driving cars [18]. These devices often consist of several sensors which utilizes real-time data. In self-driving cars for example, it would be essential to have up-to-date information about the surroundings to operate safely. Federated learning can provide privacy for the users while at the same time

provide systems with training models that adapt to the environment efficiently.

2.8 Attacks

As stated earlier, sensitive information could be revealed when updates are communicated to the server [19]. Information may be vulnerable to an adversary server or third-party, illustrated in figure 2.7. Such an evil server could make observations over time of the updates provided from the different individuals, or even interfere in the process of training. An adversary third-party could be an observer who surveys the parameters held by the server and could further exploit this information to change its own parameters before uploading them. Additionally, such observers could create a backdoor to the server that is hidden.

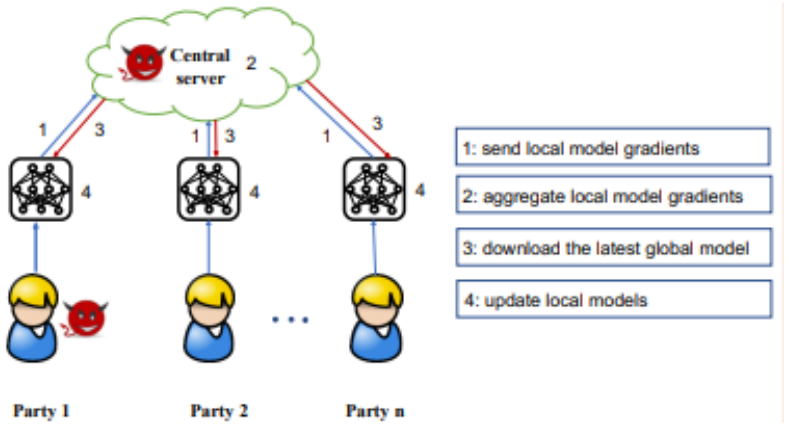


Figure 2.7: Illustration of the federated learning procedure with an adversary server and an adversary third-party retrieved from Lyu *et al.* [19].

One form of attack against the federated learning models are single attacks [19]. A single individual may choose some inputs where the goal is to make the model wrongly classify them confidently. Another form of attack is byzantine attacks. Such individuals are hard to detect because they often have a random behavior and adapts their outputs in a fashion that provides the same distribution as the non-adversary updates. Further, sybil attacks are also a form of attack against the FL models. These attackers exploit fake individuals, or even take advantage of individuals from the study with revealed identity, to attack the model.

Adversaries can perform attacks in two different settings: honest-but-curious and malicious [19]. When attackers are honest-but-curious they aim to gain knowledge from other parties' private states. Additionally, they only observe the gradient at the server who have already been aggregated or averaged, not other individuals' data or gradients. In contrast, the malicious attackers aim to gain knowledge from other parties' private states while diverging from the FL protocol. They can alter or remove messages from other individuals to obtain destructive attacks.

Depending on if the attacker aims to attack in the training phase or in the

inference phase, they are called respectively poisoning attacks and evasion attacks [19]. With poisoning attacks, the adversary can inject fake data to the training pool to influence the outcome of the training. This data can either be injected in the individual's collection of local data, also called data poisoning, or into the training process of the model, also called model poisoning. Figure 2.8 illustrates these two types of poisoning attacks. With model poisoning, backdoors to the global model can be inserted.

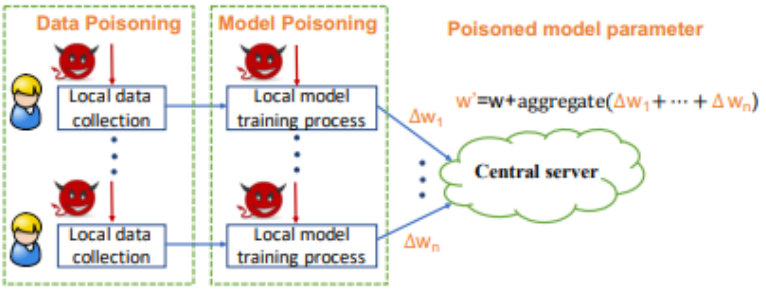


Figure 2.8: retrieved from Lyu *et al.* [19].

Evasion attacks aims to force the model to construct wrong outputs [19]. Another goal is to gather information about the characteristics of the model. Therefore, the information available concerning the model controls the attacks effectiveness.

Chapter 3

Differential Privacy

The promise of differential privacy aims to gain knowledge about a population without revealing any information about an individual [20]. If the conclusion of a study remains the same, whether or not a specific individual is contained in the data set, the promise of differential privacy holds. Regardless of an individual's contribution to the data set, any sequence of responses to queries should be obtained with the same probability.

The main idea behind this definition is giving people opportunity to be a part of a study and share their data without reviling any sensitive information. An individual should not be affected or harmed by participating in an analysis. By this definition, sensitive data such as in healthcare, can be utilized to obtain greater knowledge within an important field.

In machine learning it is interesting to use data that contains a lot of information to obtain good results [20]. For this reason, data needs to be anonymized as well as remain useful for analysis. By using differential privacy, re-identification of the data is neutralized because being differentially private is not related to the information contained in the data set. In addition, the amount and type of queries are also factors in the concern of safety. Neither answering questions about specific individuals in the data set nor rejecting them can be done without compromising the privacy.

3.1 Definition

A database D containing data from individuals, is held by a trusted curator [20]. Assuming that each row corresponds to one single individual's data, the aim is to protect each individual while at the same time be able to perform statistical analysis from the database. To gain knowledge about the database D, queries can be applied. These are questions asked to D by the data analyst. Previous responses to queries are used by the model to decide which is the next query to pose. Further, the best accuracy is obtained when all queries are given preliminary. Thus, when the queries' structure is known, noise can be correlated by the model. Therefore, if the number of queries increases, the accuracy will decrease to ensure privacy.

Dwork and Roth [20] defines differential privacy as in definition 1.

Definition 1 *"A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ϵ, δ)-differentially private if for all $\mathcal{S} \subseteq Range(\mathcal{M})$ and for all $x, y \in \mathbb{N}^{|\mathcal{X}|}$ such that $\|x - y\|_1 \leq 1$:*

$$Pr[\mathcal{M}(x) \in \mathcal{S}] \leq exp(\epsilon) Pr[\mathcal{M}(y) \in \mathcal{S}] + \delta \tag{3.1}$$

" [20].

In definition 1, $\mathcal{M}$ is a random algorithm, $\mathcal{X}$ is the set of all database rows, $\mathcal{S}$ is all possible outputs of $\mathcal{M}$, x and y are respectively the entries in the database and the parallel database, and δ is the probability of leakage of information [20]. As mentioned earlier, ϵ defines the privacy loss or privacy budget. Hence, given a query on database (x) and (y), the maximum distance between these two are defined as ϵ.

Given the definition of differential privacy, different algorithms will achieve ϵ-differential privacy for a computational task [20]. The parameter ϵ defines the amount of privacy and how much interaction with the database is allowed. A smaller ϵ corresponds to a small amount of interaction with the database, hence more privacy is obtained. Although, it can be difficult to find an algorithm for the task with high accuracy when ϵ is small. Therefore, it is important to find a balance between the amount of privacy and the accuracy of the task.

3.1.1 The SuLQ Framework

Some of the earliest applications of differential privacy was introduced by Blum *et al.* [21] and is called the SuLQ framework. Consider a situation where the aim is to gain useful knowledge by applying algorithms to a data set containing individual private information, as well as preserving privacy. Furthermore, consider the sub-linear queries (SuLQ) output-perturbation framework where the pair (S, f) defines a query [21]. Given a database D, the function f maps S into $\{0, 1\}$. This results in a response containing noise defined as $\sum_{r \in S} f(DB_R)$. Assuming that the number of database rows is sub-linear to the total number of queries, only a small portion of noise is required to gain a solid type of privacy. This reflects the framework term sub-linear queries.

As stated, a database can be defined as a domain D with elements $(d_1, d_2, ..., d_n)$ [21]. Given T queries with answers and a predicate $f : D \rightarrow \{0, 1\}$ (while assuming independence between the elements d_i), $p_0^{i,f}$ and $p_T^{i,f}$ defines respectively the *a priori* and the *a posteriori* belief that $f(d_i) = 1$. For all $i' \neq i$, all the rows will have the value $d_{i'}$. In addition,

$$conf(p) = log\left(\frac{p}{1-p}\right) \tag{3.2}$$

defines the 1-1 mapping $conf : (0, 1) \rightarrow \mathbb{R}$, which is monotonically-increasing [21]. This means ideally that the probability should not increase from the *a priori*

belief to the *a posteriori* belief. Hence, the user does not know any more information about the data set after the analysis than before the query was applied. Therefore, the individual in the database is not harmed by the analysis if nothing is learned. The keynote of defining this privacy is to prevent the user from gaining knowledge about the data set, but still be able to use the data for training. Furthermore, by using $conf$ to smudge the probabilities between 0 and infinity, the system will not make any hard decisions.

Given a small ϵ and similar query inputs, results will give a similar output as well [22]. That being the case, an adversary attacker will struggle to analyze the results and therefore not be able to gain information. For that reason, the smaller the ϵ, the higher level of privacy is obtained. ϵ may therefore be determined by the user or analyst to condition the amount of privacy needed for the task.

3.1.2 (ϵ)-Differential Privacy and (ϵ, δ)-Differential Privacy

By definition 1, two different definitions can be formed: (ϵ)-DP and (ϵ, δ)-DP [20]. (ϵ)-DP was defined first, but since $\delta = 0$, no leakage is allowed. Therefore, this definition is not always useful in practice because of its strictness. As a consequence, the (ϵ, δ)-DP was defined. Here, a small amount of accidental leakage is allowed with the intention of making the definition useful in practice.

With (ϵ, δ)-DP there is allowed some leakage, i.e. this is a relaxation of the (ϵ)-DP definition [22]. This relaxation may be used due to larger database sizes or higher sensitivity values. Such definition provides more flexibility and is therefore used more often. The analyst has more freedom when designing, and the utility trade-off is theoretically lower.

δ should be chosen such that $\delta < \frac{1}{\|x\|_1}$ [20]. This means that the inverse of the size of the database is bigger than the amount of leakage allowed. These values of δ prevent outcomes where the privacy is violated, but does not prevent leakage in other forms [23].

In view of the fact that (ϵ)-DP cannot guarantee privacy when the Gaussian mechanism is utilized, (ϵ, δ)-DP was introduced [23]. The definition of the Gaussian mechanism is provided in section 3.2.

3.1.3 Global vs. Local Differential Privacy

Differential privacy can be classified as either local DP or global DP depending on where the noise is inserted [24]. Figure 3.1 retrieved from Fathima [24] illustrates these two situations.

In local DP the noise is added to the input of the database, either directly or to the data set of the individuals prior to the database [24]. Although, accuracy will be lost by using averaged values. Then, the accuracy of the results will be less if the privacy protection is high. An advantage of using local DP is that the data curator does not have to be trusted by the client, i.e. the clients data is not exposed and privacy is ensured. However, this will result in a larger amount of noise in total because each client must add noise. As a consequence, to obtain

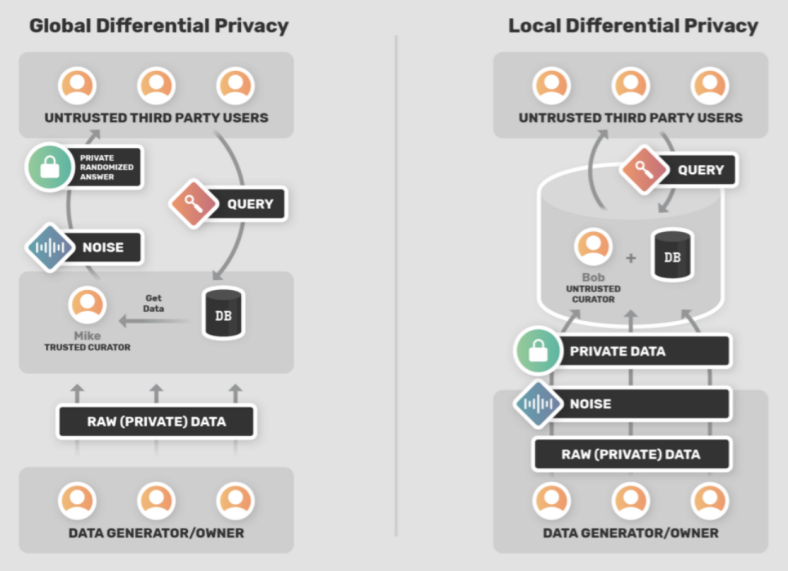


Figure 3.1: Illustration of global and local differential privacy retrieved from Fathima [24].

useful results, a great amount of clients would need to participate. Therefore, ϵ tends to be high in practice to reduce this effect.

Local DP can be useful in practice and is used by different companies today [24]. For example in *RAPPOR* which is Googles way of gathering information from users. Another example is *private count mean sketch* which is Apples way of gathering information from clients use of e.g. emojis and words.

When the output of the database is injected with noise, global DP is performed [24]. Here, the data curator adds noise only once after the process. Moreover, the curator has access to all the private information, and it has to be trusted by the clients. This DP provides more accurate results, and only a small amount of noise needs to be injected with a low ϵ to obtain reasonable results. However, when collecting all the data in the same database, the consequences of an attack will be significantly increased.

When the owner of the database is trustworthy, there is only one difference between local and global DP, which is that the global DP obtains more accuracy in the results [24]. Therefore, global DP should be utilized when the curator is trustworthy. On the other hand, local DP should be used when the user cannot share its private information with the curator.

3.1.4 Queries

Queries, or questions, are functions that can be applied by a data analyst to the database in order to interact with the model [20]. Previous query responses are observed to decide the order of the proceeding queries. The best accuracy is obtained when a non-interactive model knows every query ahead of time. If not,

every potential query must be answered by the model, which in turn causes challenges. In fact, when the amount of queries increase, the accuracy of the responses will decrease in order to provide privacy.

3.1.5 Privacy Budget

As stated earlier, ϵ is also called the privacy budget [22]. All individuals in the study have their own privacy budget. When a calculation of DP is performed on an individual's data, their privacy budget is decreased. This means that the privacy budget defines how much interaction with the data is allowed. Therefore, a low ϵ means less interaction with the data, and a high ϵ means a lot of interaction. However, a high ϵ may provide lower privacy depending on which queries are asked. Additionally, a malicious server or a third-party may gain too much information when ϵ is high.

3.2 Differentially Private Mechanisms

When a database, a universe $\mathcal{X}$ containing types of data, arbitrary bits, and a group of queries, performs as inputs to an algorithm, then this algorithm can be called a mechanism [20]. Given these queries, the goal is to decode the output to construct considerably accurate answers.

Several mechanisms are developed to obtain differential privacy. Some of the most utilized methods today are presented below.

3.2.1 The Laplace Mechanism

A fundamental database query is the function $f : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathbb{R}^k$, also called numeric queries [20]. Here, the database is mapped by the query to k real numbers. The accuracy of the answer to these queries are given by the sensitivity parameter l_1. This sensitivity is defined by Dwork and Roth [20], given in definition 2.

Definition 2 *"The l_1-sensitivity of a function $f : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathbb{R}^k$ is:*

$$\Delta f = \max_{x,y \in \mathbb{N}^{|\mathcal{X}|}, \|x-y\|_1 = 1} \|f(x) - f(y)\|_1 \tag{3.3}$$

" [20].

This definition describes the maximal impact or change one single individual's data can have on the function f [20]. As a consequence, this also represents the amount of uncertainty that has to be added in the response to not reveal the individual's participation in the study. In order to obtain privacy, this definition provides an upper bound of how much the output needs to be perturbed.

When f has been computed by the Laplace mechanism, noise is added to each coordinate to perturb the output [20]. This noise is modeled by the Laplace distribution, defined by Dwork and Roth [20] in definition 3. The sensitivity of

f/ϵ decides the calibration of the noise. Further, Dwork and Roth [20] defines the Laplace mechanism as given in definition 4.

Definition 3 *"The Laplace Distribution (centered at 0) with scale b is the distribution with probability density function:*

$$Lap(x|b) = \frac{1}{2b} exp\left(-\frac{|x|}{b}\right) \tag{3.4}$$

" [20].

Definition 4 *"Given any function $f : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathbb{R}^k$, the Laplace mechanism is defined as:*

$$\mathcal{M}_L(x, f(\cdot), \epsilon) = f(x) + (Y_1, ..., Y_k) \tag{3.5}$$

where Y_i are i.i.d. random variables drawn from $Lap(\Delta f/\epsilon)$ " [20].

Definition 4 of the Laplace mechanism provides the strongest form of privacy, in other words, it ensures (ϵ)-DP [20]. This is one of the most common methods in practice to obtain differential privacy.

Some of the drawbacks with the Laplace mechanism are that it is not that good for queries with high sensitivity, and if there is need for a lot of queries, a large value of epsilon is required [25]. Additionally, this mechanism only operates on numeric queries.

3.2.2 The Exponential Mechanism

In some cases, adding noise to the output of the query can ruin the essence of the data [20]. Therefore, the exponential mechanism was developed. Given a utility function $u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} \rightarrow \mathbb{R}$, where $\mathcal{R}$ defines an arbitrary range and the database is mapped to utility scores by u, then the exponential mechanism can be determined in relation to this function u. It is preferred that the maximal utility score is the output of $\mathcal{R}$ given a fixed database x. Ideally, every possible $r \in \mathcal{R}$ that obtains a probability proportional to $exp(\epsilon u(x, r)/\Delta u)$ should be outputs of the mechanism. Additionally, the privacy loss can be defined as:

$$ln\left(\frac{exp(\epsilon u(x, r)/\Delta u)}{exp(\epsilon u(y, r)/\Delta u)}\right) = \epsilon\left[u(x, r) - u(y, r)\right]/\Delta u \leq \epsilon \tag{3.6}$$

If an individual is added in the database, some r would increase or decrease, which is not taken into account by this intuitive view [20]. Therefore, a normalization term is included in the definition of the exponential mechanism from Dwork and Roth [20], given in definition 5. Equivalent to the Laplace mechanism, this exponential mechanism also ensures (ϵ)-DP [20].

Definition 5 *"The exponential mechanism $\mathcal{M}_E(x, u, \mathcal{R})$ selects and outputs an element $r \in \mathcal{R}$ with probability proportional to $exp(\frac{\epsilon u(x,r)}{2\Delta u})$" [20].*

An advantage of this mechanism is that it can handle both numerical and categorical queries [25]. Additionally, this mechanism can provide precise answers while at the same time ensuring differential privacy. This is the main difference between the Laplace mechanism and the exponential mechanism. Here, a member of $\mathcal{R}$ is always the output of the mechanism, which is especially useful when a precise answer is necessary.

Despite all these advantages, there are some drawbacks of this mechanism [25]. This is a truly general mechanism, which in turn provides looser bounds and implementation can be complicated in practice.

3.2.3 The Gaussian Mechanism

Instead of adding Laplacian noise, it is possible to add Gaussian noise [20]. Here, the l_2 sensitivity is used to scale the noise. The Gaussian mechanism is defined by Dwork and Roth [20], given in definition 6. In this mechanism, zero-mean Gaussian noise is added in every k with a variance b.

Definition 6 *"The l_2-sensitivity of a function $f : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathbb{R}^k$ is:*

$$\Delta_2(f) = \max_{x,y \in \mathbb{N}^{|\mathcal{X}|}, \|x-y\|_1 = 1} \|f(x) - f(y)\|_2 \tag{3.7}$$

" [20].

Unlike the two other mechanism mentioned above, this mechanism is (ϵ, δ)-DP [20]. In practice, this guarantee will not be experienced as a weakness if there is a reasonably small δ. Indeed, the Gaussian mechanism can be utilized in every way as the Laplace mechanism for real-valued functions [25]. However, this mechanism is not as accurate as the Laplace mechanism.

One of reasons why the Gaussian mechanism sometimes is used instead of the Laplace mechanism is because the Gaussian mechanism can utilize both l_1 and l_2 sensitivity [25]. Sometimes the l_1 sensitivity can be significantly larger than the l_2 sensitivity. Then, by using the Gaussian mechanism with the l_2 sensitivity, considerably less noise can be added.

3.2.4 The Sparse Vector Technique

Occasionally, a great amount of queries have to be answered to obtain a good study [20]. To guarantee privacy however, this is not possible in the other mechanisms mentioned above. Additionally, when it is only necessary to identify queries greater than a given threshold, the sparse vector technique can be utilized. Here, the queries below the threshold will be ignored. Simply put, this technique will insert noise and only report the values that falls above the threshold. A huge difference from the other mechanisms is that instead of having the privacy degrade with the total amount of queries, here it degrades solely with the amount of queries that is greater than the threshold. As a consequence, if the total amount of queries

is a lot higher than the amount of queries above the threshold, then this technique can have enormously savings. In other words, this means that the answer vector is sparse.

Given a known threshold, the queries that are above this value will reveal a sequence of events [20]. Further, a vector will indicate which event that has occurred or not. Noise will be added to the response of each query and compared to the threshold. If the response has a value greater than the threshold, it will be revealed.

This is a brief description of the sparse vector technique. There are a lot of different variants of the sparse vector technique which will obtain various amounts of differential privacy. Some of these are presented by Lyu *et al.* [26].

3.3 Variants of Differential Privacy

A motivation for developing new variations of (ϵ)-differential privacy is that when multiple computations are utilized, it degrades smoothly and predictably [27]. Given k (ϵ)-differentially private computations, then the total computation will be $(k\epsilon)$-differentially private. The most known variation is the relaxation (ϵ, δ)-DP, which was described in section 3.1.

There exist several variations of differential privacy, and in this section three of the most commonly used variations are presented.

3.3.1 Concentrated Differential Privacy

Concentrated differential privacy (CDP) is a relaxation of the (ϵ)-DP [28]. It behaves in the same way as the (ϵ, δ)-DP, yet the two relaxations are still very different. If equation 3.8 holds, then it is said that an algorithm is (μ, τ)-CDP. Here, μ corresponds to the mean of the privacy loss random variable and ξ corresponds to the resulting random variable. Further, τ is the standard of the sub-Gaussian ξ [29].

$$Pr[\xi \geq x] \leq exp\left(-\frac{x^2}{2\tau^2}\right) \text{ and } Pr[\xi \leq -x] \leq exp\left(-\frac{x^2}{2\tau^2}\right) \tag{3.8}$$

From this equation, the privacy loss that can be expected with CDP is μ, and $e^{-\frac{t^2}{2}}$ is the bounded probability that the mean of the loss is exceeded with $x = t\tau$ [28]. There are two advantages of CDP over (ϵ, δ)-DP. First and foremost, the CDP provides improved accuracy. In fact, the privacy-utility-trade-off is improved by a factor $\sqrt{2}$. Additionally, given a mechanism that satisfies (ϵ)-DP and a group of size s, CDP satisfies $(s^2 \cdot \mu, s \cdot \tau)$-concentrated differential privacy.

Dwork and Rothblum [28] defines (μ, τ)-concentrated differential privacy as stated in definition 7. Here, $D_{subG}(\mathcal{M}(x)||\mathcal{M}(y))$ means that $\mathcal{M}(x)$ and $\mathcal{M}(y)$ are sub-Gaussian divergent. Given a mechanism with privacy loss that is sub-Gaussian and has a small mean, this definition states that the mechanism satisfies concentrated differential privacy.

Definition 7 *"A randomized algorithm $\mathcal{M}$ is (μ, τ)-concentrated differentially private if for all pairs of adjacent databases x, y, we have $D_{subG}(\mathcal{M}(x)||\mathcal{M}(y)) \preceq (\mu, \tau)$." [28].*

3.3.2 Zero-Concentrated Differential Privacy

Bun and Steinke [27] presented zero-concentrated differential privacy (zCDP) which is an alternative to concentrated differential privacy. They define it as stated in definition 8.

Definition 8 *"A randomized mechanism $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ is (ξ, ρ)-zero-concentrated differentially private (henceforth (ξ, ρ)-zCDP) if, for all $x, x' \in \mathcal{X}^n$ differing on a single entry and all $\alpha \in (1, \infty)$,*

$$D_\alpha(\mathcal{M}(x)||\mathcal{M}(x')) \leq \xi + \rho\alpha, \tag{3.9}$$

where $D_\alpha(\mathcal{M}(x)||\mathcal{M}(x'))$ is the α-Rényi divergence between the distribution of $\mathcal{M}(x)$ and the distribution of $\mathcal{M}(x')$.

We define ρ-zCDP to be $(0, \rho)$-zCDP." [27].

In this definition, $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ corresponds to a random algorithm [27]. Here, $\mathcal{X}^n$ is the data set and $\mathcal{Y}$ is the computational outcome. This definition is a relaxation of concentrated differential privacy defined in the previous section. Specifically, a mechanism that is (μ, τ)-CDP is the same as being a $(\mu - \tau^2/2, \tau^2/2)$-zCDP mechanism. Both CDP and zCDP typically utilizes the Gaussian mechanism.

A difference between the two definitions is that the sub-Gaussian that bounds the loss of privacy is centered around different values [27]. For zCDP it is centered around zero, while for CDP it is centered around the mean. In practice, zCDP often provides satisfying privacy guarantees, and the stricter definition, CDP, is therefore most times not necessary.

For every value of $\delta > 0$, zCDP ensures the same promises as (ϵ, δ)-DP [27]. Additionally, (ϵ)-DP provide $(\frac{1}{2}\epsilon(e^\epsilon - 1))$-zCDP. Actually, algorithms that satisfy (ϵ)-DP, often also satisfy zCDP.

3.3.3 Rényi Differential Privacy

Another relaxation of DP is Rényi differential privacy (RDP) [23]. It provides a definition which is stronger than (ϵ, δ)-DP, and is based on the Rényi divergence. The definition of Rényi divergence is stated by Mironov [23] in definition 9.

Definition 9 *"For two probability distributions P and Q defined over $\mathcal{R}$, the Rényi divergence of order $\alpha > 1$ is*

$$D_\alpha(P||Q) \triangleq \frac{1}{\alpha - 1} log E_{x\ Q}\left(\frac{P(x)}{Q(x)}\right)^\alpha \tag{3.10}$$

" [23].

When $\alpha = \infty$, there is an immediate relation between differential privacy and the Rényi divergence. Therefore, the relaxation Rényi differential privacy is defined by Mironov [23] as stated in definition 10.

Definition 10 *"A randomized mechanism $f : \mathcal{D} \to \mathcal{R}$ is said to have ϵ-Rényi differential privacy of order α, or (α, ϵ)-RDP for short, if for any adjacent $\mathcal{D}, \mathcal{D}' \in \mathcal{D}$ it holds that*

$$D_\alpha(f(D)||f(D')) \leq \epsilon \tag{3.11}$$

" [23].

When the outcomes are less likely, the privacy bound of the Rényi differential privacy is waeker [23]. In comparison with (ϵ)-DP, the RDP provides weaker guarantee and is more difficult to use in practice. In spite of that, the RDP can lead to bounds that are stronger for small values of δ as well as it is simpler in analysis when comparing with (ϵ, δ)-DP.

3.4 Promises

Differential privacy promises that an individual should be protected and not be harmed in any way by sharing their information with a private database x [20]. If an individual is harmed after the release of the results $\mathcal{M}(x)$, DP promises that their participation in the study has not increased the probability of being harmed. This promise of DP is practical because it is easier for an individual decide to share their data. Indeed, the individuals main concern would be the probability of their data being harmed by participating in the study. By this promise, an individual can be certain that whether or not they participate in the study, the probability that their data would be harmed would not change. Therefore, DP can be the convincing factor for an individual to participate in the study.

When (ϵ)-DP is guaranteed, it is promised that a factor $exp(\epsilon) \approx (1 + \epsilon)$ is the upper limit of harm which an individual can expect [20]. For numerous of individuals in the study, this promise holds simultaneously even though their utility functions are entirely dissimilar.

DP promises to not reveal any information from the individuals data or the identity of the individual itself [20]. However, it does not promise protection against attackers. Unfortunately, in some cases it may be possible to conclude from the results of the study some information from an individual. Observing that a specific private attribute from a study and a public attribute correlate, does not violate the differential privacy. That is, whether or not an individual participated in the study, the probability of observing this correlation would be approximately the same. DP only promises disclosure of an individual's participation in the study, as well as disclosure of the individuals data.

3.5 Example of Differential Privacy Application

Given a person who is reflecting on whether or not to join a study, the main concern of the person would often be the protection of their data and the consequences of sharing information. Consider this being a study in a hospital where they need several patients to join. By utilizing the promises of differential privacy, patients could be certain that their sensitive information would not be exposed or misused in any other way. Their data would be kept private, and therefore it cannot be harmed. Hence, DP can convince more patients to join the study which in turn would provide a more robust study. Without a definition like DP, such studies might not be possible to perform because patients would not want to risk their privacy.

3.6 Differential Privacy Compared with Other Privacy Preserving Methods

Several other methods have been proposed to preserve privacy, for example l-diversity [30], t-closeness [31], k-anonymity [32], and M-invariance [33]. In comparison with these methods, differential privacy guarantees strong privacy for the individuals, and thus is reviewed as a model which provides well founded privacy [34]. Additionally, the DP mechanism is truly robust against background attacks. Even for attackers who aims to gain knowledge about any individual in the study, the DP mechanism provides guarantees which are more efficient than any of the other methods.

Chapter 4

Differential Privacy in Machine Learning

Often in data analysis, machine learning is utilized [20]. Even though differential privacy is a limitation, it is possible to perform several tasks of machine learning with these conditions. Actually, both aims to learn from a data set. All in all, they both wish to learn from a given data set at such extent that no single data point is dependent on the learning. Therefore, the two of them are strongly related.

Some of the most used machine learning classifiers are linear regression, logistic regression, deep neural network, support vector machine (SVM), and random forest. These models are defined below, as well as their adaptations to differential privacy.

4.1 Linear Regression

Linear regression aims to learn from a combination of linear features, and then visualize the results on a straight line [35]. A loss function is minimized to obtain the regression results, e.g. mean square error. An illustration of the linear regression classifier is depicted in figure 4.1 [36]. Here, the best potential line of the prediction is represented by the red line, while the blue dots represent the data points.

Gong *et al.* [35] presents one way of performing linear regression with differential privacy. Given a database, the features at the input are split by their relevance into either strongly relevant or weakly relevant. Further, the calculations of the privacy budget of both the strong and weak features are based on the input-output relevance. To obtain differential privacy, Laplacian noise is added to the objective function's coefficients. At last, the DP objective function can be minimized to optimize the parameters. All in all, linear regression with differential privacy can be performed by first analyzing the relevance of the features, and then based on this, perturb the objective [35].

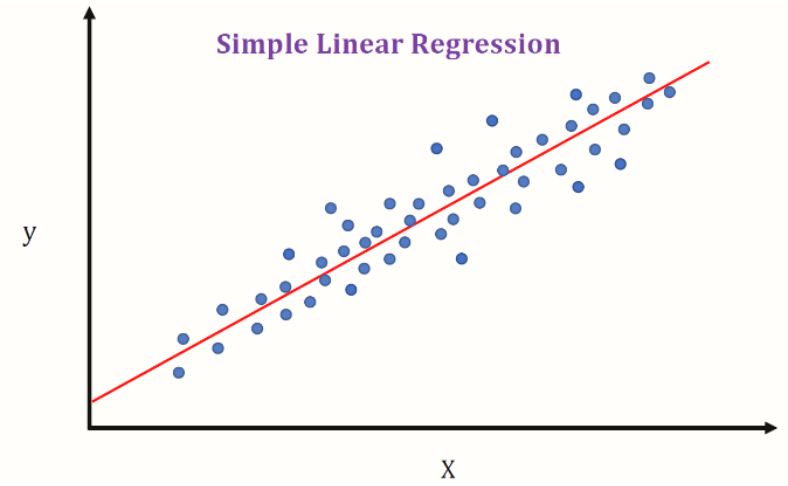


Figure 4.1: Illustration of the linear regression classifier retrieved from Nevrella [36].

4.2 Logistic Regression

Another type of regression analysis is the logistic regression, illustrated in figure 4.2 [37]. In this example, the data is divided into two classes: diabetes and non-diabetes corresponding to respectively the black circles and the white circles. Given a person's features, the analysis aims to predict if the person has diabetes or not. This analysis results in a straight line, here represented by the bold, black line. The points are placed based on their features, and points above the line corresponds to diabetes and points below the line corresponds to non-diabetes.

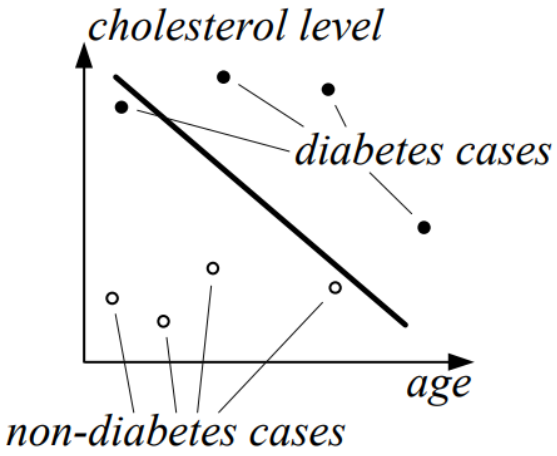


Figure 4.2: Illustration of the logistic regression classifier retrieved from Zhang *et al.* [37].

To obtain (ϵ)-differential privacy in logistic regression, the approach is the same as for linear regression [37]. Basically, instead of perturbing the results of the optimization problem, the objective function is perturbed. In this case, the results ensures (ϵ)-DP. Zhang *et al.* [37] utilizes the functional mechanism to obtain DP. This mechanism solves the problems of non-trivial noise injection and the fact that some objective functions are not valid when noise is injected.

4.3 Deep Neural Network

A deep neural network is defined in section 2.4.1 where a typical deep neural network is illustrated in figure 2.4. There are several methods to obtain differential privacy in deep neural networks, but only some of them provides outstanding methods.

Phan *et al.* [38] presented a model of a private convolutional deep belief network and Phan *et al.* [39] introduced an algorithm of a deep private auto-encoder, where both of them were constructed from the mechanism of perturbing the objective. To protect the training data, the functional mechanism can be utilized to inject noise to a polynomial's coefficients, which can be obtained by modifying the objective function.

A method to gain knowledge from deep learning models without having to reveal any information was introduced by Shokri and Shmatikov [10]. This method builds on perturbation of gradients and distributed selective SGD. Local models are trained on the training data held by the participants, and only perturbed gradients are uploaded. Additionally, shared parameters can be downloaded to their local models.

An algorithm with DP SGD was proposed by Abadi *et al.* [40]. Here, a random training subset is utilized to construct a gradient, which in turn is clipped according to the l_2 norm. Further, noise is injected to a group of multiple batches which holds the accumulated gradients. The cost of privacy is tracked by the moments accountant for the whole training procedure, thus the privacy loss bound is improved.

Xie *et al.* [41] proposed a model of generative adversarial networks that was differentially private. Throughout the process of training, the discriminator gradients can be injected with noise to preserve privacy.

Gong *et al.* [34] presented a deep neural network framework of adaptive differential privacy preserving learning which builds on analysis of relevance. Based on each neuron's contribution to the output of the model, gradients are injected with noise. Neurons who are less relevant in the output are injected with more noise in their gradients in the backward propagation process.

4.4 Support Vector Machine (SVM)

Given two classes, the support vector machine (SVM) aims to obtain a plane that maximizes the margin [42]. The separation of the classes could be different types of hyperplanes, depending on the number of features. Figure 4.3 illustrates the SVM classifier. The blue circles and the red squares represent two different classes, and the green lines represents different options for optimal hyperplanes. The dotted green line represents the maximum margin. Here, the blue circle and the two red squares that are filled represents the support vectors with most influence.

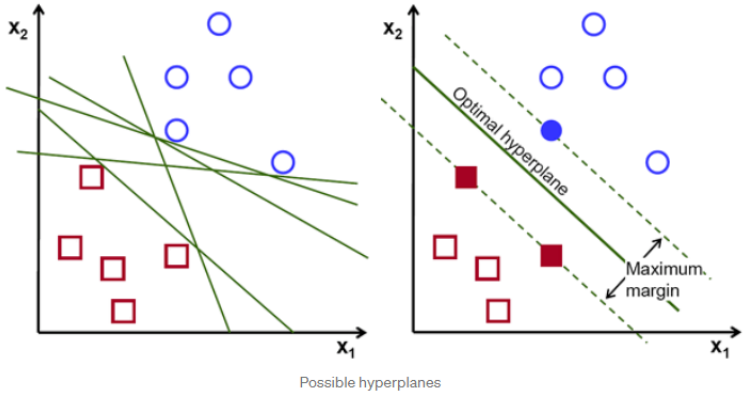


Figure 4.3: Illustration of the SVM classifier retrieved from Gandhi [42].

Over the years, many different approaches to obtain differential privacy for support vector machines have been proposed. However, these approaches have some problems [43]. Firstly, accuracy decreases when a large training set is utilized due to large time consumption and a large amount of noise is necessary. Additionally, the approaches lead to the objective function being too limited, as well as the solution being restricted to only particular types of the training set. Zhang *et al.* [43] introduced an approach of achieving DP with SVM established from the dual variable perturbation.

In the solution of solving the stated problems, Zhang *et al.* [43] utilizes the sequential minimal optimization (SMO) [44], which is an algorithm that trains support vector machines faster. First and foremost, the expectation of the support vector's true and estimated values is computed. Furthermore, all the support vector's expectations were summed, and the ratio of this and all the support vector's expectations are computed. This ratio defines the amount of Laplacian noise that is injected to the dual variables.

4.5 Random Forest

A classifier can be retrieved by sampling random vectors from an input vector [45]. Such a classifier can be called a tree classifier, and by combining several of

these classifiers, a random forest classifier can be obtained. Each tree will suggest a class that is the most popular for the given input vector.

Figure 4.4 illustrates a differentially private random forest (DPRF) algorithm [46]. To achieve differential privacy in the random forest classification, noise is injected in all the decision trees.

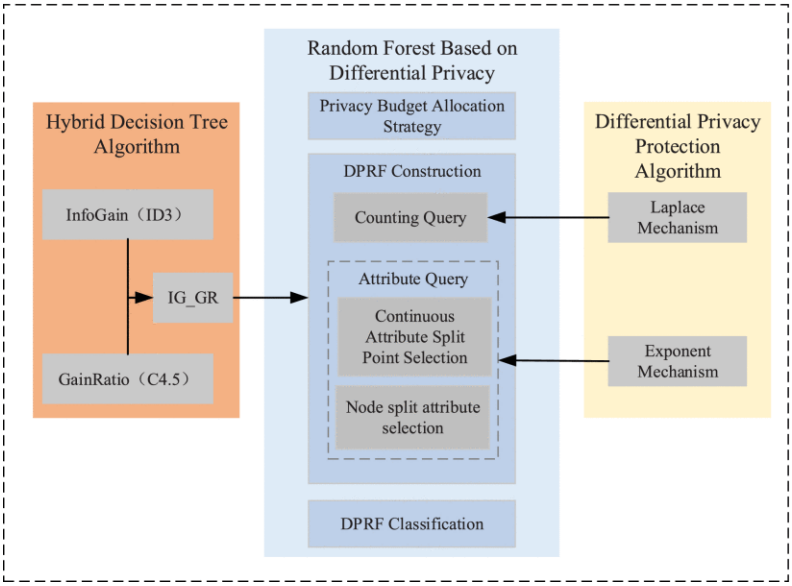


Figure 4.4: Architectural diagram of the differentially private random forest algorithm retrieved from Hou *et al.* [46].

Differential privacy is obtained by utilizing the Laplace and exponent mechanisms [46]. Firstly, a privacy budget is determined in the DPRF algorithm, then Laplacian noise is injected in the query counting of the DPRF algorithm. In the hybrid decision tree algorithm, the node splitting attribute is chosen based on the gained information, called *InfoGain*. The ratio between this value and the value of the continuous attribute splitting point is called the *GainRatio*. Further, by linearly combining these two attributes, this output can be inserted to the DPRF algorithm. In the query attribute in the DPRF algorithm, exponent noise is added. Lastly, a DPRF classifier is obtained.

Two processes define the differentially private random forest algorithm: a random forest is constructed and a test set classifier is developed [46]. Briefly, the training data set is utilized to generate an algorithm of random forest that obtains DP. From this algorithm, a test set can be classified, and for every sample of data, the classification results are returned.

Chapter 5

Related Work

To examine the privacy-utility-trade-off, three papers are selected and presented below. Their implementation choices and results are described here, and discussed in chapter 6.

5.1 Paper: End-to-End Privacy Preserving Deep Learning on Multi-Institutional Medical Imaging

Kaissis *et al.* [47] developed an open-source framework called privacy-preserving medical image analysis (PriMIA) [48] and tested it on a real-life case study. A deep convolutional neural network (CNN) was used on a data set containing chest X-rays. The aim was to develop a differentially private federated model which could be used as a classifier on medical images [47].

In the test scenario of the PriMIA framework, three hospitals were considered as data owners, training their data on a deep neural network model [47]. For the training phase, the threat against the model was assumed to be honest-but-curious. As mentioned in section 2.8, this means that the attacker will try to learn or gain private information from the data set, without tackling with the learning protocol.

Both differential private stochastic gradient descent (DP-SGD) and securely aggregated DP are used in the PriMIA framework [47]. In DP secure aggregation the statistics of the training, also known as the mean and standard deviation, are aggregated [47] [1]. On each individual node, there is used a DP procedure of query to avert leakage of private data. The value of ϵ determines the amount of Laplacian noise added to the statistics and is defined by the user.

DP-SGD is performed with gradient descent in the neural network training [47] [2]. Two modifications are applied to obtain (ϵ, δ)-DP: the global gradient norm is clipped and the bounded gradient is added with random Gaussian noise before the gradient descent step is performed.

[1] Taken from the *Supplementary Information* to the paper [47].

[2] See footnote 1.

The federated training process in the paper begins with an untrained model which is distributed to the data owners to be used in local training [47]. When the training converges, the local model securely averages, and an update is sent back to the server. Subsequently, the global model preforms an update based on all the data owners local training and redistribute the new current model. After all the iterations of training is completed, a securely aggregated model is obtained in the server.

It is assumed that for each training data set, patients are only included once [47]. In PriMIA, local DP is implemented at each data owner to guarantee patient-level privacy. The Rényi differential privacy accountant [49] is used to determine the privacy budget of each node. Additionally, by using secure multi-party computation (SMPC) [50], different parties may contribute to the training without revealing any of their individual inputs [47]. This is depleted by using secure aggregation on the weighted updates.

A pretrained model and a publicly available data set was used on the DP-SGD algorithm to tune the parameters [47] [3]. Transfer learning can be utilized to *freeze* large amounts of the model, and fewer training steps are needed to reach convergence. As a consequence, less noise can be added to the data. To avoid depletion of the privacy budget, they have not used hyperparameter optimization runs.

A data set containing about 100,000 chest X-rays was used in the study [47] [4]. Here, it is modified to classify "normal" and "abnormal" radiographs and divides the test set into a validation set containing 11,211 images and a test set containing 14,384 images.

Accidentally leaked information, δ, is set to 1.9×10^{-4} while the privacy budget, ϵ, varies between $0 - 10$ [47] [5]. This means that when ϵ reaches 10, the training stops automatically. Three different models were used in the training. Firstly, the ResNet18 architecture [51] is trained from scratch. Then, the same model is pretrained on ImageNet [52]. Thirdly, the ResNet18 model is pretrained on the data set containing chest X-rays to converge.

Training utility based on different values of ϵ from $0 - 10$ is presented in figure 5.1. Here, the green curve represents the model trained from scratch, the blue curve represents the model pretrained on ImageNet, while the purple curve represents the model pretrained on the chest X-ray data set. The training utility is represented by the Matthews correlation coefficient (MCC) [53]. If the value of the MCC is equal to -1, it means that the prediction is completely wrong, and if it is equal to 1, it means that the classification was completely right.

The results prove that the untrained model provides a lower utility, while the model pretrained on the chest X-ray data set utilizes the highest utility. Specifically, a MCC equal to 0.78 is achieved when ϵ is equal to 6 for the model pretrained on the chest X-ray data set.

[3] Taken from the *Supplementary Information* to the paper [47].

[4] See footnote 3.

[5] See footnote 3.

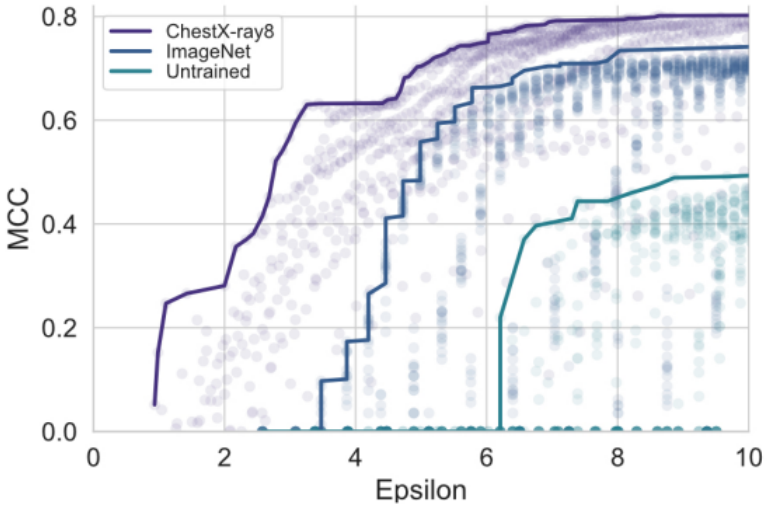


Figure 5.1: Results demonstrating the trade-off between utility and privacy budget, ϵ, retrieved from the Supplementary Information from Kaissis *et al.* [47].

5.2 Paper: Evaluating Differentially Private Machine Learning in Practice

Jayaraman and Evans [54] utilizes both logistic regression and neural network models to investigate the privacy-utility-trade-off. To obtain differential privacy, they use empirical risk minimization for the logistic regression model, and non-convex learning for the neural network model. Four different variations of DP is implemented:

1. Naïve composition (NC)
2. Advanced composition (AC)
3. Zero-concentrated differential privacy (zCDP)
4. Rényi differential privacy (RDP)

To evaluate the accuracy loss of the model, they utilized a baseline model that is non-private [54]. This accuracy loss can be defined as in equation 5.1.

$$\text{Accuracy Loss} = 1 - \frac{\text{Accuracy of Private Model}}{\text{Accuracy of Non-Private Model}} \tag{5.1}$$

An attacker holds 10,000 records from both the training set and the test set [54]. The training set records are labeled *members*, while the rest is labeled *non-members*. Given an input, the attacker will predict if the record is a member or a non-member.

This paper utilizes the CIFAR-100 [55] data set. 10,000 images are randomly selected for both the training set and the test set [54]. Further, the training exploit the l_2 sensitivity, along with $\delta = 10^{-5}$. Since the inverse of 10,000 is larger

than the value of δ, the requirements are satisfied. Gradient perturbation is implemented to train the DP models, where the values $0.01 - 1,000$ are used for ϵ. Additionally, the training utilizes the ADAM optimizer and a learning rate set to 0.01.

Per-instance clipping is performed in the gradient perturbation [54]. This is more efficient with respect to the privacy budget than batch clipping. The TensorFlow Privacy framework [56] is implemented with a threshold $C = 1$.

For the logic regression baseline model, the accuracy of the training set and test set reaches respectively 0.225 and 0.155 [54]. Figure 5.2 depicts the accuracy loss for the different variations when the privacy budget, ϵ, varies for the logic regression model. Since there are 100 classes in the CIFAR-100 data set, and the naïve composition performs around 0.01 when ϵ equals to 10 or less, this means that the NC is randomly guessing. This variation does not achieve zero accuracy loss until ϵ is close to 1,000. When ϵ is 100 or more, too much noise is added in the advanced composition to make use of it. For the zCDP and RDP much less noise is necessary. Therefore, the accuracy loss is about 0 for zCDP when $\epsilon = 500$ and for RDP when $\epsilon = 50$.

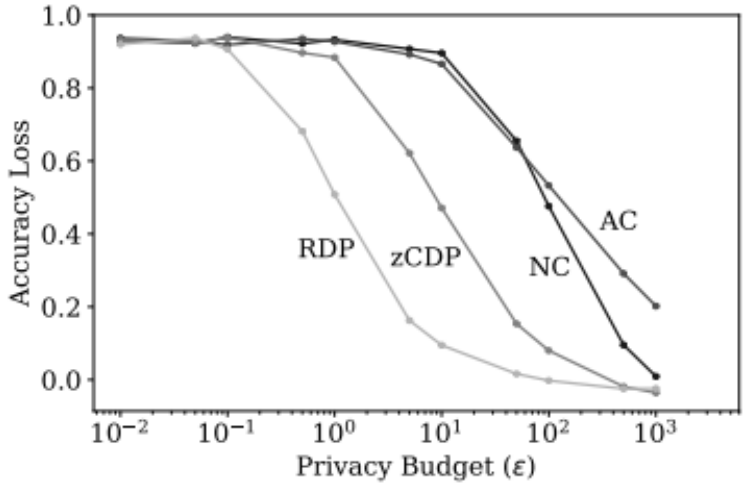


Figure 5.2: Results demonstrating the trade-off between the accuracy loss and the privacy budget, ϵ, for a logistic regression model retrieved from Jayaraman and Evans [54].

The architecture of the neural network model includes an output layer as well as two hidden layers [54]. A ReLU activation is used on the 256 neurons in the hidden layers. A softmax layer is utilized as the output layer, consisting of 100 neurons representing each class label. For the neural network baseline model, the accuracy of the training set and test set reaches respectively 1.000 and 0.168 [54].

Figure 5.3 illustrates the accuracy loss for different values of ϵ for the neural network model. For all the values of ϵ less than 100, both the naïve composition and the advanced composition are not useful at all. However, for the same values

of ϵ, the accuracy loss of zCDP and RDP are around 0.24. In this setting, zero accuracy loss is not accomplished for any of the variations.

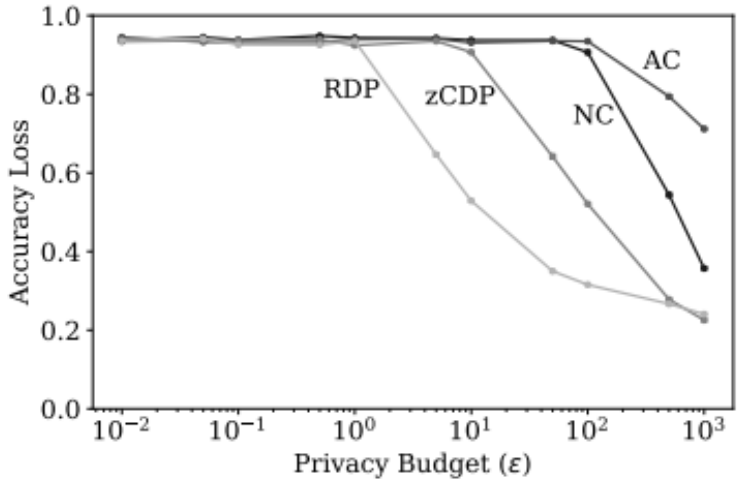


Figure 5.3: Results demonstrating the trade-off between the accuracy loss and the privacy budget, ϵ, for a neural network model retrieved from Jayaraman and Evans [54].

5.3 Paper: Deep Learning with Differential Privacy

To examine the trade-off between accuracy and privacy budget, Abadi *et al.* [40] utilizes the differential privacy stochastic gradient descent algorithm on both the CIFAR-10 [57] and MNIST [58] data sets. TensorFlow [59] is utilized to implement the algorithm with neural network models.

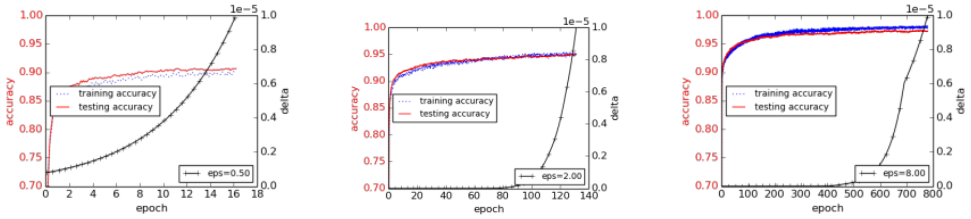
Given a set of parameters and an empirical loss function, the model is trained by minimizing this loss [40]. A random subset of samples is selected for every step of the SGD, and their gradient is computed and clipped. Further, the average is computed, and noise is injected. Noise adding is performed by the Gaussian noise mechanism. The privacy accountant is exploited to compute the privacy loss.

Each time the training data is accessed, it leads to a cost in the privacy which can be computed by an accountant [40]. During the progression of the training, the privacy cost is accumulated. Since the gradients are computed at multiple layers for each step in the training process, the privacy cost of all these gradients needs to be accumulated by the accountant. Additionally, a sanitizer is implemented to clip the gradient norm and inject noise before the parameters of the network are updated.

To capture the input data's main features, the principal component analysis (PCA) method can be utilized [40]. Although the use of PCA results in privacy cost, the quality of the model is improved as well as the training time is reduced.

For both of the experiments, Abadi *et al.* [40] chose $\delta = 10^{-5}$, while the privacy budget, ϵ, is computed as a function of E, which is the training epoch. The first experiment exploits the MNIST data set containing 60,000 training samples and 10,000 test samples. Here, a feed forward neural network is implemented containing 10 classes in a softmax layer. Additionally, a ReLU unit is utilized as an activation layer as well as an input layer consisting of a PCA. The baseline model achieves after 100 epochs an accuracy of 98.30%.

Figure 5.4 depicts the results from the MNIST experiment, where figure 5.4a, 5.4b and 5.4c corresponds to the cases where ϵ is respectively equal to 0.50, 2.00 and 8.00. The figures illustrates that the accuracy achieves 90%, 95% and 97% for these values of ϵ.

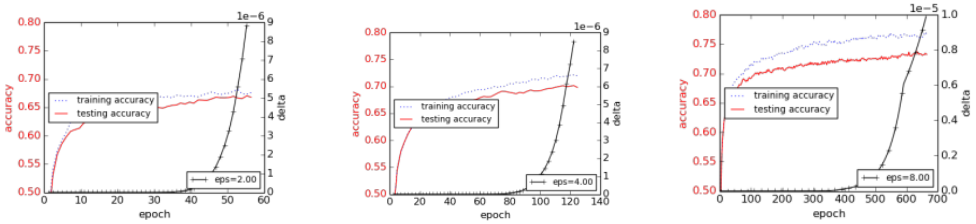


(a) Results when $\epsilon = 0.50$. **(b)** Results when $\epsilon = 2.00$. **(c)** Results when $\epsilon = 8.00$.

Figure 5.4: Illustration of the accuracy results for different values of ϵ on the MNIST data set retrieved from Abadi *et al.* [40].

The second experiment of Abadi *et al.* [40] exploits the CIFAR-10 data set, which consists of 50,000 training samples and 10,000 test samples. A convolutional neural network model is utilized involving two layers of convolution and two layers that are fully connected. Additionally, a ReLU layer is utilized as well as max pools. As a baseline, without obtaining privacy, this architecture achieves around 86% accuracy in epoch 500.

For this experiment, pre-trained convolutional layers and fully connected layers are utilized [40]. Figure 5.5 illustrates the results from the CIFAR-10 experiment. In figure 5.5a, 5.5b and 5.5c, ϵ has a value of respectively 2.00, 4.00 and 8.00. Here, there is a difference around 7% in accuracy between the baseline and the DP model, while for the MNIST experiment the difference is around 1.3%.



(a) Results when $\epsilon = 2.00$. **(b)** Results when $\epsilon = 4.00$. **(c)** Results when $\epsilon = 8.00$.

Figure 5.5: Illustration of the accuracy results for different values of ϵ on the CIFAR-10 data set retrieved from Abadi *et al.* [40].

Chapter 6

Discussion

6.1 Federated Learning

Today, federated learning is utilized in e.g. healthcare research and by service providers. For example, FL was utilized by Google in 2016 to increase prediction accuracy for keyboard inputs from Android mobiles in a private manner [2]. Additionally, a multi-FL network was developed to collect real health data, which hopefully can be utilized in the future by doctors. Other classification tasks are performed with FL, such as detection of COVID-19, diagnosis of cancer, and autism spectrum disorder. Therefore, with FL patients can be treated earlier, which is a huge benefit for doctors and enabling them insights in risks.

However, even though FL gives rise to several benefits in some areas, it is still a relatively new discipline [2]. Multiple variations of FL have been developed aiming to improve the approach, however, as stated in section 2.6, there exists several challenges within the field. Indeed, federated learning is a current working field, and some questions are not yet answered. Some of them are listed down below.

- Required extent of communication
- Reduction of communication between federated training techniques
- Further study of asynchronous approaches and bulk synchronous techniques
- Prediction of federated networks' amount of heterogeneity
- Further study of the limitations of mixed privacy
- Scalability, heterogeneity, and privacy problems beyond supervised learning
- Practical issues within FL production
- Improvement of benchmarking tools and implementations

One of the main concerns in FL is attacks. The approach per say might obtain GDPR, however, there exists no defence mechanisms against attackers. Differential privacy mechanisms on the other hand, are more robust against attackers. Therefore, by combining the FL approach with DP, it can be sturdier against adversaries.

6.2 Differential Privacy

When differential privacy is provided in machine learning, the privacy-utilization-trade-off is affected by the choice of classifier and method. Results from Gong *et al.* [35] reveals that when privacy is not preserved with a linear regression classifier, the model achieves 86% accuracy. When $\epsilon = 1.6$ and $\epsilon = 0.05$, their privacy preserving algorithm achieves an accuracy of respectively 85% and 83%. These results indicates that even with a smaller ϵ, which means stronger privacy, the utility of the model does not decrease significantly.

Another experiment with both linear and logistic regression achieves other results. Zhang *et al.* [37] presents the differentially private m-estimators (DPME) approach [60], the filter-priority (FP) approach [61] and the functional mechanism. Here, it is clear that with decreasing ϵ, both FP and DPME achieves a larger amount of errors. The main reason for this result is because a smaller ϵ corresponds to more noise injection. Specifically, for $\epsilon = 3.2$ the mean square error for FM and the non-private approach is equal to 0.1, while for FP and DPME it is equal to 0.15. However, when $\epsilon = 0.1$, FM has approximately the same mean square error, but FP and DPME has a mean square error equal to approximately 0.35. These results clearly reveal that the impact on the privacy-utility-trade-off depends on the methods that are chosen. Even with the FM method, the mean square error increases when ϵ decreases significantly, but not to the same extent as the two other methods. Therefore, the choice of method is an important part of the study.

Zhang *et al.* [43] found similar results when preserving differential privacy with a support vector machine classifier. When $\epsilon = 1$, their proposed algorithm achieves 90% accuracy, which is the same as the non-private approach. However, when $\epsilon = 0.05$ and $\epsilon = 0.005$, their approach achieves respectively 85% and 65%. Also, here the smaller the ϵ, the more Laplacian noise is added, resulting in a lower accuracy. Clearly, this experiment reveals that to obtain an acceptable performance of the algorithm, the privacy budget has to be large enough.

Unlike the other experiments, Hou *et al.* [46] found out that for a differentially private random forest algorithm, the accuracy remains approximately steady for the chosen values of ϵ. The results reveal that the non-private algorithm achieves 87.5% accuracy and the DP random forest achieves around 85%. Even though there is a decrease in accuracy from the non-private to the privately preserved algorithm, as well as the privately preserved algorithm does in fact increase slightly in accuracy when ϵ increases, it appears that there is not a significant trade-off in privacy and utility with this algorithm.

However, in the case of deep neural network models, Gong *et al.* [34] proves that there exists a privacy-utility-trade-off. All the different approaches to obtain DP stated in section 4.3 [10, 34, 38–41] was tested on five different data sets: Adult, MNIST [58], CIFAR-10 [57], DCCC and MIMIC-III. This experiment reveals that all the methods yield a high accuracy for larger values of ϵ, which correlates to models with less noise injected. Results unveil that almost all the methods achieve

accuracy over 80% on almost all the data sets when $\epsilon = 8$. However, for $\epsilon = 0.2$, most methods decrease with around 10%.

From all these experiments, it is clear that the privacy-utility-trade-off truly depends on the chosen algorithm. Some of the experiments implies that some classification algorithms can achieve less trade-off. However, it is also revealed that the approach chosen to obtain differential privacy with the classification algorithm does play an important role. Additionally, these studies have chosen a selection of ϵ ranging from as low as 0.0005 to as high as 8. This can impact the results. For example, the study of the random forest classifier obtained results showing that there was no significant trade-off between privacy and utility. However, they only used values of ϵ from 0.1 to 1.0. If they had used even lower or even higher values of ϵ, the results might be different.

6.3 Experiments from Related Work

From the experiment presented in section 5.1 it is clear that a pretrained model provides less trade-off between privacy and utility. The reason for this is that when a model is trained from scratch, a large part of the privacy budget is consumed. However, by utilizing a pretrained model, the budget can be used to only fine tune the parameters.

A relatively high variation of values of ϵ is explored in the experiments from section 5.2. It is demonstrated that some variations of differential privacy provides higher utility with lower values of ϵ. However, for the logistic regression model, only one variation obtains an accuracy lower than 20% with ϵ equal to around 10, while the other variations needs ϵ equal to around $100 - 1000$. This is quite high values of ϵ which in turn will provide a lower noise injection. Therefore, when utilizing such high values of ϵ, the model would be more vulnerable against inference attacks [54]. For the neural network model, the results are even worse. As a consequence, to gain an acceptable utility, a relatively high value of ϵ is required.

The study from section 5.3 indicates that if it is possible to obtain a high value of ϵ, the results can be approximately the same when privacy is preserved compared with no privacy preservation. Only when ϵ decreases towards a low value, the trade-off between privacy and utility becomes a problem. Again, this is due to more noise being added as well as it depends on how the privacy budget is utilized.

Even though several experiments prove the trade-off between utility and privacy, DP is applicated today by large companies such as Apple, Facebook, and Amazon. Apple utilizes local differential privacy in order to gain knowledge about the user community [62], Facebook gathers data from users' behavior to advertise targeted campaigns [63], and Amazon gathers shopping preferences from their users [64]. This proves that many big companies spend time and money on privacy research, in specific differential privacy. Even though the methods are not perfect today, they can still be utilized, and they are under constant development.

Chapter 7

Conclusion and Future Work

Although the concept of federated learning ensures privacy as required from the general data protection regulation, it is clear that there still are some challenges with the approach. However, in combinations with other privacy preserving algorithms such as differential privacy, the FL approach can be useful. Therefore, federated learning is an outstanding concept requiring further research.

Even though the definition of differential privacy has already existed for several years, the mechanisms that obtains this privacy are fairly new. However, the approaches are already utilized by big companies and are under constant developments. One of the main challenges in DP is that privacy can come at the expense of utility. Previous research proves that the privacy-utility-trade-off is still an open question today. Some studies reveal no trade-off, while other reveals a large trade-off.

In general, it can be difficult to quantify which results are acceptable and not. Additionally, a limit for acceptance may be chosen on beforehand by superiors, such as the government. In some experiments values are arbitrarily chosen in hope that they are suitable. Therefore, it is demanding to choose a general value of ϵ which provides acceptable privacy. Because the mechanisms utilized to obtain DP is relatively new, analysts have to try out different values to gain knowledge about the consequences. Additionally, it has been proven that the values of ϵ obtaining an acceptable amount of privacy really depends on the choices that are made in the implementation. Further research should aim to improve the differential privacy mechanisms in order to achieve less privacy-utility-trade-off.

Bibliography

[1] Z. Ji, Z. C. Lipton and C. Elkan, *Differential privacy and machine learning: A survey and review*, Accessed 28.04.22, 2014. [Online]. Available: https://arxiv.org/abs/1412.7584.

[2] S. Bharati, M. R. H. Mondal, P. Podder and V. S. Prasath, 'Federated learning: Applications, challenges and future directions,' *International Journal of Hybrid Intelligent Systems*, vol. 18, no. 1–2, pp. 19–35, 2022. DOI: 10.3233/his-220006.

[3] B. McMahan and D. Ramage, *Federated learning: Collaborative machine learning without centralized training data*, Accessed 02.05.22, 2017. [Online]. Available: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

[4] B. Jayaraman and D. Evans, *Evaluating differentially private machine learning in practice*, Accessed 29.04.22, 2019. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/jayaraman.

[5] Q. Yang, Y. Liu, T. Chen and Y. Tong, 'Federated machine learning: Concept and applications,' *ACM Trans. Intell. Syst. Technol. 10*, 2019. DOI: https://doi.org/10.1145/3298981.

[6] J. Konecný, B. McMahan, D. Ramage and P. Richtárik, *Federated optimization: Distributed machine learning for on-device intelligence*, Accessed 04.05.22, 2016. [Online]. Available: https://arxiv.org/pdf/1610.02527.pdf.

[7] P. Kairouz, H. B. McMahan and B. Avent, 'Advances and open problems in federated learning,' *Foundations and Trends in Machine Learning*, vol. 4, no. 1, 2019. DOI: https://doi.org/10.48550/arXiv.1912.04977.

[8] I. Kholod, E. Yanaki, D. Fomichev, E. D. Shalugin, E. Novikova, E. Filippov and M. Nordlund, 'Open-source federated learning frameworks for iot: A comparative review and analysis,' *Evolution of Distributed Computing in Sensor Systems*, vol. 21, no. 1, 2021. DOI: https://doi.org/10.3390/s21010167.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson and B. A. Arcas, 'Communication-efficient learning of deep networks from decentralized data,' *Proceedings of the 20 th International Conference on Artificial Intelligence and Statistics*,

vol. 54, pp. 3–5, 2017. DOI: https://doi.org/10.48550/arXiv.1602.05629.

[10] R. Shokri and V. Shmatikov, 'Privacy-preserving deep learning,' 2015. DOI: http://dx.doi.org/10.1145/2810103.2813687.

[11] Geeksforgeeks, *Difference between batch gradient descent and stochastic gradient descent*, Accessed 27.05.22, 2022. [Online]. Available: https://www.geeksforgeeks.org/difference-between-batch-gradient-descent-and-stochastic-gradient-descent/.

[12] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson and M. Jirstrand, 'A performance evaluation of federated learning algorithms,' *Proceedings of the Second Workshop on Distributed Infrastructures for Deep Learning*, pp. 1–8, 2018. DOI: https://doi.org/10.1145/3286490.3286559.

[13] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough and V. Saligrama, 'Federated learning based on dynamic regularization,' 2021. DOI: https://doi.org/10.48550/arXiv.2111.04263.

[14] S. Vahidian, M. Morafah and B. Lin, 'Personalized federated learning by structured and unstructured pruning under data heterogeneity,' 2021. DOI: https://doi.org/10.48550/arXiv.2105.00562.

[15] S. Paul, *Pruning in deep learning model*, Accessed 29.05.22, 2020. [Online]. Available: https://medium.com/@souvik.paul01/pruning-in-deep-learning-models-1067a19acd89.

[16] I. Consulting, *General data protection regulation gdpr*, Accessed 30.05.22, 2018. [Online]. Available: https://gdpr-info.eu/.

[17] N. Truong, K. Sun, S. Wang, F. Guitton and Y. Guo, 'Privacy preservation in federated learning: An insightful survey from the gdpr perspective,' 2020. DOI: https://doi.org/10.48550/arxiv.2011.05411.

[18] T. Li, A. K. Sahu, A. Talwalkar and V. Smith, 'Federated learning: Challenges, methods, and future directions,' *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020. DOI: https://doi.org/10.48550/arXiv.1908.07873.

[19] L. Lyu, H. Yu and Q. Yang, 'Threats to federated learning: A survey,' 2020. DOI: https://doi.org/10.48550/arXiv.2003.02133.

[20] C. Dwork and A. Roth, 'The algorithmic foundations of differential privacy,' *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014. DOI: 10.1561/0400000042.

[21] A. Blum, C. Dwork, F. McSherry and K. Nissim, 'Practical privacy: The sulq framework,' *24th ACM SIGMOD International Conference on Management of Data / Principles of Database Systems, Baltimore*, pp. 128–138, 2005.

[22] S. Fathima, *Differential privacy definition*, Accessed 19.05.22, 2020. [Online]. Available: https://medium.com/@shaistha24/differential-privacy-definition-bbd638106242.

[23] I. Mironov, 'Rényi differential privacy,' *IEEE*, pp. 263–275, 2017. DOI: https://doi.org/10.1109/CSF.2017.11.

[24] S. Fathima, *Global vs local differential privacy*, Accessed 01.06.22, 2020. [Online]. Available: https://medium.com/@shaistha24/global-vs-local-differential-privacy-56b45eb22168.

[25] S. Fathima, *Differential privacy — noise adding mechanisms*, Accessed 31.05.22, 2020. [Online]. Available: https://becominghuman.ai/differential-privacy-noise-adding-mechanisms-ede242dcbb2e.

[26] M. Lyu, D. Su and N. Li, 'Understanding the sparse vector technique for differential privacy,' 2016. DOI: https://doi.org/10.48550/arxiv.1603.01699.

[27] M. Bun and T. Steinke, 'Concentrated differential privacy: Simplifications, extensions, and lower bounds,' 2016. DOI: https://doi.org/10.48550/arxiv.1605.02065.

[28] C. Dwork and G. N. Rothblum, 'Concentrated differential privacy,' 2016. DOI: https://doi.org/10.48550/arxiv.1603.0188.

[29] P. Rigollet, 'Chapter 1: Sub-gaussian random variables,' *Lecture Notes from High-Dimensional Statistics at MIT*, pp. 14–32, 2015.

[30] A. K. V. Machanavajjhala, D. Kifer, J. E. Gehrke and M. Venkitasubramaniam, 'L-diversity: Privacy beyond k-anonymity,' *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, pp. 3–8, 2007. DOI: https://doi.org/10.1145/1217299.1217302.

[31] N. Li, T. Li and S. Venkatasubramanian, 'T-closeness: Privacy beyond k-anonymity and l-diversity,' *2007 IEEE 23rd International Conference on Data Engineering*, 2007. DOI: 10.1109/ICDE.2007.367856.

[32] R. C. Wong, J. Li, A. W. Fu and K. Wang, '(alpha,k)-anonymity: An enhanced k-anonymity model for privacy-preserving data publishing,' *ACM SIGKDD international conference on knowledge discovery data mining*, pp. 754–759, 2006.

[33] X. Xiao and Y. Tao, 'M-invariance: Towards privacy preserving re-publication of dynamic datasets,' *Proceedings of the ACM SIGMOD international conference on management of data*, pp. 689–700, 2007.

[34] M. Gong, K. Pan, Y. Xie, A. K. Qin and Z. Tang, 'Preserving differential privacy in deep neural networks with relevance-based adaptive noise imposition,' *Neural Networks*, vol. 125, pp. 131–141, 2020. DOI: https://doi.org/10.1016/j.neunet.2020.02.001.

[35] M. Gong, K. Pan and Y. Xie, 'Differential privacy preservation in regression analysis based on relevance,' *Knowledge-Based Systems*, vol. 173, pp. 140–149, 2019. DOI: https://doi.org/10.1016/j.knosys.2019.02.028.

[36] S. C. Nevrella, *Linear regression*, Accessed 02.06.22, 2021. [Online]. Available: https://saichandra1199.medium.com/linear-regression-1e279814e2bb.

[37] J. Zhang, Z. Zhang, X. Xiao, Y. Yang and M. Winslett, 'Functional mechanism: Regression analysis under differential privacy,' *Proceedings of the VLDB Endowment (PVLDB)*, vol. 5, no. 11, pp. 1364–1375, 2012. DOI: https://doi.org/10.48550/arxiv.1208.0219.

[38] N. Phan, Y. Wang, X. Wu and D. Dou, 'Differential privacy preservation for deep auto-encoders: An application of human behavior prediction,' *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pp. 1309–1316, 2016.

[39] N. Phan, X. Wu and D. Dou, 'Preserving differential privacy in convolutional deep belief networks,' *Machine Learning*, vol. 106, pp. 1681–1704, 2017. DOI: https://doi.org/10.1007/s10994-017-5656-2.

[40] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar and L. Zhang, 'Deep learning with differential privacy,' pp. 308–318, 2016. DOI: DOI:http://dx.doi.org/10.1145/2976749.2978318.

[41] L. Xie, K. Lin, S. Wang, F. Wang and J. Zhou, 'Differentially private generative adversarial network,' 2018. DOI: https://doi.org/10.48550/arxiv.1802.06739.

[42] R. Gandhi, *Support vector machine — introduction to machine learning algorithms*, Accessed 02.06.22, 2018. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

[43] Y. Zhang, Z. Hao and S. Wang, 'A differential privacy support vector machine classifier based on dual variable perturbation,' *IEEE*, vol. 7, 2019. DOI: https://doi.org/10.1109/ACCESS.2019.2929680.

[44] J. C. Platt, 'Sequential minimal optimization: A fast algorithm for training support vector machines,' *Microsoft Research*, 1998.

[45] M. Pal, 'Random forest classifier for remote sensing classification,' *International Journal of Remote Sensing*, vol. 26, no. 1, pp. 217–222, 2007. DOI: https://doi.org/10.1080/01431160412331269698.

[46] J. Hou, Q. Li, S. Meng, Z. Ni, Y. Chen and Y. Liu, 'Dprf: A differential privacy protection random forest,' *IEEE Access*, vol. 7, pp. 130 707–130 720, 2019. DOI: 10.1109/ACCESS.2019.2939891.

[47] G. Kaissis, A. Ziller and J. Passerat-Palmbach, *End-to-end privacy preserving deep learning on multi-institutional medical imaging*, Accessed 14.05.22, 2021. [Online]. Available: https://www.nature.com/articles/s42256-021-00337-8.

[48] G. Kaissis, *Primia*, Accessed 14.05.22, 2020. [Online]. Available: https://github.com/gkaissis/PriMIA.

[49] I. Mironov, K. Talwar and L. Zhang, 'Rényi differential privacy of the sampled gaussian mechanism,' 2019. DOI: https://doi.org/10.48550/arxiv.1908.10530.

[50] G. A. Kaissis, M. R. Makowski, D. Rückert and R. F. Braren, 'Secure, privacy-preserving and federated machine learning in medical imaging,' *Nature Machine Intelligence*, vol. 2, pp. 305–311, 2020. DOI: https://doi.org/10.1038/s42256-020-0186-1.

[51] P. Team, *Resnet*, Accessed 18.05.22. [Online]. Available: https://pytorch.org/hub/pytorch_vision_resnet/.

[52] L. Fei-Fei, J. Deng, O. Russakovsky, A. Berg and K. Li, *Imagenet*, Accessed 18.05.22, 2020. [Online]. Available: https://image-net.org.

[53] 'The matthews correlation coefficient (mcc) is more informative than cohen's kappa and brier score in binary classification assessment,' *IEEE*, vol. 9, pp. 78 368–78 381, 2021. DOI: 10.1109/ACCESS.2021.3084050.

[54] B. Jayaraman and D. Evans, 'Evaluating differentially private machine learning in practice,' 2019. DOI: https://doi.org/10.48550/arxiv.1902.08874.

[55] TensorFlow, *Cifar100*, Accessed 06.06.22, 2022. [Online]. Available: https://www.tensorflow.org/datasets/catalog/cifar100.

[56] G. Andrew, S. Chien and N. Papernot, *Tensorflow privacy*, Accessed 06.06.22, 2019. [Online]. Available: https://github.com/tensorflow/privacy.

[57] TensorFlow, *Cifar10*, Accessed 07.06.22, 2022. [Online]. Available: https://www.tensorflow.org/datasets/catalog/cifar10.

[58] TensorFlow, *Mnist*, Accessed 07.06.22, 2022. [Online]. Available: https://www.tensorflow.org/datasets/catalog/mnist.

[59] TensorFlow, *An end-to-end open source machine learning platform*, Accessed 07.06.22, 2022. [Online]. Available: https://www.tensorflow.org/.

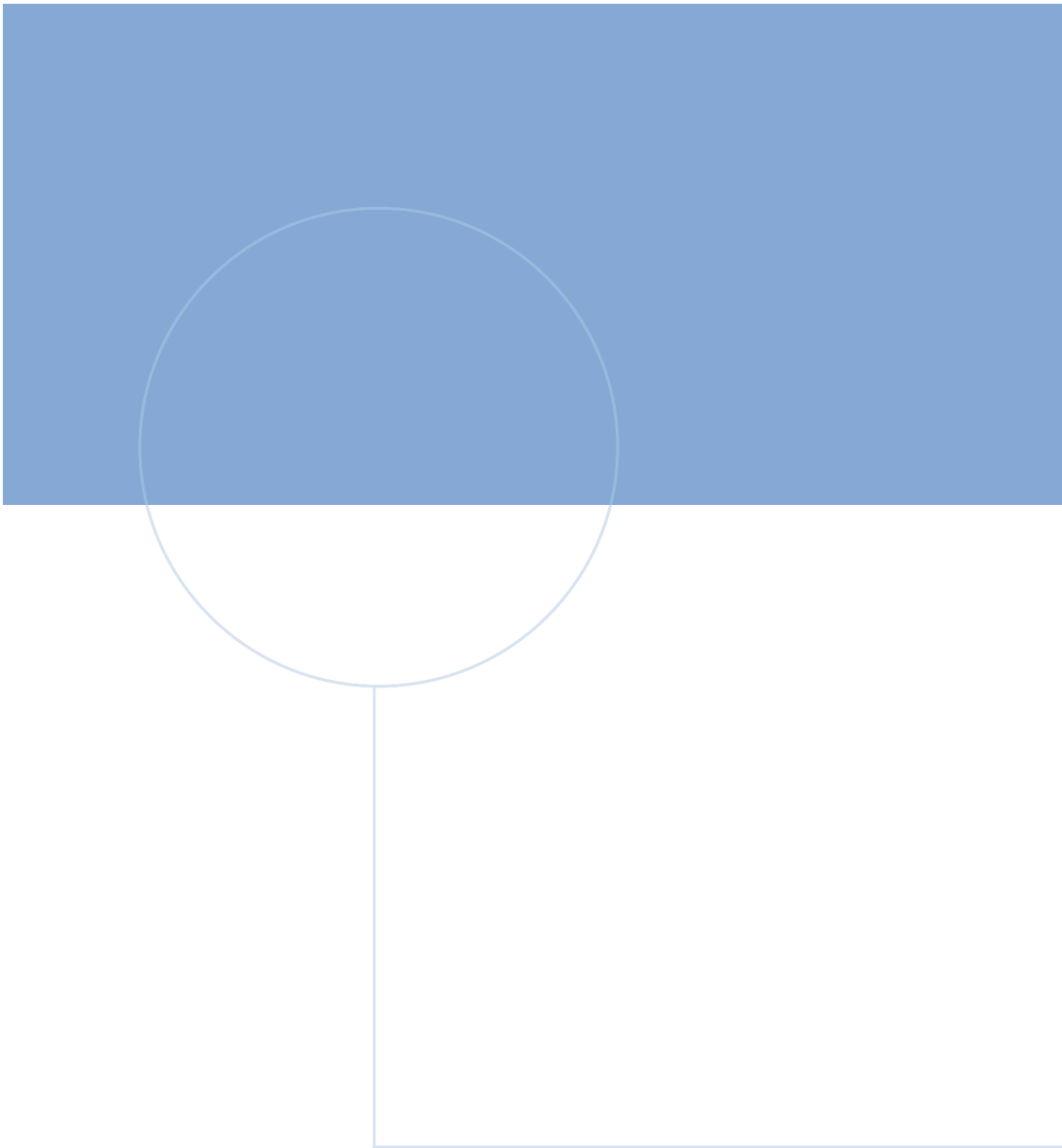
[60] J. Lei, 'Differentially private m-estimators,' *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, 2011.

[61] G. Cormode, M. Procopiuc, D. Srivastava and T. T. L. Tran, 'Differentially private publication of sparse data,' *Proceedings of the 15th International Conference on Database Theory*, 2012.

[62] Apple, 'Differential privacy,' *apple.com*,

[63] C. Nayak, *New privacy-protected facebook data for independent research on social media's impact on democracy*, Accessed 12.06.22, 2020. [Online]. Available: https://research.facebook.com/blog/2020/02/new-privacy-protected-facebook-data-for-independent-research-on-social-medias-impact-on-democracy/.

[64] J. Majmudar, C. Dupuy, C. Peris, S. Smaili, R. Gupta and R. Zemel, *Differentially private decoding in large language models*, Accessed 12.06.22, 2022. [Online]. Available: https://www.amazon.science/publications/differentially-private-decoding-in-large-language-models.



NTNU

Kunnskap for en bedre verden