Université du Québec
Institut national de la recherche scientifique
Centre Énergie, Matériaux, et Télécommunications

# Differential Privacy in Deep Learning Models

By

Mehdi Amian

A thesis submitted in partial fulfillment for the degree of
*Master of Sciences*, M.Sc.
in Telecommunications

**Jury members**

| | |
|---|---|
| External examiner | Mustapha Kardouchi, PhD<br>University of Moncton |
| Internal examiner | Douglas O'Shaughnessy, PhD<br>INRS-EMT |
| Supervisor | Amar Mitiche, PhD<br>INRS-EMT |
| Co-Supervisor | Jacob Benesty, PhD<br>INRS-EMT |

# Remerciements

Je remercie le directeur de recherche, le professeur Amar Mitiche, et le co-directeur de recherche, le professeur Jacob Benesty de leur supervision et leur soutien concernant du présent projet. Je remercie aussi les membres du jury de l'évaluation de ce mémoire, le professeur Douglas O'Shaughness et le professeur Mustapha Kardouchi d'accorder du temps et de leurs commentaires constructifs.

# Résumé

Dans ce memoire, le sujet de la confidentialité dans les modèles d'apprentissage profond est étudié. Dans le contexte actuel, la confidentialité désigne à la protection des modèles qui sont publiquement partagés, contre la fuite non souhaitable des informations. Il se trouve que ces modèles sont vulnerables dans le sens où la mémorisation de données d'entraînement risque la divuglation des informations confidentielles par des utilisateurs malveillants. La nécessité et l'importance de ce projet viennent du fait qu'à nos jours, il y a des modèles d'apprentissage profond qui sont entraînés sur des données personnelles et potentiellement confidentielles, qui sont susceptibles d'être partagés publiquement. Ces modèles n'incluent généralement pas de méthodes permettant de protéger la confidentialité des données. D'autres méthodes offrent la confidentialité aux modèles d'apprentissage profond mais avec une dégradation inacceptable de la performance de modèle de point de vue pratique et surtout commercial. Dans ce projet, une modification à l'algorithme de descente de gradient stochastique différentiellement confidentiel (DGSDC) est présentée afin d'apporter la confidentialité aux modèles d'apprentissage profond avec une meilleure performance des modèles en offrant un niveau acceptable de confidentialité pour des utilisations pratiques et commerciales. La méthode proposée est basée sur la filtration par la fonction tangente hyperbolique (tanh) permettant de règler le problème d'explosion de gradient au niveau des réseaux de neurones récurrents (RNR).

**Mots-clés** apprentissage profond, confidentialité des données, modèles récurrents.

# Abstract

In this thesis, the concept of privacy in deep learning models is studied. Privacy here refers to protection of a publicly shared model against unwanted information leakage. It is found out that deep learning models are susceptible to memorizing the training data and consequently are at risk of disclosing confidential information in response of malignant inquiries. The necessity and importance of this project come from the fact that nowadays many deep learning models are trained on personal data with potentially confidential and sensitive contents and then are publicly shared. Such models naturally lack privacy protecting components. There are many examples of reversing the training path and getting access to the training data by adversaries. There are some approaches that offer the privacy in deep learning models but with an unacceptable decline in the performance of the model from a real world and particularly commercial point of view. In this study, an adjustment to the differentially private stochastic gradient descent (DPSGD) algorithm which is the state-of-the-art in facilitating privacy in deep learning models is proposed. The motivation is to improve the performance of the model while offering an acceptable amount of privacy for real life and commercial applications. The proposed approach is based on tanh filtering and can also be extended to the recurrent neural network (RNN) to solve the exploding gradient problem.

**Keywords** data, deep learning, model, privacy, utility.

# Contents

# List of figures

# List of Tables

# Chapter 1

# Introduction

In general, privacy might be confused with security as some overlap might be perceived in a general point of view. However, in computer science and in the context of data science, these two notions have different meanings. Basically, security refers to the case where two parties communicate with each other through a communication channel while no third party could reveal information. However, when it comes to privacy, there is no third party. Instead, if a party makes a query and collects more information than those they are supposed to get, it is referred to as breaching the privacy of the respondent party. For example, if individual A is asked about their hobbies, then by providing the response from A, knowledge will be obtained about the hobbies and probably beyond. If more information than hobbies themselves is acquired, then it will be said that the privacy of A is broken. For instance, if A talks about playing golf, going to church/mosque/temple on a regular basis, participating in some specific social/political gatherings regularly, one might infer other information than just the hobbies such as social, religious, or political interests/trends. More precisely, it might be concluded that A is rich because of playing golf. So, in this example, some unwanted information is leaked. The core discussion in privacy context is to prevent such information leakages.

By emerging big data bases and their availability on the internet, preserving the confidentiality of the people whose data are shared becomes more and more important. Advancing the technologies hired by hackers or cyber-attackers on the other hand, makes the issue more and more challenging. The privacy concerns are not limited to the shared data. Sharing the models can also be subject to identity leakage. Here, the model refers to deep learning models which are trained on personal

data with potentially confidential contents. The more training data and the more realistic data, the higher the performance of the model. So, to promote the performance of deep learning models, especially in commercial applications, more data and more realistic data which potentially contain confidential or sensitive information are used to train the models. This will lead to an increase in performance of the model but also raises the risk of confidential information disclosure. Thus, privacy components are required to be added to these models in order to protect the privacy of the data by blocking the reverse path towards the training data.

Basically, the privacy can be discussed in two contexts: the data and the model. As for the data, when a dataset associating with some people is publicly shared, it should be guaranteed that these people are not identifiable through any query over the data. A fundamental question is: why a big data set should be shared publicly? Here is the answer. Essentially, big data sets serve to train artificial neural networks. Today, thanks to powerful computer processors it became possible to train huge neural networks which are also called deep learning models with millions of parameters. On the other hand, such models require big datasets for training. The bigger the model, the bigger data dataset for training. Hence, the big data sets are collected and then shared publicly for research and development purposes. To preserve the privacy of individuals, the first step is to anonymize data, i.e., to remove names and any other unique identifiers. Is anonymization enough to ensure privacy? The answer is unfortunately negative. In fact, to preserve the privacy, anonymization is necessary but not sufficient. Therefore, other measures should be taken to protect the information leakage when sharing data and models with the public.

There are several famous stories in the history about information leakage when sharing data publicly. In the 1990s, in Massachusetts, the state group insurance commission published the hospital admission report of the state employees. They anonymized the data and then published it. However, a computer scientist could successfully re-identify many individuals including William Weld, the governor of Massachusetts by matching the public data set with a registry vote i.e. another publicly available data set. This was an example of publishing health data which is one of the main domains of sharing data. We are currently living in the COVID-19 era after which many data bases will be published for research and development purposes. Therefore, serious measures should be taken to protect the privacy of individuals whose data are published; otherwise, severe problems of re-identification similar to the Massachusetts example will not be far from expectation. Medical records are rigorously regulated in order to protect the patients' privacy. In practice, the health-

care data are strictly shielded under laws and regulations such as the Health Insurance Potability and Accountability Act or shortly HIPAA [15]. As an example from a different domain, Netflix published anonymized movie records of their subscribers as training data for a recommendation competition. The subscribers were re-identified as result of a matching between the dataset and the Internet Movie Database (IMDb).

The privacy is not limited to the data. It can be discussed on the context of sharing models as well. In fact, privacy on the model ensures that for a publicly shared model which is trained on possibly sensitive data, no query can disclose the confidential information. As an example, some writing recommendation applications like Gboard (Google mobile keyboard) [30], are trained on the data (text messages, search flow, emails, etc) of all users who use the application. In such applications, when the user starts typing, the next word is recommended by the model. Now, imagine if somebody types "Donald trump credit card number is", and then the model recommends the right number, then there will be a huge problem. Another example is Google's Smart Compose which is a commercial model for email composition recommendation that is trained on emails of millions of users [25].

These examples are to illustrate the importance and the necessity of developing privacy-preserving components for sharing data and models in public usages which is the core in this thesis.

Essentially, the privacy is a qualitative and abstract notion. However, when it comes to analysis and modelling, it becomes essential to settle quantitative grounds. There are several quantitative definitions for privacy among which the most popular ones are k-anonymity, l-diversity, t-closeness [36], and differential privacy (DP) [26, 27]. The DP is a well-suited definition of privacy that so far reflects the best the abstract notion of the privacy in a quantitative manner. Throughout this thesis, the DP is the quantitative measure of the privacy. To highlight the significance of DP in real world, it is useful to note that in 2020 US census it is decided to use DP to preserve the privacy of Americans [6]. In other words, they publish statistics in differentially private way. Also, Apple uses DP to ensure the privacy of their users when performing operations on data coming from their devices [6].

The thesis is organized as follows. The next chapter introduces the differential privacy essentials in terms of the preliminary required materials as well the formal definition in a detailed manner. Then, the concept of the differential privacy in the context of deep learning models is addressed.

After, a review of relevant and important previous research works is provided. The contribution of the thesis is described afterwards. Finally, the results as well as the interpretation and suggestions for future works are presented and discussed.

# Chapter 2

# Differential Privacy

Differential privacy (DP) promises that if data is publicly shared, no other study or analysis based on other information is able to affect the data. A main way of re-identification on a data set is to match with other datasets. So, provided that the DP condition is met, the concern of matching dataset with other data sets for revealing information is entirely ruled out. The DP guarantees protection against any auxiliary information. It should be noted that anonymization is an essential and primary step of protecting privacy. So, throughout this thesis, it is assumed that the dataset is anonymized i.e. all unique identifiable information such as names are removed.

Differential privacy is not an algorithm but a definition to quantify the privacy in an appropriate way. For some specific task with a given amount of differential privacy, there can exist unlimited number of algorithms. Offering privacy is not for free and there will be obviously some cost. The cost of adding privacy is to lose some amount of utility. To clarify, utility is equivalent to the performance or accuracy of the model in the context of deep learning. The key challenge in developing privacy protecting components is to settle a reasonable trade-off between privacy and utility. Maintaining such trade-off is not simple such that despite the essential need, almost no state-of-the-art machine algorithm utilizes privacy preserving elements due to unacceptable decline in their utility. Therefore, it is an open problem in machine learning to provide privacy at a reasonable cost of losing utility.

In the context of differential privacy, a *query* refers to a function that is applied to a data set. Also, a *mechanism* is an algorithm that takes a data set as input and produces a string as output.

*Adjacent* datasets are two datasets whose members are the same except for one member [27]. In other words, adjacent datasets are different in only one member.

## 2.1 Randomized Response

Randomization is the essential part of DP. Randomness refers to adding noise which can exist inherently or be induced manually. For instance, consider a dataset of postal codes in Canada to be shared publicly. The postal code in Canada is made of six characters including three digits and three alphabet letters. To provide the privacy in sharing a postal code, one way is to manipulate one character. In other words, the privacy is provided by introducing randomness. The strategy of *randomized response* is invented in the social sciences in collecting statistical data for studying illegal or embarrassing behaviors in society to protect the privacy of individuals in responding to such questions [27]. The technique asks the participant to respond to the question of whether they have the behavior B based on the following rule:

1. Flip a regular coin.

2. If head, then provide the true response.

3. If tail, then flip a second regular coin and provide Yes in case of tail and No otherwise.

Basically, the privacy comes from the possibility of denying the responses. In other words if somebody is assigned the behavior B, they can deny it even if in reality the allegation holds true. In this case, the privacy of the individual is protected. More precisely, the deniability or what we call *uncertainty* in the response to query brings privacy to the respondent. In the example of studying behavior B, there is always a probability of 1/4 (or 25%) for deniability. The average or the expected number of Yes responses is 75% of the number of individuals with behavior B plus 25% of the individuals without behavior B. Consequently, if the actual portion of behavior B individuals is assumed as $p$, then the number of Yes answers on average will be $(3/4)p + (1/4)(1\text{-}p)$ which is equal to $p/2 + (1/4)$.

$$Yes = p/2 + (1/4) \rightarrow$$

$$p = 2(Yes) + 1/2$$

$$= 2(p/2 + (1/4))$$

Now, the portion $p$ can be estimated as twice the number of individuals who responded Yes minus 1/2.

$$Yes = p + 1/2 \rightarrow$$

$$p = Yes - 1/2$$

So far, the materials are directed gradually to later reach to the formal definition of the DP. Before presenting the formal definition, some relevant materials are introduced.

**Definition 2.1.** [27] For a discrete set A, the *probability simplex* over A is represented and defined as:

$\Delta(A) = \{x \in R^{|A|}$ such that $x_i \geq 0$ and $\sum_{i=1}^{|A|} = 1\}$

As for notation, a dataset is represented here by its histogram as $x \in R^{|A|}$ where $x_i$ is the number of members of the dataset of type $i \in A$

**Definition 2.2.** [27] A randomized algorithm $\mathcal{A}$ with domain D and discrete range $\mathcal{R}$ is a mapping $M : \mathcal{A} \rightarrow \Delta(R)$.

**Definition 2.3.** [27] the $l_1$ norm of a dataset $x$ is represented and defined as:

$$\| x \|_1 = \sum_{i=1}^{|X|} |x_i| \tag{2.1}$$

Also, for two datasets $x$ and $y$, the $l_1$ distance between them is denoted by $\| x - y \|_1$.

On the notation, the $l_1$ norm of a dataset is an indicator of its size i.e. the number of the members of the dataset. Also, the $l_1$ distance of two datasets indicates the number of different members between them.

At this point, the preliminary required materials related to the differential privacy are provided such that the formal definition can be presented now.

**Definition 2.4.** [27] A randomized algorithm $\mathcal{A}$ is $(\epsilon, \delta)$ differentially private for any subset $\mathcal{S}$ of its range if for all $\mathcal{D}$ and $\mathcal{D}'$ adjacent datasets in its domain:

$$Pr[\mathcal{A}(\mathcal{D}) \in \mathcal{S}] \leq e^{\epsilon} Pr[\mathcal{A}(\mathcal{D}') \in \mathcal{S}] + \delta \tag{2.2}$$

As a rule of thumb, the $\delta$ which is referred to as the *probability of failure* is set to be in order of the inverse of any polynomial in the size of the dataset i.e. in the scale of $\| x \|_1$. In other words, the $\delta$ is the probability of accidental information leakage. As an example, for the MNIST [1] dataset whose size is 70k, the $\delta$ can be set as $10^{-5}$ which is conventional in experiments. As a special case and as in earlier definition of DP in literature, if $\delta = 0$, then the algorithm is called $\epsilon$ diffrentially private.

The $\epsilon$ is called the *privacy loss*, which is in fact the measure of privacy. Calculating $\epsilon$ is not always straight-forward. In practice, special cases and simplifications are considered to simplify calculating the privacy loss. Two of these special cases are Laplace and Gaussian mechanisms which will be presented later in this chapter.

## 2.2 Differential Privacy Properties

The differential privacy is currently the gold standard definition of the privacy in data analysis due to several interesting properties, of which the most important ones are described in this section.

### 2.2.1 Robustness Against Auxiliary Information

One of the very important properties and promises of DP is its robustness against auxiliary information. In the classical examples of information leakage in history, as mentioned earlier such as the Massachusetts governor or any other cases of linkage or matching databases, the adversaries use auxiliary information for re-identification. However, if a dataset or a model is differentially private, there will be no concern about information leakage as a result of existing auxiliary information.

### 2.2.2 Post-Processing

One of the interesting properties of DP is its robustness against post-processing, which is presented in the following proposition.

**Proposition 2.1.** [27] Assume that randomized algorithm $A : N^{|X|} \rightarrow R$ meets $(\epsilon, \delta)$ DP condition. Any randomized mapping mapping $M : R \rightarrow R'$ that is applied to the algorithm $M(A) : N^{|X|} \rightarrow R'$ is also $(\epsilon, \delta)$ differentially private.

Immunity of DP to post-processing implies that without knowledge about the dataset, no function can be designed on the output of a differentially private algorithm to make it less differentially private.

### 2.2.3 Composability

Another interesting property of DP is its composability which allows for designing a combination of multiple differentially private mechanisms. This will be presented later in this chapter with more details.

### 2.2.4 Group Privacy

Here, the $(\epsilon, 0)$ differentially private algorithms are considered. Essentially, the DP protects the privacy between adjacent datasets which differ in only one member. The idea can be extended to neighbor datasets which differ in more than one element. In the former case, it is referred to as *individual* privacy while in the latter it is known as *group* privacy. For the group, the differential privacy declines linearly proportional to the size of the group or equivalently the number of different members of neighbor datasets. This property states that if $(\epsilon, 0)$ differentially private algorithms are grouped, the ensemble collection is also differentially private but with a linear drop in privacy guarantee proportional to the size of the group.

**Theorem 2.1.** [27] For two neighbor datasets $D$ and $D'$ which differ in $k$ members or in other words $\| D - D' \|_1 \leq k$ for any subset $\mathcal{S}$ of the range of the algorithm A, the algorithm is $(\epsilon, 0)$ differentially private if it meets the following condition:

$$Pr[\mathcal{A}(\mathcal{D}) \in \mathcal{S}] \leq e^{(k\epsilon)} Pr[\mathcal{A}(\mathcal{D}') \in \mathcal{S}] \tag{2.3}$$

## 2.3 Practical Example

To show how differential privacy is calculated in practice, we show here that the example of the randomized response which was mentioned earlier is $(\ln 3, 0)$ differentially private.

*Proof.* $\delta$ here is 0. To calculate the privacy loss:

$$e^{\epsilon} = \frac{Pr(answer = Yes | truth = Yes)}{Pr(answer = Yes | truth = No)}$$

$$= \frac{3/4}{1/4} = 3 \rightarrow$$

$$\epsilon = \ln 3$$

## 2.4 Laplace Mechanism

As stated earlier, calculating the privacy loss in general is challenging. So, in practice, modifications and simplifications are made to facilitate the calculation. One of the cases is Laplace mechanism. Before providing the formal definition, some preliminary materials are provided.

**Definition 2.5.** [27] For a function $f$ with a domain $N^{|X|}$ and a range of $R^k$, the $l_1$ sensitivity is denoted and defined as following.

$$\Delta f = \max_{x,y \in N^{|X|}, \|x-y\|_1 = 1} \| f(x) - f(y) \|_1 \tag{2.4}$$

In fact, the $l_1$ sensitivity of a function indicates how far the function changes as a result of adding or removing one member of the dataset.

In the statistics and probability theory, the Laplace distribution is well known and is defined as following.

**Definition 2.6.** [27] The Laplace distribution with parameter $a$ is defined as follows.

$$Lap(x|a) = \frac{1}{2a} e^{-\frac{|x|}{a}} \tag{2.5}$$

The mean of the distribution is 0 and the variance is $2a^2$.

The Laplace distribution is depicted in Fig. 1 for three different standard deviations.

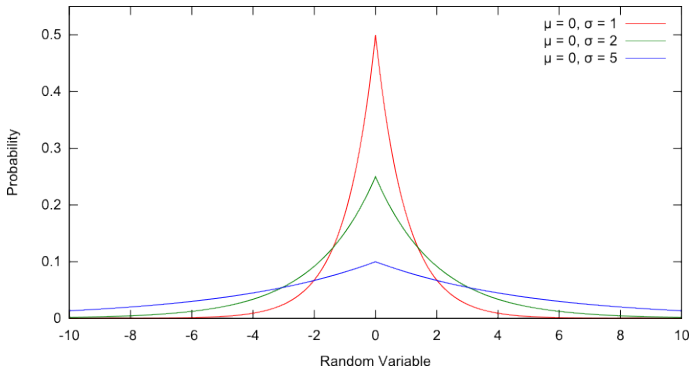


**Figure 2.1: Laplace distribution [9]**

As the figure shows, the diagram is symmetric, centered at origin and spreads over all real values of the horizontal axis. By increasing the standard deviation (or equivalently the variance in terms of effect), the curve spreads over more horizontal values and the value on the origin decreases. It is obvious that the area under the curve is the same for all different standard deviation values and is equal to 1.

At this point, the Laplace mechanism can be introduced. It is useful to recall that the DP works by introducing randomness or more precisely by introducing noise. For the Laplace mechanism, as the name implies, a noise from a Laplace distribution is added to each coordinate of the mechanism. Here is the formal definition.

**Definition 2.7.** [27] For a function $f$ from domain $N^{|X|}$ to range $R^k$, the Laplace mechanism is defined as $f(x) + (Y_1, Y_2, ..., Y_k)$ where $Y_i$ are independent identically distributed random variables coming from $Lap(\Delta f/\epsilon)$.

**Theorem 2.2.** [27] The Laplace mechanism is $(\epsilon, 0)$ differentially private.

So, to provide a privacy loss of $\epsilon$ to a mechanism, it is enough to add a noise with a Laplace distribution with parameter $\Delta f/\epsilon$ in every coordinate. It is useful to recall that $\Delta$ here is the $l_1$ norm. In the following, the proof to the theorem is provided.

*Proof.* Assume mechanism $M$ to be Laplacian with domain $N^{|X|}$ and range $R^k$. Also, consider two datasets $x$ and $y$ from the same domain $N^{|X|}$. The probability distributions of applying mechanism $M$ to datasets $x$ and $y$ are considered as $p_x$ and $p_y$ respectively at a given point like $z$. By assuming $\delta = 0$ we will have:

$$\begin{aligned}
\frac{p_x(z)}{p_y(z)} &= \prod_{i=1}^{k} \frac{\exp(-\frac{\epsilon|M(x)_i - z_i|}{\Delta M})}{\exp(-\frac{\epsilon|M(y)_i - z_i|}{\Delta M})} \\
&= \prod_{i=1}^{k} \exp(\frac{\epsilon}{\Delta M}(|M(y)_i - z_i| - |M(x)_i - z_i|)) \\
&\leq \prod_{i=1}^{k} \exp(\frac{\epsilon}{\Delta M}(|M(x)_i - M(y)_i|)) \\
&= \exp(\frac{\epsilon}{\Delta M} \parallel M(x) - M(y) \parallel_1) \\
&\leq \exp(\epsilon)
\end{aligned}$$

The first inequality is derived from the triangle inequality. The last inequality is based on the assumption that $\parallel x - y \parallel_1 \leq 1$. In a similar way and regarding the symmetry, it can be shown that

$$\frac{p_x(z)}{p_y(z)} \geq \exp(-\epsilon)$$

## 2.5 Composition Theorem

One of main subjects in context of differential privacy in machine learning and particularly in deep learning is the composition i.e. combining two or more independent differentially private mechanisms and calculating overall privacy of the combined mechanism.

**Theorem 2.3.** [27] Consider two mechanisms $M_1$ and $M_2$ from the same domain $N^{|X|}$ to ranges $R_1$ and $R_2$ receptively to be $(\epsilon_1, 0)$ and $(\epsilon_2, 0)$ differentially private. Now, define the combination of these two mechanisms as $M_{1,2} : N^{|X|} \rightarrow R_1 \times R_2$. The combined mechanism is $(\epsilon_1 + \epsilon_2, 0)$ differentially private.

*Proof.* First of all let $\delta = 0$. Assume that $x$ and $y$ belong to the domain of mechanisms. Also, $r_1$ and $r_2$ belong to the combined range space $R_1 \times R_2$.

$$
\begin{aligned}
&\frac{Pr[M_{1,2}(x) = (r_1, r_2)]}{Pr[M_{1,2}(y) = (r_1, r_2)]} = \frac{Pr[M_1(x) = r_1]Pr[M_2(x) = r_2]}{Pr[M_1(y) = r_1]Pr[M_2(y) = r_2]} \\
&= (\frac{Pr[M_1(x) = r_1]}{Pr[M_1(y) = r_1}](\frac{Pr[M_2(x) = r_2]}{Pr[M_2(y) = r_2}) \\
&\leq \exp(\epsilon_1)\exp(\epsilon_2) \\
&= \exp(\epsilon_1 + \epsilon_2)
\end{aligned}
$$

In the inequity, it is assumed that $\| x - y \|_1 \leq 1$. Similarly and given to the symmetry it is concluded that

$$
\frac{Pr[M_{1,2}(x) = (r_1, r_2)]}{Pr[M_{1,2}(y) = (r_1, r_2)]} \geq \exp(-(\epsilon_1 + \epsilon_2))
$$

The composition theorem that was just presented and defined for two mechanisms can be extended to any number more than two as well.

**Theorem 2.4.** [27] Assume that mechanisms $M_i : N^{|X|} \rightarrow R_i$ for $i = 1, 2, ..., k$ to meet $(\epsilon_i, \delta_i)$ DP condition. Now, the combination of these mechanisms is defined and denoted as $M : N^{|X|} \rightarrow \prod_{i=1}^{k} R_i$. Then, the combined mechanism is $(\sum_{i=1}^{k} \epsilon_i, \sum_{i=1}^{k} \delta_i)$ differentially private.

## 2.6 Gaussian Mechanism

As mentioned earlier, privacy calculation is not always straightforward. To tackle the complexity of calculation, some simplifications and special cases are considered in practice. Laplace mechanism as one of such cases was introduced in previous section. In this section, Gaussian mechanism as one of the very useful, natural, and practical cases is presented. While Laplace mechanism works on the $l_1$ sensitivity, Gaussian mechanism is linked to the $l_2$ sensitivity which is defined as following.

**Definition 2.8.** [27] For a function $f$ with a domain $N^{|X|}$ and a range of $R^k$, the $l_2$ sensitivity is denoted and defined as following.

$$\Delta_2 f = \max_{x,y \in N^{|X|}, \|x-y\|_1=1} \| f(x) - f(y) \|_2 \tag{2.6}$$

As it can be guessed, the Gaussian mechanism adds a zero-mean normal (or Gaussian) noise in every coordinate. The following theorem states the privacy measures regarding Gaussian mechanism.

**Theorem 2.5.** [27] The Gaussian mechanism with parameter $\sigma$ which is greater than $c\frac{\Delta_2(f)}{\epsilon}$ is $(\epsilon, \delta)$ differentially private, provided that $c^2 > 2\ln(1.25/\delta)$ for any $\epsilon$ between 0 and 1.

*Proof.* Assume that mechanism $f$ applies to dataset $D$ by adding a normal noise of $N(0, \sigma^2)$. By considering real value function, we will have

$$\Delta_2 f = \Delta_1 f = \Delta f$$

To calculate the privacy loss:

$$\ln \frac{\exp\left(-\frac{x^2}{2\sigma^2}\right)}{\exp\left(-\frac{(x+\Delta f)^2}{2\sigma^2}\right)}$$

We start by the privacy loss:

$$\ln \frac{\exp\left(-\frac{x^2}{2\sigma^2}\right)}{\exp\left(-\frac{(x+\Delta f)^2}{2\sigma^2}\right)} = \ln \exp -\left(\frac{x^2-(x+\Delta f)^2}{2\sigma^2}\right)$$
$$= -\frac{x_2-(x+\Delta f)^2}{2\sigma^2}$$
$$= \frac{2x\Delta f+(\Delta f)^2}{2\sigma^2}$$

Now, we force the privacy loss to be less than $\epsilon$. Then, we will get:

$$\frac{2x\Delta f+(\Delta f)^2}{2\sigma^2} < \epsilon$$

$$\rightarrow x < \sigma^2\epsilon/\Delta f - \Delta f/2$$

To enforce this bound with a probability at least $1-\delta$:

$$Pr[|x| \geq \sigma^2\epsilon/\Delta f - \Delta f/2] < \delta$$

As $x$ comes from a Gaussian distribution, we will have:

$$Pr[x \geq \sigma^2\epsilon/\Delta f - \Delta f/2] < \delta/2$$

The tail bound of the Gaussian is used here to calculate the last inequality.

$$Pr[x > t] < \frac{\sigma}{\sqrt{2\pi}} e^{-t^2/2\sigma^2}$$

So,

$$\frac{\sigma}{\sqrt{2\pi}} e^{-t^2/2\sigma^2} < \delta/2 \rightarrow$$

$$\ln(t/\sigma) + t^2/2\sigma^2 > \ln(2/\sqrt{2\pi}\delta) = \ln\left(\sqrt{\frac{2}{\pi}}\frac{1}{\delta}\right)$$

Now, we replace $t = \sigma^2\epsilon/\Delta f - \Delta f/2$ in the first term. We also take $\sigma = c\Delta f/\epsilon$. At this point, in order to bound the parameter $c$, we take into account the non-negativity condition of $t$ as it is the argument of ln. Thus,

$$\frac{1}{\sigma}(\sigma^2\frac{\epsilon}{\Delta f} - \frac{\Delta f}{2}) = c - \frac{\epsilon}{2c}$$

We assume that $\epsilon \leq 1$ and $c \geq 1$. Consequently, $c - \frac{\epsilon}{2c} \geq c - 1/2$. Therefore, to enforce the condition $t \geq 0$, it is led to $c \geq 3/2$

Now, we consider the second term.

$$(\frac{1}{2\sigma^2}\frac{\sigma^2\epsilon}{\Delta f} - \frac{\Delta f}{2})^2 = \frac{1}{2}(c^2 - \epsilon + \epsilon^2/4c^2)$$

This sentence is monotonically increasing as its derivative with respect to $c$ is positive. Regarding $c \geq 3/2$ and $\epsilon \leq 1$, we will have

$$c^2 - \epsilon + \epsilon^2/4c^2 \geq c^2 - 8/9$$

So, we get to

$$c^2 - 8/9 > 2\ln\left(\sqrt{\frac{2}{\pi}}\frac{1}{\delta}\right)$$

We continue

$$c^2 > 8/9 + 2\ln\left(\sqrt{\frac{2}{\pi}}\frac{1}{\delta}\right) = \ln e^{8/9} 2\ln\left(\sqrt{\frac{2}{\pi}}\frac{1}{\delta}\right)$$

Given that $(2/\pi)e^{8/9} < 1.55$, we can conclude that $c^2 > 2\ln(1.25/\delta)$.

## 2.7 Renyi Differential Privacy

As a generalization and relaxation to the traditional definition of differential privacy which particularly provides more convenient composition calculation, Renyi differential privacy (RDP) is introduced based on Renyi divergence.

**Definition 2.9.** [38] A randomized algorithm $\mathcal{A}$ is $(\alpha, \epsilon)$ RDP with $\alpha \geq 1$, if for any $\mathcal{D}$ and $\mathcal{D}'$ neighboring datasets in its domain, the Rényi divergence satisfies:

$$\frac{1}{\alpha - 1} \log E_{\delta \sim \mathcal{A}(\mathcal{D}')}[(\frac{\mathcal{A}(\mathcal{D})}{\mathcal{A}(\mathcal{D}')})^{\alpha}] \leq \epsilon \tag{2.7}$$

**Lemma 2.1.** [38] For two randomized mechanisms $M_1$ and $M_2$ which are $(\alpha, \epsilon_1)$ and $(\alpha, \epsilon_2)$ Renyi differential private, composing these two mechanisms will be $(\alpha, \epsilon_1 + \epsilon_2)$ Renyi differential private.

It can be shown that in limit when $\alpha \to \infty$, the Renyi divergence converges to privacy loss $\frac{Pr[\mathcal{A}(\mathcal{D})]}{Pr[\mathcal{A}(\mathcal{D}')]}$ i.e. the traditional definition of differential privacy. In fact, the privacy guarantee of RDP lies between $(\epsilon, 0)$ and $(\epsilon, \delta)$. The following proposition presents the relation between RDP and DP.

**Proposition 2.2.** [38] A mechanism that is $(\alpha, \epsilon)$ RDP is also $(\epsilon + \frac{\log 1/\delta}{\alpha - 1}, \delta)$ differentially private for arbitrary value of $\delta$ between 0 and 1.

To make a better comparison between RDP and DP, the RDP can be written as follows [2].

$$Pr[A(D)] \leq (e^{\epsilon} Pr[A(D')])^{1-1/\alpha}$$

Now, it is easier to see that by appraoching the $\alpha$ to infinity, the RDP converges to the $\epsilon$-DP (without term $\delta$ in the definition).

In fact, RDP provides a privacy guarantee in between $\epsilon$-DP and $(\epsilon, \delta)$-DP.

# Chapter 3

# Differential Privacy in Deep Learning

## 3.1 Introduction

As stated earlier, privacy can be discussed in context of the data or the model. The focus of the present research is the privacy on the model. By model, it is meant machine learning models, and more precisely deep learning models.

Essentially, deep learning refers to an artificial neural network (ANN) with many hidden layers. Basically, an ANN is composed of an input layer, an output layer and one or several hidden layers. Within each layer there are neurons which may also be called nodes or units. In each layer (obviously except for the output layer), the outputs of individual neurons are weighted and summed. Then, the weighted sum passes through an activation function which is typically nonlinear even though it can be linear as well. A typical ANN is shown in Fig. 3.1.

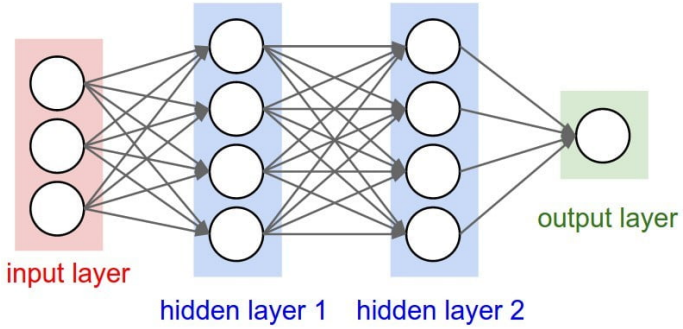


**Figure 3.1: A typical artificial neural network including input, output and hidden layers (two hidden layers in this example) [11]**

Is any simple ANN with several hidden layers named as deep model? Theoretically yes, but technically no. Even though Multi-Layer Perceptron (MLP) i.e. simple ANN with multiple hidden layers is the opening gate towards inventing deep models, but it is not necessarily considered as a deep model. Instead, a deep model can only have one single hidden layer but still be considered as deep model. The criterion for a deep model is the number of parameters i.e. weights and biases. The number of parameters of a deep model is in the order of thousands at least. Now, it can be understood how an ANN with just a single hidden layer can be considered as a deep model. It only needs to encompass big-enough number of parameters in no matter how many hidden layers. As an example of deep learning models, Alex Net [35], as one of the first and popular architectures in deep learning that was introduced in 2012 is represented in Fig. 3.2 For an input of size $224 \times 224 \times 3$, the number of parameters of Alex Net will be 62.3 million.

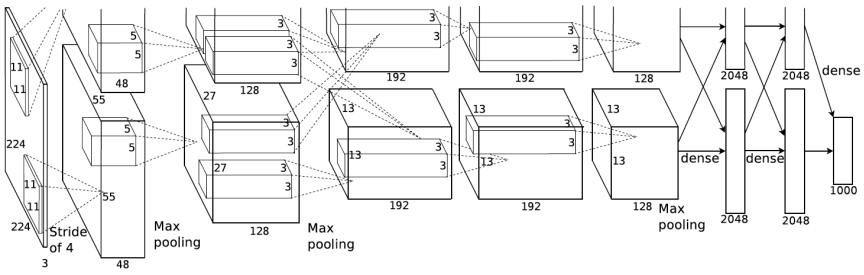


**Figure 3.2: An example of deep learning model: ALex Net [7]**

## 3.2 Training Neural Networks

The training procedure in deep learning and in ANN in general is performed as following. The training is essentially an iterative procedure. In practice, it is not possible (or at least recommended) to devise a direct formula to return the optimal values of parameters (weights and biases). What happens in practice is that at the very beginning, the parameters are chosen arbitrarily i.e. randomly or otherwise. Then, a *batch* of training data is chosen. Ideally, the entire data should be taken as a batch. However, in practice, the computational cost of passing the whole training dataset together in a single trial is exhaustive. So, a (relatively small) portion of training set is (randomly) picked every time. The batch or more formally *minibatch* is passed to the network and a *forward* step through the network is taken and the output of the system is calculated. If the calculated outputs are equal (or sufficiently close) to actual outputs, then the work is done and the current

parameters are the final and optimal parameters. If not, the *distance* between the estimated and actual outputs is calculated. Defining the distance function itself is a controversial step and an open problem. Presently, for regression tasks the euclidean distance and for classifications tasks the cross-entropy are conventionally used. Once the distance or formally *loss* function is calculated, a *backward* step through the network is taken and parameters are *adjusted* or in formal words *updated* according to the distance between estimated and actual outputs. Actually, the parameters are adjusted according to the *gradient* of the loss function which is the derivative of the loss function with respect to parameters. The conventional optimization solution in convex problems is to take steps in the opposite direction of the gradient. This is called *gradient descent* which is the algorithm of finding the minimum point in convex problems. The parameter updating rule is as following.

$$\omega \leftarrow \omega - \eta g$$

where $\omega$ is the parameter, $g$ is the gradient, and $\eta$ is the learning rate which controls the size of the steps to be taken.

So far, one trial or formally *iteration* of training ANN is performed. Another batch is chosen randomly and the procedure is repeated and the parameters are updated. The iterations continue until the loss function ideally becomes zero or practically becomes sufficiently small.

## 3.3 Need for Big Datasets in Deep Learning

By increasing the number of parameters of an ANN, the number of calculations increases as well. Each parameter is essentially an *unknown* to be found through solving an equation system. Theoretically, to find $n$ unknowns, $n$ equations are required. Therefore, by growing the number of parameters (or equivalently the number of unknowns), more equations and consequently more processing will be required which demands more powerful processors.

By emerging powerful hardware processors in computers e.g. graphical processing unit (GPU) and tensor processing unit (TPU), it became possible to increase the number of parameters greatly. Apart from the need for powerful processors in deep models, there emerges another need for big datasets to train the networks on. Again, theoretically, for solving $n$ equations corresponding to $n$ unknowns, $n$ examples are required. Thus, if the number of parameters in an ANN increases, so

will the number of required samples for training the network. That is, deep learning models need big datasets for training.

For real life and commercial applications, the deep learning models need to train on real life datasets i.e. the actual data of the people in the world. Such datasets sometimes contain personal private or confidential information which should not be divulged to the public. The model is trained on datasets with potentially confidential information. After being trained, the dataset is detached from the model forever. Then the model is shared publicly. So, the public will have access to the model. So far, no problem is observed. However, in reality, there exists a severe issue as will be seen in the following section.

## 3.4 The Problem: Unintended Memorization in Deep Learning

It is shown that deep learning models are naturally susceptible to memorizing the input data and, as a result, are prone to give in the private data as the response of curious and possibly malignant queries [23]. The issue stems from the fact that ANNs contain no privacy-protecting component in their architecture. So, it happened and can happen that ANNs return the confidential information on which they are trained and were not supposed to be divulged as a response to inquiries by malevolent experts.

The memorization incident in deep learning models is different from overtraining [50]. It might be helpful to recall that overtraining or overfitting in neural networks refers to the case where the performance on the training set is high while on the validation (or test) set is poor. So the memorization cannot be resolved by applying the overfitting handling techniques such as regularization, weight decay, and dropout. As the data size grows, the vulnerability of the model augments as well. Carlini et al [24] demonstrated this issue by introducing attack on GPT-2, a language model, released by OpenAI, which is trained on giant public Internet data. They could successfully reveal hundreds of training data including information such as the names, phone numbers, physical address, and email addresses of people. It is useful to recall the example presented earlier, the writing recommendation application on cell phones which is actually a language model that is trained on the data of all users who use the application. The training data can contain confidential and sensitive information. To enlighten the severity of the problem, imagine how it would be if somebody can

reverse the training path and extract the training data of particularly confidential information, e.g., credit card number, social insurance numbers, and secret industrial strategies.

Machine learning models and particularly deep learning models are designed to learn the distribution of the data. The concept of *learning* is completely different from *memorizing*. As a simple example to better understand the difference between them, consider a student who is given some questions and answers to learn. To evaluate and to see if she learned well, the student takes a test. If in the test, the student answers perfectly the questions she already saw, it doesn't necessarily prove that he/she learned well. Answering correctly to the questions that previously presented is a necessary condition, but not sufficient as it might come from simply memorizing the answers. It is sufficient only if the student answers correctly the *generalized* questions i.e. the questions within the same category of the training set, but not the same ones. A smart student is not supposed to *memorize* the examples which are presented, but to learn the hidden underlying structures and patterns or in other words the *distribution* of the examples. This is exactly the case in machine learning. A desirable machine learning model is supposed to learn the distribution of the training data, and not to simply memorize them.

Memorization or more precisely *unintended* memorization in deep learning models is different from overtraining. The two concepts are close but subtly different. More precisely, overtraining is a sufficient condition for memorization, but not necessary. In other words, it happens to have a deep model with no overtraining but with memorization [24]. So, the conventional approaches to handle the overtraining such as regularization and dropout are not helpful in overcoming the memorization issue in deep learning. Hence, it necessitates to devise privacy providing structures in deep learning models.

## 3.5 The Solution: Differentially-Private Stochastic Gradient Descent (DPSGD) Algorithm

To address the issue of the lack of a privacy protecting component in deep learning models, in 2016 the algorithm of Differentially-Private Stochastic Gradient Descent (DPSGD) [13] is proposed. It is useful to recall that differential privacy is based on inserting randomness or in other words noise to the mechanism. In the DPSGD algorithm, a Gaussian noise is added to the gradient of the loss

function. As mentioned earlier, in a Gaussian mechanism, it is assumed that for adjacent datasets, the $l_2$ sensitivity of the mechanism should be limited. To enforce this condition, a *clipping gradient* step is undertaken. In fact, if the $l_2$ norm of the gradient is greater than a predefined constant $C$, the gradient is clipped to have a $l_2$ norm of $C$. Otherwise, the gradient is preserved. Mathematically:

$$g_t \leftarrow g_t / max(1, \frac{\parallel g_t \parallel_2}{C})$$

where $g_t$ is the gradient and $C$ is the predefined constant which is usually assumed as 1. If the gradient's $l_2$ norm is less than the constant $C$, then it will remain as it is. But if the $l_2$ norm exceeds the threshold $C$, it will be clipped to $\frac{C}{\|g_t\|_2} g_t$ whose $l_2$ norm is equal to $C$. Briefly:

$$g_t = \begin{cases} \frac{g}{\|g_t\|_2} C & \text{if } \parallel g_t \parallel_2 > C \\ g_t & \text{if } \parallel g_t \parallel_2 \leq C \end{cases}$$

In the following, the DPSGD algorithm is presented.

**Algorithm 1:** The DPSGD algorithm

**Input:** parameters $\theta$, learning rate $\eta$, number of microbatches $\mu$, noise standard deviation $\sigma$

initialize $\theta$ randomly;

**for** $t \in \{T\}$ **do**

  $B_t \leftarrow$ take a batch from dataset randomly

  $\nabla_\theta \leftarrow 0$

  **for** *microbatch* $b \in B_t$ **do**

    $\nabla_\theta^\mu \leftarrow$ gradient of microbatch $b$

    $\nabla_\theta^\mu \leftarrow \nabla_\theta^\mu / max(1, \frac{\|\nabla_\theta^\mu\|_2}{C})$

    $\nabla_\theta \leftarrow \nabla_\theta + \nabla_\theta^\mu$

  **end**

  $\nabla_\theta \leftarrow \frac{1}{\mu}(\nabla_\theta + \mathcal{N}(0, \sigma^2 I))$

  $\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta$

**end**

**Output:** $\theta_T$ and calculate the total privacy cost $(\delta, \epsilon)$

## 3.6 Issue of the DPSGD Algorithm: Poor Trade-Off Between Privacy and Utility

The DPSGD algorithm offers privacy to deep learning model at the cost of losing some utility. Altering the gradient by performing the clipping and also adding noise to gradient values will have a damaging effect on the performance of the system. As an instance, by offering a privacy loss ($\epsilon$) of 1 to a classification task on the MNIST dataset, the accuracy will be around 93% [13] which may be acceptable in a theoretical ground but not in real life and commercial applications. So, there is a need towards improving the lost performance of the model while offering a sufficient amount of privacy.

The biggest issue of today in context of privacy in deep learning is the balance between privacy and utility. So far, the algorithms that offer DP in deep learning models decline the privacy to an intolerable level. Therefore, to date, no state-of-the-art machine learning algorithm uses privacy protecting components due to this unacceptable decline in the performance of the system despite the vital need. To better illustrate the problem, consider the example of antivirus in computers. It is obvious that computers are at risk of malware and virus attacks so that it is essential to use an antivirus component. However, sometimes installing an antivirus so much lowers the speed (i.e. utility or performance) of the system that users prefer not to use it despite the critical risk of virus/malware attacks. Likewise for privacy protecting components.

## 3.7 Privacy Calculation in Deep Learning

This thesis is based on modification of the DPSGD algorithm, which in turn is based on the Gaussian mechanism which has been presented earlier. In this section, some details are provided on privacy calculations in the context. As mentioned earlier, DP is offered by adding noise which is in fact the randomness that is essential in the context of privacy in general and DP in particular. The more the noise, the more the privacy. As a matter of fact, for a given amount of privacy (i.e. $\epsilon$), the required amount of noise (i.e. the variance or standard deviation) is calculated based on the theorem 2.5.

Theorem 2.5 can be applied to any Gaussian mechanism. In the DPSGD algorithm, by adding normal (Gaussian) noise to the gradient, we will have a Gaussian mechanism whose privacy can be calculated based on the theorem 2.5.

DP calculation in the DPSGD algorithm has two steps: **offline** and **online**. The privacy metrics ($\epsilon$ and $\delta$) are calculated offline while the utility metric is measured online . In fact, for the DPSGD algorithm, for a given pair of $(\epsilon, \delta)$, the amount of noise to be added is calculated according to the theorem 2.5 in an **offline** manner i.e. running no experiments. The deep learning model is turned off so far. Then, the model is switched on and the training starts to observe and calculate the utility (i.e. performance or accuracy) of the differentially private model in an **online** fashion.

### 3.7.1 A Numerical Example

As a numerical example, in the DPSGD algorithm, assume that we want to have privacy amount of $(\epsilon = 0.5, \delta = 10^{-5})$ in a classification task on the MNIST data, we will calculate how much noise ($\sigma$) should be added.

The theorem states that $\sigma$ should be greater than $c\frac{\Delta_2(f)}{\epsilon}$ while meeting the condition $c^2 > 2\ln(1.25/\delta)$. In this example,

$$c^2 > 2\ln(1.25/\delta) = 2\ln(1.25/10^{-5}) = 23.47 \rightarrow$$

$$c^2 > 23.47 \rightarrow$$

$$c > 4.84$$

So, any values greater than 4.84 are acceptable for $c$. But we are interested in the smallest ones to keep the required noise as small as possible in order to keep the deteriorating effect of the noise to the performance of the system as low as possible. Let's choose $c = 4.84$ and remember that in the DPSGD algorithm the $l2$ norm is set to be 1.

$$\sigma > c\frac{\Delta_2(f)}{\epsilon} = 4.84 \times \frac{1}{0.5} = 9.66$$

So, to provide a DP of $(\epsilon = 0.5, \delta = 10^{-5})$, it is required to add a Gaussian noise of $\sigma = 9.66$. This is the privacy calculation which is offline i.e. without running the deep network. Now, to observe and calculate the performance of the system under the $(\epsilon = 0.5, \delta = 10^{-5})$ DP condition, the network is needed to run. So, the network is switched on and the classification task starts on the MNIST dataset and the accuracy is measured to see how it is affected under the imposed DP condition.

# Chapter 4

# Literature Review

In this chapter, the objective is to review the important and representative research works in developing privacy-protecting algorithms with more focus on machine learning models. It is aimed to present the related research studies in a chronological way to show the progress, evolution, and maturation of approaches on the ground by illustrating the strong as well as weak points associated with each work. This will be helpful in identifying existing research gaps and illuminating the way through which the future studies should be conducted.

## 4.1 Developing Theoretical and Quantitative Grounds on Privacy

Mainly, the privacy is a qualitative and abstract notion. However, when it comes to analysis and modelling, it becomes essential to settle quantitative grounds. There are several quantitative definitions for privacy among which the most popular ones are k-anonymity, l-diversity, t-closeness [36], and differential privacy (DP) [26, 27]. The DP is a well-suited definition of privacy that so far reflects the best the general abstract notion of the privacy in a quantitative manner.

The idea of differential privacy as a quantitative measure of privacy in a mathematical or in better words statistical form, primarily was introduced by Cynthia Dwork [26, 27]. The concept is comprehensively covered in a book which is regarded as a principal reference for differential privacy.

## 4.2 Privacy Attacks

The research in the area was originally motivated after several privacy attacks happened to publicly-accessible data and resulted in re-identification of the people whose information was released publicly and had been promised to be protected. So, it is worthwhile to review popular examples of privacy attacks in order to learn the lessons from and also to get inspiration for developing defensive and protective approaches.

In the 1990s, in Massachusetts, the state group insurance commission decided to release the hospital admission report of the state's personnel. Even though the database was made anonymized, a computer scientist could re-identify many people including the governor of Massachusetts by matching the database with vote registration records [40]. What can be learned from this example is that anoymization is necessary but never sufficient.

Netflix collected a database of viewing history of their subscribers. The database was anonymized and shared publicly as a training set for a recommendation competition about viewing history. However, the Netflix subscribers were re-identified as a result of matching the database with the Internet Movie Database (IMDb) [27]. This experience implies that anonymization fails to prevent information leakage in case of availability of auxiliary information. Re-identification in these two examples happened by *linkage* attacks to anonymized databases.

One kind of primary attacks on privacy is called *membership inference attack* through which adversaries are able to determine whether a given sample belongs to the training data [37, 48, 41, 23].

As stated earlier, memorization in deep learning models is different from overtraining (overfitting) so that the conventional overtraining techniques like regulation, weight decay, and dropout will not help with privacy protection. Carlini et al [23] evaluated and tested unintended memorization in neural networks in general and in deep models in particular. They also introduced a new metric *exposure* to measure the unintended memorization in neural networks which contributes to distinguish memorization from overtraining. They considered generative models and the special case of *language* models in which the unintended memorization can have drastic consequences. They proposed as future works to address the issue of such memorization to other types of neural networks such as image classifiers. Carlini et al in a recent work [24] demonstrated again the issue by introducing attack on GPT-2, a language model, released by OpenAI, the variant of Transformer

[22, 45, 44], which is trained on giant public Internet data. The authors group consists of twelve researchers with well-known affiliations. They could successfully reveal hundreds of training data including information such as the names, phone numbers, physical address, and email addresses of people. By this work, they undermined the dominant assumption that information leakage stems from overfitting [22, 55]. The traditional intuition supports this claim by pointing out training of modern deep models over a few epochs and consequently their inclination towards overfitting [46].

## 4.3 Developing Privacy-Protecting Approaches

Abadi et al [13] proposed the differential privacy stochastic gradient descent (DPSGD) algorithm which induces privacy into deep learning models. The algorithm delivers privacy at the cost of losing some accuracy. The intuition was to clip individual gradients in order to diminish the extreme effect of some individual inputs that might otherwise lead to unintended memorization of the system. It is also further supported by adding noise to the sum of the individual gradients. The approach leverages scaling down the gradients whose second-order norm is greater than some predefined threshold while keeping other gradients unchanged. The privacy is provided in this way but at the cost of losing some performance. The lost performance raises controversy over the effectiveness of this approach.

Nasr et al [42] introduced a modified version of the original DPSGD algorithm based on encoding gradient and denoising which restored some lost performance. However, the approach still fails to resolve the issues existing in the original work, plus working at an exorbitant computational cost.

A relatively new emerging area where privacy of model becomes urgent is the federated learning where data inputs come from various users and then are integrated within a federated model. Several studies addressed privacy concerns in federated learning [19, 49, 34, 18, 56, 41].

As emphasized throughout this work, one important area for privacy discussion is health data. To highlight even more the significance of the area, consider the current situation of the COVID19 global pandemic. Most likely, many databases will be released publicly for future research and development purposes. Protecting the identity of the people whose medical records are shared is of utmost importance. Such protections demand developing rigorous privacy protecting approaches while preserving the utility of the data.

One solution to address the privacy issue when publicly sharing the medical records is to generate synthetic data and replace the original data with those data. In other words, the objective is to generate synthetic health data which are privacy protected and similar enough to the original data, and then share them publicly. In the following, the recent advancements in generating privacy-preserved synthetic health data are reviewed.

One of the first developments in generating synthetic health data is MedGAN [17] which generates high-dimensional inputs including binary and count variables. It leverages a combination of generative adversarial network (GAN) and autoencoder to generate realistic synthetic medical data. There is no additional privacy providing component other than the GAN itself in the model. The intuition comes from the fact that given to a training set, essentially GAN generates new-brand samples. In other words, the data samples generated by GAN are similar to the training data but not a copy. So, they justified that the data generated by GAN can be considered as privacy protected data. However, now we know that neural networks and GAN in particular can memorize and consequently divulge the training data. As a result, MedGAN suffers from identity as well as attribute disclosure.

As a new perspective to the problem, it is aimed to introduce privacy protection units somehow into the network. Empirical results show that GAN can easily remember and memorize the training data which stems from the high complexity inherent in deep networks. Consequently, GAN in its plain shape is susceptible to divulge the confidential health records. The proposed network is called DPGAN [54] which, as the name implies, adds the DP into the GAN.The proposed network is devised in the context of WGAN which is a variant of GAN based on optimizing the Wasserstein distance. The normal (Gaussian) noise is added to the gradients of the loss function. The privacy providing step is further enforced by a clipping operation on the final parameters (weights) which allegedly bounds the gradient of each training data. To date, the state-of-the-art is a model named PATE-GAN [32] which introduces the privacy in a different way. It tailors the private aggregation of teacher ensembles (PATE) to GAN. The privacy preserving element is introduced through the discriminator of the GAN. The proposed algorithm is evaluated on three different settings: 1- train and test on real data, 2- train and test on synthetic data, and 3- train on synthetic while test on real data. Compared with GAN and DPGAN, the proposed structure PATE-GAN achieved better results in terms of AUROC (area under the receiver operating characteristics curve) and AUPRC (area under the precision recall curve) criteria.

Torfi et al [52] proposed correlation capturing Generative Adversarial Network (CorGAN). They combined Convolutional Generative Adversarial Networks and Convolutional Autoencoders in order to capture the correlation among the training data. They applied their model to two medical datasets MIMIC-III and UCI Epileptic Seizure Recognition. They also compared their model with Stacked Deep Boltzmann Machines (DBMs), Variational Autoencoder (VAE), and MedGAN. They evaluated the quality of the synthetic data based on the *area under rectified operating characteristic* (AUROC) and *area under precision recall curve* AUPRC metrics. Their results outperformed the other models in their study. However, the models does not contain any addition privacy preserving element. In a later work, they introduced privacy preserving elements into generative models. They proposed a model that employs Renyi Differential Privacy and Convolutional Generative Adversarial Networks which is called RDP-CGAN [53] . They compared their model with other models MedGAN, TableGAN [43], DPGAN, and PATE-GAN. The models are applied to six medical datasets MIMIC-III, UCI Epileptic Seizure Recognition, Kaggle Cervical Cancer, PTB Diagnostic, Kaggle Cardiovascular Disease, and MIT-BIHArrhythmia [12, 28, 39, 29, 20, 33, 16]. They evaluated the models according to AUROC and AUPRC measures. The overall performance showed to be higher for their proposed model against other examined models in the study. In another work, they introduced a new measure to evaluate the quality of the generated synthetic data [51]. They call the metric as Siamese Similarity Score (SSS) or Siamese Distance Score (SDS) and showed that it behaves similar to F-1 score. The metric is inspired by the siamese structure or contrastive loss that tries to minimize the intra-distance while maximizing the inter-distance for a given set of data [21]. In other words, siamese or contrastive loss assigns a big value if two elements belong to two different classes while assigning a small value if two samples belong to the same class. They concluded that their metric better reflects the quality of the generated images in comparison with metrics Fréchet Inception Distance (FID) score [31] and Inception score [47].

The works reviewed in this chapter demonstrate various efforts in developing privacy-preserving structures. Some of them show relatively promising results but not entirely satisfactory yet. Thus, more studies are demanded to be conducted to offer a satisfying level of effectiveness especially in real-world and commercial applications where a perfect or nearly-perfect performance is required. The present research aims to contribute to fill this existing gap in constructing privacy-providing elements with an acceptable trade-off between privacy and utility in deep learning models.

# Chapter 5

# Contribution

In this chapter, the contribution of the thesis is presented. The core in this work is to provide privacy and in particular differential privacy to deep learning models. The severe threats to a privacy-protectionless machine learning model have been elaborated in previous chapters. It has also stated that machine learning models and particularly deep learning models naturally contain no privacy-protecting components to prevent information leakage in response to curious and especially malevolent queries. Today, there exist numerous deep learning models for federated learning or daily usages such as next-word recommendation applications on cell phones available for public users which are trained on personal data of many users. Unless strict privacy measures are taken, such models are at risk of memorizing and divulging sensitive information as a response to users' queries. So, it is necessary to equip publicly-shared deep learning models with privacy-preserving components. Several approaches are proposed to address the issue by developing a variety of privacy-protecting structures. As a novel and original contribution, the differentially private stochastic gradient descent (DPSGD) algorithm is proposed in 2016 [13] which works as following. In every epoch of training, the algorithm performs a clipping operation on the gradients to restrict the $l_2$ norm of the gradient to ensure staying less than unity. Further, Gaussian noise as essential randomness for differential privacy is added to the clipped gradients. The differential privacy is achieved in this way at an exchange of utility of the model. However, the paid cost here is hardly acceptable in real-world and especially commercial applications. As a result, the state-of-the-art machine algorithms are reluctant in hiring such privacy-providing structures.

The main challenge in the context of privacy in machine learning is to establish a reasonable trade-off between privacy and utility i.e. to offer privacy at the least cost of utility. Despite many contributions to date, it still remains as a challenging open problem.

## 5.1 Tangent Hyperbolic: Privacy and Beyond

In this research, the idea is to offer privacy to deep learning models at a less decline in their performance. It is proposed here to replace the *clipping gradient* part with a tangent hyperbolic function. By passing through the tanh filter, the inputs are smoothed no matter how big they were originally. The tanh filter trims and balances out the inputs by spreading them out between the range of $+1$ and $-1$. At the same time, the superiority order between values is preserved which means that an originally greater value stays still greater after passing through the filter. The tangent hyperbolic function is plotted in Fig. 5.1.

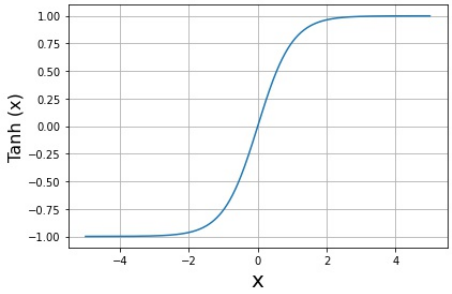


**Figure 5.1: Tangent hyperbolic function**

So, the tanh can be considered as an appropriate candidate for privacy protection in deep learning models. On one side, the tanh restricts and suppresses the extreme input values which would leave big traces on the memory of the system that could further lead to unintended memorization of those values. On the other side, the relative greatness among input values is preserved so that less damage to the utility can be expected. Preserving the sign (positivity or negativity) of the input is a key feature of tanh that prevents degradation of the performance.

## 5.2 Modified DPSGD Algorithm

The modified DPSGD algorithm is proposed and described as following. Just as for clarification on terminology used here, the number of microbatches should be distinguished from the microbatch size. The former refers to the number of microbatches within a single batch while the latter denotes the number of samples inside a microbatch.

The algorithm is initialized with (usually) random initial parameters (weights and biases) which are denoted as $\theta$. One hyperparameter that should be defined, among others, at the beginning is the activation range $k$ which determines the horizontal extension of the tanh function which will be introduced later in this chapter. The dataset is divided into a number of mini-batches. At each trial (whose total number is denoted as $T$), one batch is randomly selected and then is divided into smaller units named as micro-batch. Each micro-batch is passed to the network and then the gradient is calculated. To provide privacy, the gradient is passed through the tangent hyperbolic filter whose role is to balance out and suppress the extreme values of individual gradients while preserving their superiority accordingly. All filtered gradients corresponding to micro-batches within a mini-batch are summed, to which the noise is added then to enforce the essential randomness which is required to provide the DP. Next step is to update parameters and take a step in the opposite direction of the gradient. The process is recursively continued for all mini-batches.

**Algorithm 2:** Modified DPSGD algorithm

**Input:** parameters $\theta$, learning rate $\eta$, number of microbatches $\mu$, noise standard deviation $\sigma$, the activation range $k$

initialize $\theta$ randomly;

**for** $t \in \{T\}$ **do**

  $B_t \leftarrow$ take a batch from dataset randomly

  $\nabla_\theta \leftarrow 0$

  **for** *microbatch* $b \in B_t$ **do**

    $\nabla_\theta^\mu \leftarrow$ gradient of microbatch $b$

    $\nabla_\theta^\mu \leftarrow \tanh \nabla_\theta^\mu / k$

    $\nabla_\theta \leftarrow \nabla_\theta + \nabla_\theta^\mu$

  **end**

  $\nabla_\theta \leftarrow \frac{1}{\mu}(\nabla_\theta + \mathcal{N}(0, \sigma^2 I))$

  $\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta$

**end**

**Output:** $\theta_T$ and calculate the total privacy cost $(\delta, \epsilon)$

The tanh function is frequently used in deep learning models as the activation function. However, it has never been used as a privacy protecting element.

The properties of tanh function in smoothing the input values can be useful in other parts of machine learning as well. One application of tanh is just presented as privacy protecting element. Another application can be offered in the recurrent neural network (RNN) to address the issue of gradient explosion. In fact, one solution to fix the gradient exploding problem was the gradient clipping. Thus, it can be replaced by the tanh filter.

### 5.2.1 The Issue of Tangent Hyperbolic

By observing more carefully the tanh in Fig. 5.1, an almost fast regime of saturation can be recognized within the function. In other words, the outputs tend to converge to 1 (or -1) for inputs greater than let's say 3 (or -3). It means that for all inputs outside this range (between -3 and +3) that we call *activation range* (AR), the output is almost the same. More precisely, the values outside AR are indistinguishable by the tanh filter. For instance, all values like 10, 11, 16, 29, 47, 635, and 719306658 are transformed to almost the same values. As will be seen in the results section, this issue will appear as instabilities in the performance of the model.

### 5.2.2 Solution to the Issue

As a natural solution to the issue of relatively small AR, it can be proposed to expand the interval. Experiments in this study show that enlarging the AR results in stabilizing the performance of the model. On the other hand, as the experiments will show, changes (increasing or decreasing) along the vertical axis contribute to no considerable changes in the performance. The horizontal enlargement factor of the tanh is denoted as $k$ in the Algorithm 2.

## 5.3 Privacy Discussion

Theoretically, the $l_2$-sensitivity 1 condition of DPSGD does not hold when applying the TH filter to gradients. The $l_2$-norm a vector $G$ of size $n$ with entries $g_i$ is defined as following.

$$\| G \|_2 = \sqrt{\sum_{i=1}^{n} g_i^2} \tag{5.1}$$

After passing the gradient element-wise through the TH filter, we will have:

$$|g_i| < 1 \rightarrow g_i^2 < 1 \rightarrow \sum_{i=1}^{n} g_i^2 < n \rightarrow \sqrt{\sum_{i=1}^{n} g_i^2} < \sqrt{n}$$

$$\rightarrow \| G \|_2 < \sqrt{n}$$

This shows that by passing the gradient through the TH function, the $l_2$-sensitivity of the gradient can be as much as $\sqrt{n}$ where $n$ is the number of elements of the gradient. As stated earlier, the $l_2$-sensitivity determines how much noise should be added to a Gaussian mechanism for a given amount of privacy. Obviously, as the $l_2$-sensitivity grows, so does the amount of noise to be added. In other words, mathematically, for a given privacy amount ($\epsilon$), the approach in this work requires more noise than DPSGD. Consequently, more noise will degrade the performance of the system. However, in practice, the reality is different. In fact, the gradient rapidly converges to values close to zero as the training progresses by approaching to the minimum point (local optimum). So, in practice, shortly after starting the training, most gradient elements are placed around and close to zero. It might be useful to refer to Nasr's work [42] where they plotted the gradient histogram. In other words, as the training proceeds, the actual norm of gradient will be much lower than the loose bound of $\sqrt{n}$. Hence, in practice, for the proposed approach, the assumption of unity for the $l_2$-sensitivity of the gradient is not far from reality. At least, it can be stated that the gradient is bounded. This fact ensures offering a reasonable amount of privacy while improving the performance at the same time which is a promising step in privacy-preserving deep learning area and makes the

proposed approach more suitable for real-world and especially commercial applications where the high utility is fundamental.

## 5.4 Contribution to RNN

Recurrent neural network (RNN) is a family of deep learning models that takes into account continuity in the data. In other words, it is well suited to time series data in which the data at a given point is related to the previous time points. Such dependence between data points can be seen in language data or more specifically in natural language processing (NLP) as any word in a phrase is related to neighbor (previous or after) words. A typical schema of RNN is shown in Fig. 5.3 in which connections between different units are represented by arrows. In the figure, the blue units represent input units, while yellow and gray units represent hidden and output units respectively.

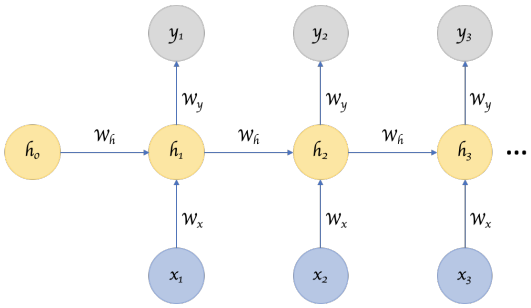


**Figure 5.2: A typical schema of recurrent neural network (RNN)**

RNN is designed to perform properly in processing time series or language data. However, this family of deep models suffers from an issue emerged as *gradient exploding/vanishing* which refers to the case where limit values of gradient (too small or too big) lead to instability in backpropagation and updating parameters. More specifically, the gradient exploding happens when the gradient contains extremely big entries. This issue can cause serious problems and eventually paralyze the network by stopping the training process. As an ultimate solution, the long short term memory (LSTM) network was later proposed to solve the problem and replaced RNN. One solution to the exploding gradient problem was *gradient clipping* i.e. cutting the gradient when it is greater than a predefined threshold. However, it raised then utility concerns as it compromises the performance of the model.

Now that the idea of replacing the clipping gradient by a tanh filter performs relatively well in privacy preserving context, it can be extended to RNN to fix the gradient exploding issue. The tanh filter will smooth and balance out the extreme values of the gradient while preserving their superiority order to prevent the performance from wildly degrading.

# Chapter 6

# Results and Discussion

In this chapter, the experimental results collected in the project are presented. At first, the datasets that are used in the experiments are introduced. Afterwards, the technical implementation information is described. Then, the simulation results are presented. A discussion and interpretation on the results as well as suggestions for future works are given at the end of the chapter.

In this study, two standard datasets, MNIST and CIFAR-10 [10], are used that are prevalently recognized as benchmark datasets for different machine learning tasks. MNIST [1] is a collection of 70000 handwritten digits (0-9) in 10 classes. The training and test sets contain 60000 and 10000 images respectively. The images are available in black and white with the size of $28 \times 28$. Some examples of the MNIST dataset are demonstrated in Fig. 6.1.

**Figure 6.1: Examples of the MNIST dataset [3]**

CIFAR-10 [10] is a set of 60000 color images of size $32 \times 32$ in 10 classes with 6000 images per class. The size of the training and test sets is 50000 and 10000 respectively. Samples of CIFAR-10 are depicted in Fig. 6.2.

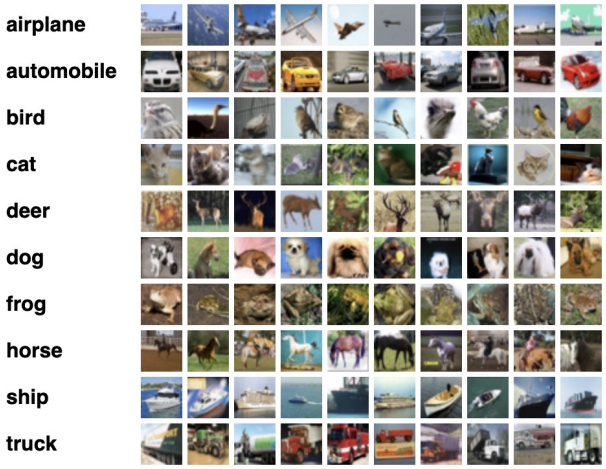


**Figure 6.2: Samples of CIFAR-10 dataset [8]**

## 6.1 Privacy Measurement

In all experiments, for the proposed as well as all other algorithms, the privacy measures are calculated based on Rényi Differential Privacy (RDP). The privacy budget for the proposed approach is calculated in the same way as for the original DPSGD algorithm. In other words, the same amount of noise is considered for DPSGD and the proposed approach. It should be noted that theoretically, the DPSGD assumption of $l_2$-sensitivity 1 is not held in the proposed approach. Theoretically, $l2$-sensitivity of a TH filtered gradient is $\sqrt{n}$ where $n$ is the number of the gradient elements. However, in practice, the gradient values are placed quite close to zero such that the bound of $\sqrt{n}$ is too loose for $l_2$-sensitivity. It will be useful to refer to [42] for further details and evidences. In this work, the motivation is to improve the performance of the model in a way that makes it suitable for real-world and commercial applications at a reasonable and acceptable compromise in privacy.

The experiments are implemented with TensorFlow [14] on shared GPUs of Google Colaboratory [5] online platform. The Tensorflow Privacy library [4] was used to implement algorithms and calculate privacy and utility metrics.

A summary of hyperparameter values is listed in Table 1. These are default values in all experiments in this work unless mentioned otherwise.

**Tableau 6.1: Hyperparameters list**

| Hyperparameter | Value |
|---|---|
| failure probability $\delta$ | $10^{-5}$ |
| noise covariance $\sigma$ | 1.1 |
| sampling rate for MNIST | 0.427 % |
| sampling rate for CIFAR-10 | 0.512 % |
| minibatch size | 256 |
| microbatch size | 1 |

The performance of the proposed algorithm as well as the original one on MNIST and CIFAR-10 datasets is shown in Figs. 6.3 and 6.4 respectively. The effect of translating inputs into bigger output ranges is also investigated and is plotted for some values of $c$ by which the regular range (-1, 1) is multiplied. As can be seen in these figures, no significant effect on accuracy associated with the range of output can be observed which is not far from expectation.

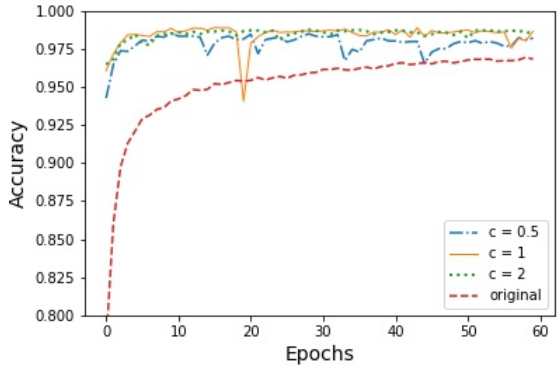


**Figure 6.3: Performance of model with different output ranges of filter compared with original algorithm over training steps for the MNIST dataset**

As the figures show so far, the proposed algorithm offers an improvement in performance compared with the original DPSGD, at the cost of some instability though. Furthermore, a saturation regime in performance results over training epochs is observed such that improving accuracy over more training time seems less likely. As can be seen in Fig. 5.1, in tangent hyperbolic (TH) function, the outputs almost saturate for the inputs greater than, let's say, 3 (we are not looking for the exact value). In fact, what we call the *active range* (AR) is small in TH. All inputs outside the AR have almost the same output. So, a natural solution is to enlarge the AR and encompass more inputs.

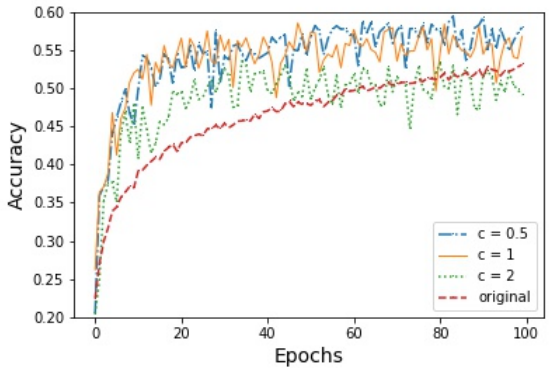


**Figure 6.4: Performance of model with different output ranges of filter compared with original algorithm over training steps for CIFAR-10 dataset**

The results of expanding the AR by a factor of $k$ are presented in Figs. 6.5 and 6.6. As these figures imply, expanding the AR contributes to not only stabilizing the results but also retaining the ascending trend which suggests that by taking further training steps, better performance is expected to achieve. The other observation is that bigger expanding coefficients, on one hand, lead to more stable results, and on the other hand lower the accuracy. So, a new hyperparameter is added here to choose the appropriate expanding coefficient.

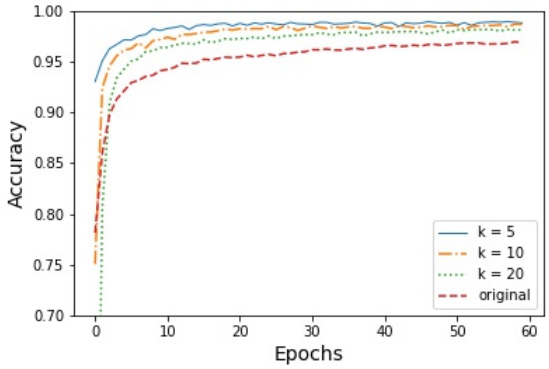


**Figure 6.5: Stabilizing model performance by using different expanding coefficients for MNIST dataset**

## 6.2 The Effect of Changing the Variance of Noise

As an ablation study, the effect of changing the variance (or standard variation) of Gaussian noise which is added to the gradient is investigated here. The results are presented in Figs. 6.7 and 6.8.

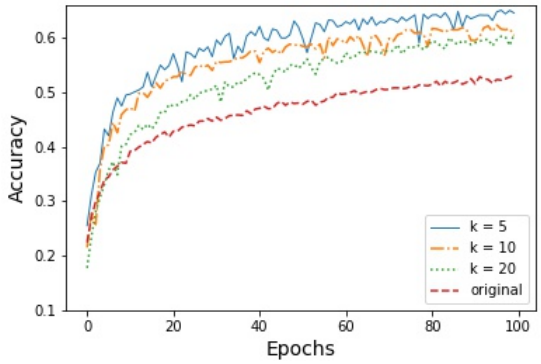


**Figure 6.6: Stabilizing model performance by using different expanding coefficients for CIFAR-10 dataset**

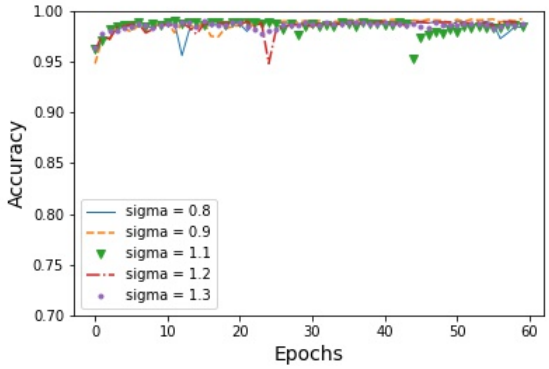


**Figure 6.7: Effect of amount of noise on performance of the model for MNIST dataset**

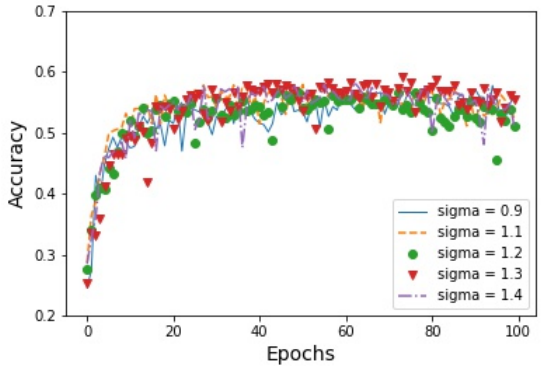


**Figure 6.8: Effect of amount of noise on performance of the model for CIFAR-10 dataset**

As can be observed from these figures, changing the variance (or standard variation) of Gaussian noise within a range that is not wide contributes to no remarkable effect on the performance of the system. Logically, increasing the noise is expected to degrade the utility.

## 6.3 The Effect of Changing the Learning Rate

As an important part of the ablation study as well as hyperparameter tuning, the effect of different learning rates on the performance of the deep learning model is examined here. The results for MNIST and CIFAR-10 datasets are presented in Figs. 6.9 and 6.10.

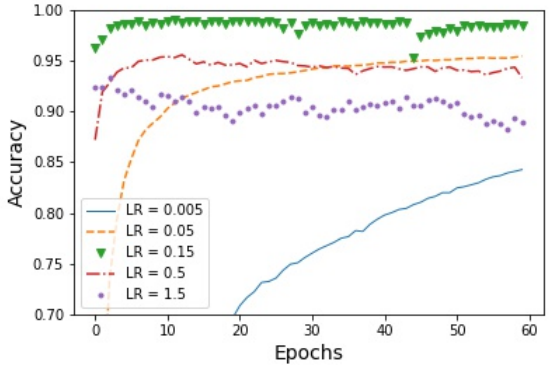


**Figure 6.9: Effect of the learning rate on performance of the model for MNIST dataset**

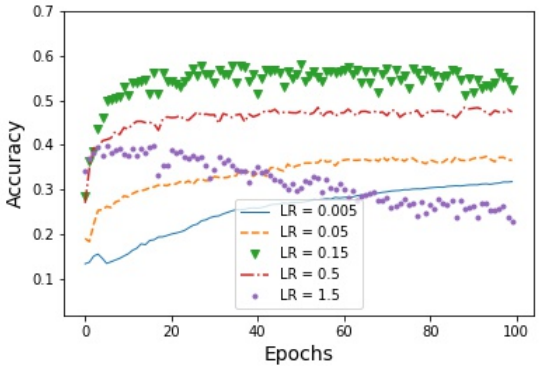


**Figure 6.10: Effect of the learning rate on performance of the model for CIFAR-10 dataset**

As can be clearly observed from the figures, choosing a proper learning rate plays a significant role on the performance of the model. In general, too small learning rate slows down the training while too big value might cause the model to jump over the (sub)optimal point and results in divergence of the model. Throughout this thesis, the $LR = 0.15$ is chosen for the experiments as it is evidently proved here to produce the best results.

## 6.4 Results of tanh and Clipping Together

The effect of applying tanh filter while keeping the clipping gradient step is also investigated. The tanh is applied first to trim and balance out the gradient values followed by the gradient clipping to enforce the unity condition for the $l_2$ norm and ensuring DP guarantee. The motivation to conduct this experiment was to investigate whether enforcing the unity condition for the $l_2$ norm to ensure DP guarantees contribute to improving the results. The results are compared with original DPSGD which contains only the clipping gradient, and presented for MNIST and CIFAR-10 datasets in Figs. 6.11 and 6.12 respectively.

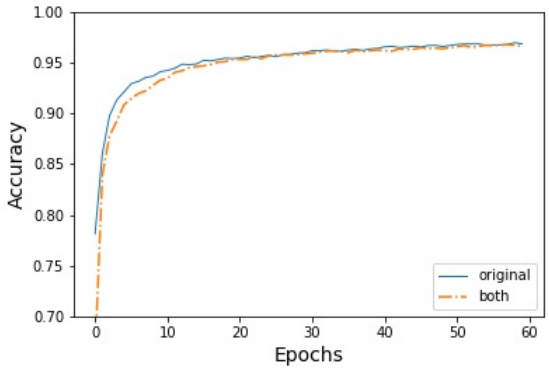


**Figure 6.11: Comparison of double operators with original one for MNIST dataset**

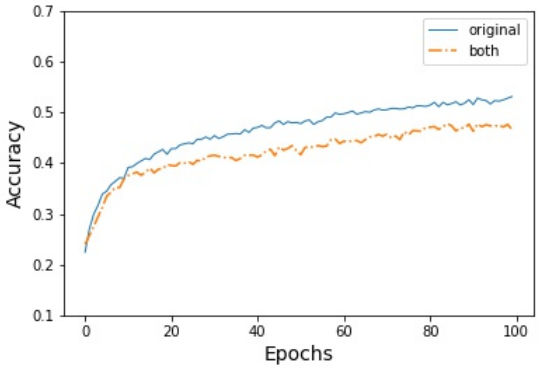


**Figure 6.12: Comparison of double operators with original one for CIFAR-10 dataset**

As can be seen in these two figures, combining the tanh with the clipping gradient makes no big difference in the performance of the model and contributes to no improvement. It can be inferred that degrading the performance in DPSGD model is associated with the clipping gradient step. So, as long as the clipping gradient is in place, utility decline remains.

## 6.5 Discussion

The main keyword in this thesis is *privacy*. In this study, privacy refers to the unintended leakage of information as a response to inquiries. The privacy is discussed when *data* or *model* is shared publicly and more precisely on the internet. In this project, privacy on the **model** is studied. What is meant here in this study is deep learning models which are basically artificial neural networks. The shared deep learning models are usually trained on the personal data that might contain sensitive and confidential contents which are supposed not to be divulged to any third party. It has been shown that the training path can be reversed in shared models and consequently the training data can become accessible.

By growing technologies and by emerging big datasets gathered from real people potentially containing confidential information, the problem becomes more critical. Nowadays, deep learning models are found in real life and routine applications. In smart phones, such models help users in writing texts by recommending next words. Such recommendation applications are actually deep learning models that are trained on the data of all users who use the application. In other words, these deep learning models are trained on text messages, emails, and search flows of people. It is obvious within such training database that there is sensitive and confidential personal information. What is publicly accessible is the model not the data. However, there are examples that the training data -that were supposed to remain out of public access- are disclosed as a result of curious inquiries. All disclosing examples are common in that they all lack a privacy protection component in the architecture of deep learning model. Even though, there are privacy preserving elements for deep learning models, the offered privacy is too expensive or it degrades the utility of the system to an intolerable extent. Now, so we are motivated by these necessities to conduct the present work towards reconciliation between privacy and utility.

In general, privacy is an abstract notion so that it is not measured and described numerically. However, when it comes to analysis and modelling especially in an engineering or computer science context, it is inevitable to define quantitative grounds. The differential privacy is appeared as a fairly effective definition of privacy. In this project, a modification to the DPSGD algorithm is proposed. The proposition is based on replacing the gradient clipping step by a tanh filtering which smooths out the wild values of the gradient that could otherwise leave a memorable trace on the memory of the system which would lead to unintended memorization and eventually information

leakage. It trims and balances out while preserving the relative order of input values. So, the tanh filter contributes to privacy of the model and at the same time maintains the utility of the model. The experimental results support this claim. Despite justifications for efficacy of the proposition in a practical point of view the theoretical grounds lack strong supports.

The main contribution of this work is offering a practical privacy protecting element for deep learning models while maintaining the utility of the model. In real life and particularly in a commercial context, the utility is very important. Such applications demand a perfect or close to perfect utility. That's why to date, almost no state-of-the-art deep learning model employs privacy preserving blocks in spite of serious necessity as the existing approaches do not meet an acceptable trade-off between privacy and utility.

Regarding the natural properties of the tanh filter and its leverage in organizing gradients that relaxes the need to gradient clipping operation, it can be extended to RNN to encounter the gradient exploding problem.

## 6.6 Suggestions for Future Works

Machine learning and in particular deep learning techniques are based on experience. The effectiveness of most of the inventions and contributions are empirically proven. Even though lack of strong theoretical grounds is sometimes considered as a shortcoming in machine learning, the importance is effectiveness in practice.

As a future work and to prove effectiveness of the proposed approach it is suggested to design and conduct attacks to deep learning models that are equipped with the privacy preserving component that is proposed in this study. If such attacks fail to reverse the training path and to disclose the training data, then the efficacy of the approach will be proven and then it can be used in real world and especially commercial applications.

Even though the proposed approach is applicable to any deep learning model, one useful application can be as generating privacy-preserving artificial health data. Publicly sharing health records is a controversial area, and preserving privacy of the people whose data is shared is challenging. One solution to the issue can be generating privacy-preserved synthetic data whose performance is

close to original data, and then share them publicly instead of the original data. The proposed approach in this study can be included as a component to generative deep models to generate artificial medical data.

# Chapter 7

# La confidentialité différentielle dans les modèles d'apprentissage profond

## 7.1 Introduction

En général, le concept de la confidentialité peut être confondu avec la sécurité vu qu'un chevauchement existe bien entre ces deux concepts. Dans un contexte scientifique et plus précisément dans le domaine de l'informatique, ces deux sont entièrement distincts et ont significations différentes. En effet, la sécurité se réfère au cas où deux parties communiquent par un canal de communication puisque aucune tierce partie ne peut accéder au message communiqué. Dans ce cas, le canal est sécurisé. Néanmoins, dans le contexte de la confidentialité, il n'y a pas de tierce partie. Cependant, lorsqu'une partie met en cause une deuxième, si l'interrogateur obtient plus d'information qu'il devait, c'est dit que la confidentialité du répondant est brisée. Donc, le sujet principal dans le contexte de la confidentialité est d'empêcher la fuite nondésirable d'informations.

La confidentialité peut être discutée dans le contexte des données ainsi que des modèles. Quand une base de données personnelles est publiquement partagée, il faut garantir que l'identité des personnes dont les données sont partagées demeure cachée. L'anonymisation (enlever les noms et les autres identifiants uniques) est une étape primordiale et nécessaire mais pas suffisante pour préserver la confidentialité. Il existe beaucoup de situations de la réidentification des données anonymisées en faisant correspondre ces données à d'autres bases de données. Dans les années 90s,

à Massachusetts, le gouvernement a décidé de publier le rapport d'hospitalisation de leurs employés. Ils ont anonymisé la base de données et l'ont publié. Une informaticiene a fait correspondre cette base de données avec une autre et par la suite a réussi à réidentifier plein de personnes y compris le gouverneur de l'état. Donc, il faut concevoir d'autres moyens que l'anonymisation pour préserver la confidentialité.

La confidentialité peut être discuté dans le contexte des modèles. Il y a des modèles publiquement partagés qui sont entraînés sur des données personnelles avec des contenus potentiellement confidentiels comme les numéros des cartes de crédit, les numéros d'assurance sociale, et les stratégies industrielles. Les données d'entraînement servent à apprendre la distribution des données par le modèle. Il n'est pas supposé de mémoriser les données d'entraînement mais d'apprendre leur distribution. Il se trouve que les modèles d'apprentissage profond (les réseaux de neurones artificiels) peuvent mémoriser des données d'entraînement et les divulguer. Par exemple, sur les téléphones cellulaires intélligents, il existe une application de récommendation du mot suivant pour écrire des textos. Ces modèles sont déjà entraînés sur les données personnelles des utilisateurs (textos, courriels, etc) ainsi d'autres usagers qui utilisent la même application. Imaginez ce qui arriverait si un utilisateur tape "Le numéro de carte de crédit de la personne A est", et ensuite l'application recommendera le bon numéro. Une telle fuite d'information peut se produire avec les modèles d'apprentissage profond du fait du manque de composants pour protéger la confidentialité dans l'architecture de ces modèles. Donc, c'est essentiel de construire des éléments protégeants la confidentialité dans les modèles d'apprentissage profond.

## 7.2 La définition technique de la confidentialité

En général, la confidentialité est une notion abstraite et qualitative. Alors, dans un contexte analytique et scientifique, il faut bâtir des fondations mathématiques et quantitatives. Présentement, la définition la plus universelle acceptée pour la confidentialité est "la confidentialité différentielle" proposée par Cynthia Dwork [26]. Voici la définition.

**Définition 7.1.** [27] Étant donné les bases de données D et D' qui ne sont différentes que d'un seul membre, un algorithme aléatoire $\mathcal{A}$ satisfait la confidentialité différentielle de $(\epsilon, \delta)$ si

$$Pr[\mathcal{A}(\mathcal{D}) \in \mathcal{S}] \leq e^{\epsilon} Pr[\mathcal{A}(\mathcal{D}') \in \mathcal{S}] + \delta \tag{7.1}$$

où $\epsilon$ est la perte (ou le coût) de la confidentialité et $\delta$ est la probabilité de la fuite.

### 7.2.1 Les caractéristiques de la confidentialité différentielle

Il y a des caractéristiques intéressantes pour la confidentialité différentielle dont les plus importantes suivront ici.

**Robustesse contre les informations auxiliaires:** C'est à dire que tant que la condition de la confidentialité différentielle soit en vigeur, aucune information auxiliaire peut aider à la réidentification.

**Immunité contre le post-traitement:** C'est à dire que la confidentialité différentielle demeure en vigure en cas de n'importe quelle fonction aléatoire effectuée sur l'algorithme.

**Confidentialité de groupe:** La définition de la confidentialité différentielle concerne les bases de données qui ne sont différentes qu'un seul membre. Alors, la définition peut être généralisée pour le cas où les bases de données sont différentes par plus qu'un seul membre. Dans ce cas, la perte de la confidentialité sera proportionnelle au nombre des membres différents.

**Le théorème de la composition:** La confidentialité différentielle rend facile l'intégration de plusieurs algorithmes qui sont déjà différentiellement confidentiels. En effet, si deux mécanismes $(\epsilon 1, 0)$ et $(\epsilon 2, 0)$ différentiellement confidentiels s'intégrent, la combinaison sera $(\epsilon 1 + \epsilon 2, 0)$ différentiellement confidentiel.

## 7.3 Le mécanisme gaussien

En général, le calcul de la perte de la confidentialité n'est pas facile. Donc, afin de simplifier des calculs, on considère des simplifications et des cas spéciaux. Un important cas spécial est le mécanisme gaussien. Comme le nom bien indique, il s'agit d'ajouter de bruit normal (ou Gaussien) au mécanisme.

**Théorème 7.1.** Un mécanism gaussien avec le paramètre $\sigma$ supérieur que $c\frac{\Delta_2(f)}{\epsilon}$ est $(\epsilon, \delta)$ différentiellement confidentiel si $c^2 > 2\ln(1.25/\delta)$ pour n'importe quel $\epsilon$ entre 0 et 1.

## 7.4 La confidentialité différentielle de Rényi

La première définition de la confidentialité différentielle n'inclut que $\epsilon$ (en excluant $\sigma$) et s'appelle la $\epsilon$ confidentialité différentielle qui est une définition assez stricte. En tant qu'une généralisation et rélaxation naturelle, la confidentialité différentielle de Rényi a été proposée qui est basé sur la divergence de Rényi.

**Définition 7.2.** Un algorithme aléatoire $\mathcal{A}$ est $(\alpha, \epsilon)$ différentiellement confidentiel de Rényi avec $\alpha \geq 1$ si pour les bases de données voisines $\mathcal{D}$ et $\mathcal{D}'$ :

$$\frac{1}{\alpha - 1} \log E_{\delta \sim \mathcal{A}(\mathcal{D}')}[(\frac{\mathcal{A}(\mathcal{D})}{\mathcal{A}(\mathcal{D}')})^{\alpha}] \leq \epsilon \tag{7.2}$$

Le théorème suivant exprime la rélation entre la confidentialité différentielle et la confidentialité différentielle de Rényi.

**Théorème 7.2.** Un algorithme qui est différentiellement confidential de Rényi est aussi $(\epsilon + \frac{\log 1/\delta}{\alpha - 1}, \delta)$ différentiellement confidential pour n'importe quelle valeur de $\delta$ entre 0 et 1.

## 7.5 La confidentialité différentielle dans les modèles d'apprentissage profond

Dans le fond, les modèles d'apprentissage profond ne comprennent pas des composants à protéger la confidentialité dans leur architecture. Par la suite, ils peuvent mémoriser des données d'entraînement qui n'est pas favorable dans le sens de la confidentialité. Il faut rappeler que le concepte de la mémorisation est différente que le surentraînement. Par conséquent, les techniques comme régularisation sont inutiles. Donc, il faut construire des éléments qui offrent la confidentialité dans les modèles profonds.

En effet, la confidentialité est fournie par l'ajout du bruit ou en général par la randomisation. En ajoutant le bruit (gaussien ou Laplace entre autres), on obtient la confidentialité mais en même temps on perd l'utilité. Donc, il y a toujours un compromis entre l'utilité et la confidentialité.

En 2016, un groupe chez Google a proposé une approche qui s'appelle l'algorithme de descente de gradient stochastique différentiellement confidentiel (DGSDC)[13] pour produire la confidentialité dans les modèles profonds. Dans l'algorithme, on coupe le gradient s'il est supérieur à une valeur prédéfinie. Le gradient est considéré comme le représentant des données d'entrée du système. Si un gradient est trop grand, il pourra laisser une trace plus grande sur la mémoire du système qui mènera à la mémorisation des données. La procédure est suivi par l'ajout d'un bruit gaussien aux gradients coupés. L'algorithme est présenté ci-dessous.

**Algorithm 3:** L'algorithme DGSDC

**Input:** les paramètres $\theta$, le taux d'apprentissage $\eta$, le nombre de micro-lots $\mu$, l'écart type du bruit $\sigma$

initialiser $\theta$ aléatoirement;

**for** $t \in \{T\}$ **do**

  $B_t \leftarrow$ prendre aléatoirement un lot des données

  $\nabla_\theta \leftarrow 0$

  **for** *micro-lot* $b \in B_t$ **do**

    $\nabla_\theta^\mu \leftarrow$ gradient du micro-lot $b$

    $\nabla_\theta^\mu \leftarrow \nabla_\theta^\mu / max(1, \frac{\|\nabla_\theta^\mu\|_2}{C})$

    $\nabla_\theta \leftarrow \nabla_\theta + \nabla_\theta^\mu$

  **end**

  $\nabla_\theta \leftarrow \frac{1}{\mu}(\nabla_\theta + \mathcal{N}(0, \sigma^2 I))$

  $\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta$

**end**

**Output:** $\theta_T$ et calculer le coût total de la confidentialité $(\delta, \epsilon)$

De cette manière, la confidentialité est produite mais avec un déclin dans la pérformance du système qui compte comme un aspect négatif. La dégradation de l'utilité n'est pas tolérable dans les usages de vrai monde et surtout commerciaux.

## 7.6 La revue de littérature

Il existe une recherche assez large sur le sujet de produire la confidentialité dans les modèles d'apprentissage profond. Dans ce projet, la littérature est divisée dans un nombre de catégories: le développement des bases théoriques et quantitatives pour la confidentialité, la conception des attaques aux modèles non-protégés, et le développement des algorithmes offrants la confidentialité.

Le travail sur les fondements théoriques est principalement conduit par Cynthia Dwork [26, 27] dont la définition de la confidentialité différentielle est universellement utilisée.

Dans l'histoire, il y a des exemples très connus de la fuite d'information à cause de partager les bases de données non-sécurisées sans composants de protéger la confidentialité. On peut mentionner l'exemple du gouverneur de l'état de Massachusetts aux états-unis [40] ou celui de Netflix [27] qui peuvent se servir à souligner l'importance du sujet de la confidentialité. Il y a aussi des exemples des attaques conçues par des scientifiques aux modèles d'apprentissage profond pour souligner leur vulnérabilité contre la mémorisation et la fuite des informations. Les recherches conduites par Carlini [23, 24] se démeurent dans cette catégorie.

En réponse du manque de composants de la confidentialité dans les modèles profonds, les différents algorithmes sont proposés dont le plus remarquable est DGSDC [13] proposé par un groupe chez Google en 2016. Il y a aussi les modifications à l'algorithme original de DGSDC comme celui par Nasr [42].

Un domaine où la confidentialité joue un rôle important est dans le domaine des données médicales. Une approche possible est de générer des données artificielles qui sont confidentiellement protégées et les partager publiquement au lieu des données originales. Il existe plusieurs algorithmes qui traitent ce sujet dans [54, 17, 43, 52, 53, 51, 32].

La revue de la littérature souligne le manque de la confidentialité dans les modèles d'apprentissage profond. Il y a des étapes très importantes qui ont été déjà effectuées mais le travail n'est pas encore finie. Les modèles déjà existant ne sont pas suffisamment efficaces pour être utiliser dans les applications réeles et surtout commerciales.

## 7.7 La méthode proposée

Dans le présent projet, une modification à l'algorithme DGSDC est proposée visant à l'amélioration des modèles d'apprentissage profond en offrant une confidentialité acceptable. L'idée est de remplacer le découpage de gradient par la fonction du tangent hyperbolique. En effet, cette fonction lisse et supprime les variations extrêmes des entrées pour les étaler entre -1 and +1 en respectant les valeurs relatives et aussi le signe des entrées. La fonction de Tanh est affiché dans la figure 7.1.

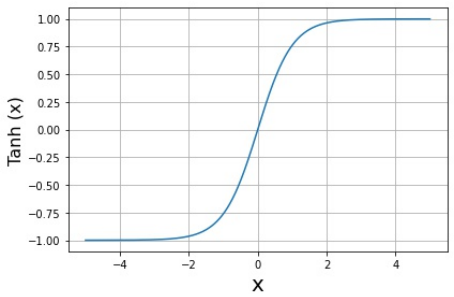


**Figure 7.1: La fonction de tangent hyperbolique**

L'algorithme modifié de DGSDC est présenté ci-dessous.

**Algorithm 4:** L'algorithme modifié de DGSDC

**Input:** les paramètres $\theta$, le taux d'apprentissage $\eta$, le nombre de micro-lots $\mu$, l'écart type du bruit $\sigma$, la gamme de l'activation $k$

initialiser $\theta$ aléatoirement;

**for** $t \in \{T\}$ **do**

  $B_t \leftarrow$ prendre aléatoirement un lot des données

  $\nabla_\theta \leftarrow 0$

  **for** *micro-lot* $b \in B_t$ **do**

    $\nabla_\theta^\mu \leftarrow$ gradient du micro-lot $b$

    $\nabla_\theta^\mu \leftarrow \tanh \nabla_\theta^\mu / k$

    $\nabla_\theta \leftarrow \nabla_\theta + \nabla_\theta^\mu$

  **end**

  $\nabla_\theta \leftarrow \frac{1}{\mu}(\nabla_\theta + \mathcal{N}(0, \sigma^2 I))$

  $\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta$

**end**

**Output:** $\theta_T$ et calculer le coût total de la confidentialité $(\delta, \epsilon)$

Comme les résultats des expérimentations démontrent, même le Tanh dans sa forme originale ne mène pas à des résultats intéressants. La raison est que la gamme de l'activation du Tanh est essez courte. C'est à dire qu'il existe un régime de la saturation assez rapide. Donc, une solution naturelle peut être l'élargissement de la gamme en sorte d'inclure plus d'entrées. Le paramètre $k$ dans l'algorithme 4 est utilisé pour cette raison.

Le sujet de l'utilisation de Tanh pour supprimer les variations extrêmes en respectant la supériorité peut être utile dans les autres domaines que la confidentialité. Comme une autre utilisation, on peut l'utiliser pour régler le problème de l'explosion de gradient qui arrive dans les réseaux de neurones récurrents (RNR).

## 7.8 Résultats

Afin de valider la méthode proposée, deux bases de données standardes sont utilisées: MNIST et CIFAR-10. MNIST [1] est un ensemble des chiffres manuscrites (0-9 en total 10 classes) puisque CIFAR-10 [10] comprend des images des petits objets de 10 catégories. Les expérimentations sont mise à œuvre en Python. La programmation a été réalisée sur Google Colaboratory [5], la plateforme gratuite fournie par Google, en utilisant les processeurs graphiques. La bibliothèque originale de Tensorflow Privacy [4] a été modifiée afin d'appliquer l'algorithme proposé. Une liste des hyperparamètres utilisés dans ce projet est présenté ci-dessous.

**Tableau 7.1: La liste des hyperparamètres**

| **hyperparamètre** | **Valuer** |
|---|---|
| probabilité d'échec $\delta$ | $10^{-5}$ |
| écart type du bruit $\sigma$ | 1.1 |
| taux d'échantillonnage pour MNIST | 0.427 % |
| taux d'échantillonnage pour CIFAR-10 | 0.512 % |
| taille de mini-lot | 256 |
| taille de micro-lot | 1 |

Les résultats de la mise en oeuvre l'algorithme proposé ainsi que celui de l'original sont présentés dans les figure 7.2 and 7.3 pour les deux bases de données MNIST et CIFAR-10.

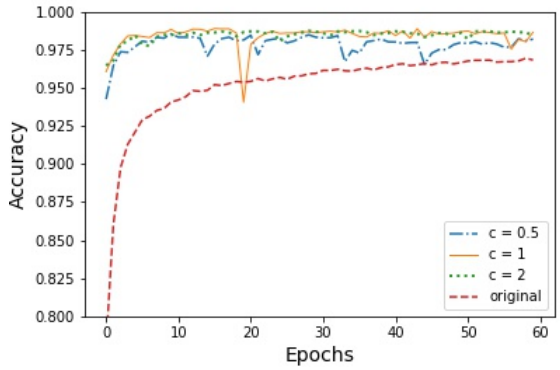


**Figure 7.2: La performance des modèles avec les differentes gammes verticales pour la base de donnée de MNIST**

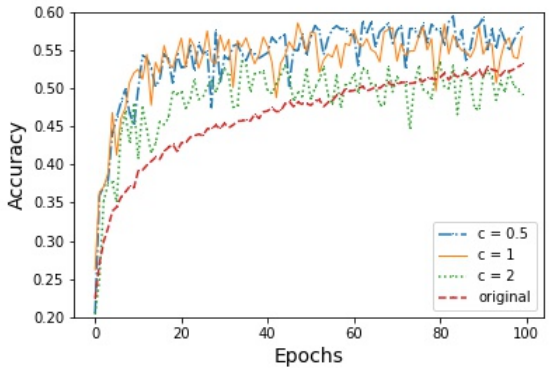


**Figure 7.3: La performance des modèles avec les differentes gammes verticales pour la base de donnée de CIFAR-10**

On peut observer des oscillations dans le diagramme de la performance qui sont causées par la gamme limitée de Tanh. Alors, les résultats d'élargir de gamme sont présentés dans les figures 7.4 and 7.5.

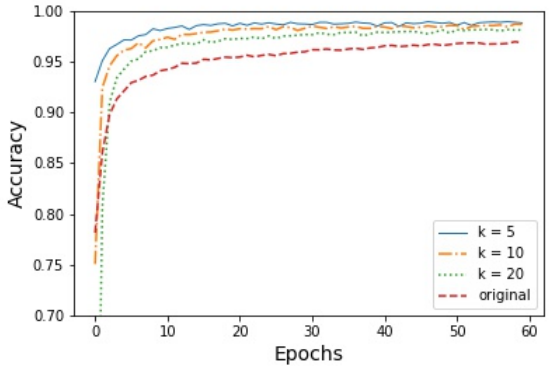


**Figure 7.4: La stabilisation de la performance avec une extension de la gamme horizontale du Tanh pour la base de donnée de MNIST**

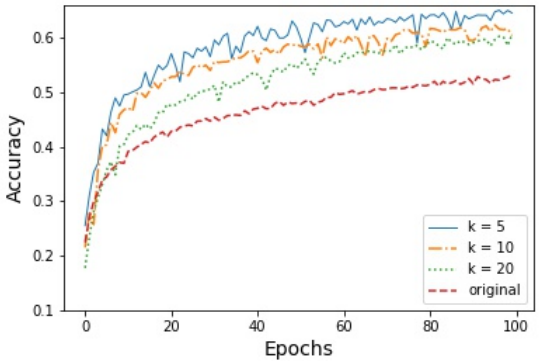


**Figure 7.5: La stabilisation de la performance avec une extension de la gamme horizontale du Tanh pour la base de donnée de CIFAR-10**

Comme ces figures démontrent, l'extension de la gamme horizontale ou autrement dit la gamme de l'activation est bien efficace dans le sens de la stabilisation de la performance du modèle.

Il faut noter que la sélection de type de filtre est très importante dans l'efficacité de l'approche. Un bon exemple est la fonction de sigmoïde qui est visualisée dans la figure 7.6.

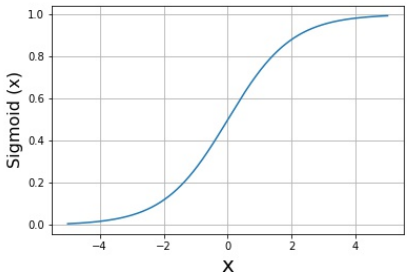


**Figure 7.6: La fonction de sigmoïde**

La sigmoïde est similaire à Tanh, mais ne respecte pas les signes des entrées. Par la suite, il y aura un gros dommage à l'utilité du modèle.

## 7.9 Proposition pour le travail futur

Afin d'approuver l'efficacité de la méthode proposée en pratique, on peut concevoir les attaques en ciblant la confidentialité aux modèles d'apprentissage profond qui sont équipés par l'algorithme

proposé dans ce projet. Si le modèle est bien immunisé contre ces attaques, on peut conclure le succès pratique de l'approche.

Un autre travail pour le futur peut être de développer les modèles génératifs en s'inspirant de l'algorithme proposé. De tels modèles peuvent être utilisés pour générer les données médicales artificielles et les partager publiquement au lieu de données réelles.

# References


[1] http://yann.lecun.com/exdb/mnist.

[2] https://courses.cs.duke.edu//fall18/compsci590.1/lectures/10-10-instructor.pdf.

[3] https://en.wikipedia.org/wiki/MNIST-database.

[4] https://github.com/tensorflow/privacy .

[5] https://Google Colabcolab.research.google.com,.

[6] https://medium.com/@shaistha24/differential-privacy-definition-bbd638106242.

[7] https://neurohive.io/en/popular-networks/alexnet-imagenet-classification-with-deep-convolutional-neural-networks/.

[8] https://paperswithcode.com/dataset/cifar-10.

[9] https://www.boost.org.

[10] https://www.cs.toronto.edu/ kriz/cifar.html.

[11] https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/.

[12] https://www.kaggle.com/sulianova/cardiovascular-disease-dataset.

[13] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep Learning with Differential Privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (ACM CCS)*, pages 308–318, 2016.

[14] M. Abadi and et al. TensorFlow: A System for Large-Scale Machine Learning. *Symposium on Operating Systems Design and Implementation*, pages 265–283, 2016.

[15] A. Act. Health Insurance Portability and Accountability Act of 1996. *Public law, vol. 104*, page 191, 1996.

[16] R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Physical Review E, vol. 64, no. 6, p. 061907*, 2001.

[17] K. Armanious, C. Jiang, M. Fischer, T. Küstner, K. Nikolaou, S. Gatidis, and B. Yang. MedGAN: Medical Image Translation using GANs. *arXiv:1806.06397*, 2018.

[18] S. Augenstein, H. B. McMahan, D. Ramage, S. Ramaswamy, P. Kairouz, M. Chen, R. Mathews, and B. Arcas. Generative Models for Effective ML on Private, Decentralized Datasets. *ICLR*, 2020.

[19] K. Bonawitz, F. Salehi, J. Konečný, B. McMahan, , and M. Gruteser. Federated Learning with Autotuned Communication-Efficient Secure Aggregation. *arXiv:1912.00131*, 2019.

[20] R. Bousseljot, D. Kreiseler, and A. Schnabel. Nutzung der ekg-signaldatenbank car- diodat der ptb über das internet. *Biomedizinische Technik/Biomedical Engineering, vol. 40, no. s1*, pages 317–318, 1995.

[21] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Sackinger, and R. Shah. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence, 7 (04)*, pages 669–688, 1993.

[22] T. Brown and et al. Language models are few-shot learners. *arXiv:2005.14165*, 2020.

[23] N. Carlini, C. Liu, U. Erlingsson, J. Kos, and D. Song. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. *Usenix 28*, 2019.

[24] N. Carlini, F. Tramer, M. J. E. Wallace, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, A. Oprea, and C. Raffel. Extracting Training Data from Large Language Models. *arXiv:2012.07805*, 2020.

[25] M. Chen and et al. Gmail smart compose: Real-time assisted writing. *arXiv:1906.00080*, 2019.

[26] C. Dwork. Differential privacy. *Encyclopedia of Cryptography and Security*, pages 338–340, 2011.

[27] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 2014.

[28] K. Fernandes, J. S. Cardoso, and J. Fernandes. Transfer learning with partial observability applied to cervical cancer screening. *Iberian Conference on Pattern Recognition and Image Analysis, pp. 243–250, Springer*, 2017.

[29] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circulation, vol. 101, no. 23*, pages e215–e220, 2000.

[30] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and A. Ramage. Federated Learning for Mobile Keyboard Prediction. *arXiv:1811.03604*, 2019.

[31] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[32] J.Jordon, J.Yoon, and M.vanderSchaar. PATE-GAN:GeneratingSyntheticDatawithDifferentialPrivacy Guarantees. *ICLR*, 2019.

[33] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-Wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific data, vol. 3*, page 160035, 2016.

[34] P. Kairouz and et al. Advances and Open Problems in Federated Learning. *arXiv:1912.04977*, 2019.

[35] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet Classification with Deep Convolutional Neural. *NIPS*, 2012.

[36] N. Li, T. Li, and S. Venkatasubramanian. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. *IEEE 23rd International Conference on Data Engineering*, 2007.

[37] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov. Exploiting unin- tended feature leakage in collaborative learning. *IEEE*, 2019.

[38] I. Mironov. Rényi differential privacy. *IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275, 2017.

[39] G. B. Moody and R. G. Mark. The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine, vol. 20, no. 3*, pages 45–50, 2001.

[40] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. *IEEE Symposium on Security and Privacy*, pages 111–125, 2008.

[41] M. Nasr, R. Shokri, and A. Houmansadr. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. *arXiv:1812.00910*, 2020.

[42] M. Nasr, R. Shokri, and A. Houmansadr. Improving Deep Learning with Differential Privacy using Gradient Encoding and Denoising. *arXiv:2007.11524*, 2020.

[43] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim. Data synthe- sis based on generative adversarial networks. *Proceedings of the VLDB Endowment, vol. 11, no. 10*, pages 1071–1083, 2018.

[44] A. Radford, J. W. R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

[45] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. 2018.

[46] A. Radford, J. Wu, D. Amodei, D. Amodei, J. Clark, M. Brundage, and I. Sutskever. Better language models and their implications. *OpenAI Blog*, 2019.

[47] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[48] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[49] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan. Can You Really Backdoor Federated Learning? *arXiv:1911.07963*, 2019.

[50] I. V. Tetko, D. J. Livingstone, and A. I. Luik. Neural network studies. 1. comparison of overfitting and overtraining. *Journal of Chemical Information and Computer Sciences, vol. 35(5)*, pages 826–833, 1995.

[51] A. Torfi, M. Beyki, and E. Fox. On the Evaluation of Generative Adversarial Networks By Discriminative Models. *25th International Conference on Pattern Recognition (ICPR)*, pages 991–998, 2021.

[52] A. Torfi and E. A. Fox. CorGAN: Correlation-Capturing Convolutional Generative Adversarial Networks for Generating Synthetic Healthcare Records. *FLAIRS-33*, 2020.

[53] A. Torfi, E. A. Fox, and C. K. Reddy. Differentially Private Synthetic Medical Data Generation using Convolutional GANs . *arXiv:2012.11774*, 2020.

[54] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou. Differentially Private Generative Adversarial Network. *arXiv:1802.06739*, 2018.

[55] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. *IEEE CSF*, 2018.

[56] W. Zhu, P. Kairouz, B. McMahan, H. Sun, and W. Li. Federated Heavy Hitters Discovery with Differential Privacy. *arXiv:1902.08534*, 2020.