Differential Privacy Preservation in Robust Continual Learning

Ahmad Hassanpour\textsuperscript{1}, Majid Moradikia\textsuperscript{2}, Bian Yang\textsuperscript{1}, Ahmed Abdelhadi\textsuperscript{2}, Christoph Busch\textsuperscript{1}, Julian Fierrez\textsuperscript{3}

\textsuperscript{1}Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Norway
\textsuperscript{2}Engineering Technology Department at University of Houston, USA
\textsuperscript{3}Department of Computer Engineering Systems, Universidad Politécnica de Madrid, Spain

Corresponding author: Ahmad Hassanpour (e-mail: Ahmad.Hassanpour@ntnu.no).

This paragraph of the first footnote will contain support information, including sponsor and financial support acknowledgment. For example, “This work was supported in part by the U.S. Department of Commerce under Grant BS123456.”

ABSTRACT Enhancing the privacy of machine learning (ML) algorithms has become crucial with the presence of different types of attacks on AI applications. Continual learning (CL) is a branch of ML with the aim of learning a set of knowledge sequentially and continuously from a data stream. On the other hand, differential privacy (DP) has been extensively used to enhance the privacy of deep learning (DL) models. However, the task of adding DP to CL would be challenging, because on one hand the DP intrinsically adds some noise that reduce the utility, on the other hand the endless learning procedure of CL is a serious obstacle, resulting in the catastrophic forgetting (CF) of previous samples of ongoing stream. To be able to add DP to CL, we have proposed a methodology by which we cannot only strike a tradeoff between privacy and utility, but also mitigate the CF. The proposed solution presents a set of key features: (1) it guarantees theoretical privacy bounds via enforcing the DP principle; (2) we further incorporate a robust procedure into the proposed DP-CL scheme to hinder the CF; and (3) most importantly, it achieves practical continuous training for a CL process without running out of the available privacy budget. Through extensive empirical evaluation on benchmark datasets and analyses, we validate the efficacy of the proposed solution.

INDEX TERMS Differential Privacy, Continual Learning, Deep learning, Privacy

I. INTRODUCTION

Recently, deep learning (DL) models have shown significant improvement as compared to the human decision making on different tasks [1]-[5]. Despite the striking results, since DL models are built upon the static models, they cannot be applied simply over data streams. More explicitly, a time frame of data stream may vanish soon due to storage constraints or privacy issues, which requires a dynamic training process to begin upon receiving the new data. This gap motivates the researchers to develop DL models, able to adapt frequently and resume learning over time. A typical example of such a system is human cognition by which one tends to learn concepts sequentially. One prominent feature of such a system is that old concepts might be revisited though it is not necessary to keep them in mind [6]. By contrast, conventional DL models cannot learn in this way and thus they suffer from catastrophic forgetting (CF) of old concepts upon learning new ones [7]. Hence, conventional DL (CDL) models often concentrate on static tasks whose data are shuffled to guarantee the independent and identically distributed (\textit{i.i.d.}) requirement. Despite performance improvement, CDL models cannot be applied over data streams as the training data is revisited over several computations. To circumvent this issue while preventing the CF, described above, Continual Learning (CL) comes into play, aimed at gradually extending attained information to be exploited for future learning.

In real world, DL algorithms are extensively vulnerable to security attacks e.g., adversarial example where an adversary fool the DL via perturbation samples [8][9]. Based on the knowledge of adversaries from the target model, the adversarial attacks belong to one of the main group of: white-box, gray-box, and black-box attacks. In black-box attack model, the attacker is not able to access to the model weights; whilst in the white-box attack, the attacker has complete access to the architecture and weights of the model. In black-box attack model, the attacker cannot access to the model weights; whilst in the white-box attack, the attacker complex access to the architecture and weights of the model.
To confront with such attacks, three well-known methods have been broadly used in several literatures: fully homomorphic encryption (HE) [10], [11], [39], k-anonymity [12], and differential privacy (DP). Although the HE offers strong data privacy preservation, it is ineffective in DL models owing to the computational burden imposed due to the dimension of training datasets. On the other hand, k-anonymity also performs weakly when facing large datasets [13][14]. Thus, both HE and k-anonymity are inefficient in case of data stream in which a large amount of data is coming in over a long period of time and it is not practically possible to keep the entire dataset in memory at once.

Recently, DP has attracted a great deal of attention in DL-based solutions due to providing the capability of analyzing a dataset without disclosure of an individual’s information for DL models [41]. The main goal of such a system is to control the cost of losing privacy, called privacy budget (PB), so that it should not exceed the predefined global privacy budget (GPB). Notably, without adding computational burden, it tries to preserve the privacy of data by perturbing the weights, objective function, or outputs of DL models systematically [15][40]. The noise added to the dataset will affect the privacy-utility trade-off. Explicitly, upon increasing the amount of noise the dataset would be useless, while reducing the noise up to the little values will degrade the privacy. Concerning using DP in DL models, a differentially private version of the SGD algorithm, is proposed in [16], where the amount of random noise and the privacy budget (PB) constantly increase upon growing the number of training epochs which is in contrast to the limited PB in practice. Shokri et al. [27] proposed a method for incorporating DP into distributed DL. They designed a practical framework that allows multiple clients to collaboratively train a DL model without sharing their training data.

To the best of our knowledge, despite the applicability of DP in DL models (DP-DL) [16][17] and stream data [18][19] separately, there is no study on adding DP into CL models such that all characteristics of a CL process meet, so far. However, this task would be challenging, because on the one hand the DP intrinsically adds some noise that reduce the utility, and on the other hand the endless learning procedure of CL is a serious impediment. Thus, to compromise between privacy and utility in the proposed DP-CL, we need to rethink and redesign the existing DP-DL models to be adapted for the CL process. To elaborate further, on the one hand difficulties arise from two significant characteristics of the CL process as follows:

**C1** The learner used in the CL process should be able to learn the new received data continuously and endlessly.

**C2** To mitigate the CF, a small portion of data or model’s parameters needs to be stored for future learner’s computations.

On the other hand, a DP-enabled algorithm has two significant limitations as follows:

**L1** Each computation of the DL algorithm not only increases the bound over data leakage, but also consumes a portion of predefined privacy budget (PB). Although it is desired that the leakage bound does not exceed the available PB, it has been shown in [16], [17] that a DL process run out of the PB after a few computations.

**L2** DP tends to perturb the data or the algorithm’s parameters by adding noise, leading to diminishing the utility.

In our proposed approach where we aim to add DP into CL, we encounter the following issues:

**I1** the L1 is in contrast to the C1, as the available PB is limited, preventing the CL process to be continued endlessly.

**I2** Moreover, lowering the utility mentioned in L2 exacerbates the detrimental impact of CF described in C2, which motivates us to look for a robust design.

In this paper, we proposed a novel robust DP-CL approach by which we tackle these issues effectively. To the best of our knowledge, this is the first paper which studies the integration of DP into CL by addressing I1 and I2, concurrently. Against this background, our contributions and novelties can be summarized as follows:

- To address I1, (or more explicitly to be able to continue the training process endlessly without running out of the PB), at each iteration of the training process, the spent PB is measured for each training sample and learner. Once the resultant PB is being exceeded to the predefined GPB, the previous samples in the temporary memory are replaced by new zero-PB ones, coming from the data stream. Similarly, we will do the same approach to substitute the previous learner with a new zero-PB one.

- To overcome I2 (or more explicitly to combat the CF), we further incorporate a robust procedure into the proposed DP-CL scheme, including three steps 1) adding a new noisy layer to the DL architecture, 2) refining the CL algorithm’s objective function (OF), and 3) filling the episodic memory (EM) more effectively. We will detail throughout the paper that how each of these steps can help to increase the robustness of our proposed algorithm. We will experimentally show that each of these steps can assist to make the DP-CL process more robust against white-box attacks.

- To evaluate the effectiveness of the proposed robust DP-enabled CL process, different adversarial attacks have been used to fool the trained models. Particularly four types of white box attacks have been used including: 1) Fast Gradient Sign Method (FGSM) [29], 2) Iterative-FGSM (I-FGSM)[32], 3) Momentum Iterative Method (MIM) [30], and 4) the attack proposed by Madry et al. [31]. Our simulation results confirm that the proposed method yields the stable and steady outputs, even when facing of such strong attacks.

The rest of the paper is organized as follows. Recent works in the context of using DP in machine learning algorithms are reviewed in Section II. A brief description of CL models, DP, and adversarial attacks are presented in Section III as a preliminary. A detailed description of the proposed...
methodology is provided in Section IV. The experimental results and discussions are reported in Sections V and VI. Finally, Section VII presents the conclusion.

II. RELATED WORKS

So far, several papers have attempted to add DP to DL algorithms [16] [20] [21] [22]. This task would be challenging in terms of limited PB and the privacy-utility tradeoff requirement. Upon DL models progresses, for example when we aim to apply DP on those DL models using dynamic dataset, some other demands will ensue which exacerbate the abovementioned issues. Some of the most prominent demands, which are close in spirit to the requirements of CL, as we need here, are listed as follows:

R1: Endless execution
R2: Multiple usage of data subsets
R3: Capability of changing DP parameters during the execution

Satisfying all the R1-R3 together is hard, therefore related papers address only one or two of these requirements. Along this line, two recent DL-based papers of [23], [24] have enabled DP to work on growing databases (dynamic datasets). More explicitly, to address R1 Cummings et al. have considered a scheduler to re-execute the DL algorithms whenever the new received data is sufficient [24]. To achieve the desired privacy loss, the privacy parameter (ε) is reduced upon increasing the size of dataset.

In order to jointly address R1 and R3, one can partition the data stream into blocks. After applying the DP on data blocks, each of which is fed into an individual learner, the learners’ outputs are aggregated [25]. Accordingly, the conventional composition theorem can be exploited to calculate the privacy loss at the block level. Now, deploying the conventional composition theorem, the data blocks incur no privacy loss from the previous learners and thus the requirements of R1 and R3 are supported. However, it is against R2 as each learner cannot access other learners’ blocks.

In another scenario, aimed at addressing R2 and R3, Lecuyer et al. have proposed a DP-DL platform including several pipelines, each of which comprised a DL algorithm, training endlessly from the growing database. Note that, since each block of data might be used by different DL algorithms corresponding to the pipelines, calculating the PB spent by the whole pipelines would be challenging. To reach this goal, the authors of [23], have proposed the so-called block composition theorem by which the DL algorithms are executed till the PB consumption of each block does not exceed the predefined GPB. To achieve the desired accuracy, with the aim of re-training the pipelines, either the relevant PB of each pipeline or the number of available samples is doubled. Therefore, each pipeline can continue till the consumed PB is smaller than GPB, violating R1.

III. PRELIMINARIES

A. CONTINUOUS LEARNING

A typical CL process, e.g., A-GEM [26], has generally two important features. First, the used learner in the CL process should be able to learn the new received data continuously and endlessly (growing database). In other words, the commonly used CL model can be fed by consecutive parts of a data stream, each of these parts includes multiple number of samples and corresponds to a particular task. Second, a small part of data will be stored model’s parameters or training data for future learner’s calculations to prevent catastrophic forgetting. Thus, CL refers to the ability of a system to learn over time from a continuous stream of data without having to revisit previously encountered training samples.

First, the ith sample of the training set D includes a triplet (x_i, t_i, y_i), where x_i ∈ X_t is a feature vector, t_i ∈ T is a task descriptor, and y_i ∈ Y is a target vector. In general, CL algorithms aim to learn a predictor f_t: X × T → Y in which θ denotes the relevant tunable parameters of predictor f.

To get more insight, in the following we succinctly explain A-GEM [26]. Using the A-GEM algorithm, the detrimental impact of catastrophic forgetting can be alleviated by allocating an episodic memory (EM), which is denoted by M and equally divided between total T tasks, to store some training samples randomly for each task k. These stored samples assist the DL model to maintain its performance for previous tasks. For a total number of T tasks, if we let D_k represents the relevant data with respect to previous tasks, i.e., k ≤ T, the abovementioned explanations can be mathematically formulated as the following constrained optimization problem

\[
\min_{\theta} \mathcal{L}_{AG}(f_\theta, D_t), \quad \text{s.t.} \quad \mathcal{L}_{AG}(f_\theta, M_k) \leq \mathcal{L}_{AG}(f_{\theta^{-1}}, M_k) \quad \forall k < t
\]

where the objective function \( \mathcal{L}_{AG}(f_\theta, D_t) \) stands for the loss of the A-GEM model on the current task t. Using the stored data of previous tasks in EM(M_k), the constraint attempts to reduce the loss of the model with respect to the loss of previous tasks.

B. DIFFERENTIAL PRIVACY

The DP technique prevents the disclosure of information corresponding to individual records of database D against any adversarial processing. Using DP, the records are contaminated with noise through a randomized algorithm A: B → R. The DP is often characterized by the parameters (ε, δ) where the privacy budget (PB) ε > 0 and the broken probability δ ∈ [0, 1] are control parameters to tune the strength of the privacy preservation. Thus, given the randomized algorithm A, the following inequality must hold true to satisfy the (ε, δ)-DP:

\[
P[A(D) \in O] \leq e^{\varepsilon}P[A(D') \in O] + \delta
\]

where \{D, D'\} ∈ B are two neighboring inputs and O ⊆ R represents any subsets of outputs. Besides, \( P[.] \) denotes the

---

1 We can interchangeably use the word of "block" and "sample" throughout the paper.
probability function with the space over the coin flips of the algorithm \( A \). The Eq. (2) implies that if we change a tuple in the database slightly, the output distribution does not vary significantly.

Now, in the following we invoke the definitions of some basic concepts used in DP, which lay the grounds for a better understanding.

1) **Privacy loss** [15]: Privacy loss is a random variable dependent on the random noise added to the algorithm. For neighboring databases \( D, D' \), auxiliary input \( aux \), and an outcome \( O \subseteq R \), define the privacy loss at \( O \) is defined as:

\[
c(O; A, aux, D, D') = \frac{P[A(aux,D)=O]}{P[A(aux,D')=O]} \tag{3}
\]

2) **Gaussian mechanism** [15]: This mechanism will be used in this paper. Using this kind of mechanism the white Gaussian noise \( \mathcal{N}(0, \sigma^2) \) is added to the output entries. Given \( \varepsilon \in (0, 1) \), the Gaussian mechanism with

\[
\sigma \geq \sqrt{2ln\left(\frac{2e^2}{\delta}\right)} \Delta_A \in (\varepsilon, \delta) – \text{DP} \quad \text{and the } l_2 \text{ sensitivity parameter } \Delta_A \text{ therein is defined as } \Delta_A = \max_{D,D'} \| A(D) - A(D') \|_2.
\]

3) **Composition theorem**: If we consider several DP subroutines, each of which applied into separate algorithms to reach a specified privacy level, incorporation of these DP subroutines relying on the composition property significantly degrades the privacy such that it is less than that of achieved by a single subroutine. In particular, based on one kind of composition theorem, namely “basic composition theorem” [28], considering \( \ell \) subroutines each of which is \((\varepsilon, 0)\)-differentially private, the privacy of an algorithm including a combination of these subroutines is degraded up to the bound of \((\varepsilon \ell, 0)\) as compared to the single subroutine.

### C. ADVERSARIAL EXAMPLES

Adversarial examples are a kind of attack against ML models, where the attacker add a small perturbation \( \alpha \triangleq \{a_i\}_{i=1}^\ell \in R^\ell \) to the given input \( x \triangleq \{x_i\}_{i=1}^\ell \in R^\ell \) of the DL model, resulting in a considerable change at the output \( y \triangleq \{y_i\}_{i=1}^\ell \in R^\ell \). The perturbation is usually specified by a \( l_p \) norm ball of radius \( \mu \), i.e., \( \Theta_\mu \triangleq \{ \alpha : \| a \|_p \leq \mu \} \) where \( p \in \{1, 2, \infty \} \) [29]. To evaluate the robustness of the proposed method, particularly four well-known white box attack algorithms are utilized to generate the adversarial samples: i) Fast Gradient Sign Method (FGSM) [29], ii) Iterative-FGSM (I-FGSM) [32], iii) Momentum Iterative Method (MIM) [30], and iv) the attack proposed by Madry et al., [31], are utilized to generate the adversarial samples.

### IV. PROPOSED ROBUST DP-ENABLED CONTINUAL LEARNING

In this section we present the notion of adding DP to A-GEM algorithm and then make the proposed DP-CL model robust. Thus, by considering the characteristics of CL processes (i.e., C1 and C2) and created limitations by DP (i.e., L1 and L2), we address the A-GEM requirements and finally propose a scheme for a DP-enabled CL process during the next subsections (i.e., 4.1 and 4.2). Then, to overcome catastrophic forgetting and reduce the impact of attacks, in subsection 4.3 we add robustness methods to DP-CL: 1) modifying the DL architecture, 2) refining the objective function (OF) of the A-GEM algorithm, and 3) filling the EM more effectively.

#### A. ADDING DP TO CL PROCESS

Given the properties of DP, as discussed above, the problem of adding DP to CL would be challenging. First, adding perturbations to the learner(s) will affect on the training accuracy and consequently worsen CF. Moreover, the composition theorem, imposes some predefined bounds for DP algorithms, including the number of subroutines (iterations \( k \)) and privacy parameters \((\varepsilon, \delta)\). As per requirements of a CL process these variables need to be updated and thus a CL-based composition theory must satisfy the following requirements:

R1: Endless execution

R2: Handling the concern of overlapping data stored in EM

R3: Capability of updating DP parameters during the execution

Hence, it is required to think about how to satisfy each of R1-R3 which are responded to, in the sequel.

1. **How to add DP while CL is executed endlessly (Addressing R1)?**

The everlasting approach of CL is a serious impediment to deploy either of the proposed solutions in [23] or [24]. In particular, if one intends to add DP to CL, the limited GPB hinders the process to be continued. To deal with these problems, we here propose a novel learning procedure, comprised of several learners in \( L = \{l_1, l_2, \ldots \} \), each of which is trained sequentially on a specific part of the data stream. Before exceeding the PB consumed by each learner from the GPB, we add a zero-PB (ZPB) learner to the process. This newly added learner starts from the point where the previous one has been halted and would be continued using the untouched data coming from the dynamic database \( S = \{b_1, b_2, \ldots \} \) (and/or the stored data in EM \( M = \{b_1, b_2, \ldots \} \), where \( b_i \) shows the \( i \)th block of database).

Based on the discussion above, selecting an appropriate composition theorem is of vital importance to calculate PB for each training step through which, we can determine the halting time of the current learner \( (t_c) \), learning the current task \( (t_c) \). We here use the moments accountant algorithm (MAA) [16], appropriate for computing the PB for each data
access in the DL models. When $l_c$ runs out of the PB, computed by MAA, this learner is left out and added to the set of trained learners $L$, i.e., $L \subseteq \{l_1, l_2, ..., l_{c-1}, l_c\}$ and the learning process will be continued via the next ZPB learner $l_{c+1}$. There are some technical concerns which must be considered in our design, listed as follows:

- **The significance of GPB parameter values ($\epsilon_p, \delta_p$):** Moreover, a large selection of the GPB leads to higher privacy leakage, despite yielding higher accuracy due to injecting less noise into the current learner $l_c$ as well as using fewer number of learners for the whole process. In contrast, although upon reducing the GPB the leakage is decreased, the accuracy is degraded, as well. The performance degradation originates from the fact that, using small PB values not only more noise is fed into the current learner $l_c$, but also more number of learners must be deployed.

- **Keeping the performance while deploying multiple learners:** In the case the PB of $l_c$ reaches to the GPB in the middle of learning $t_c$, leading to degrading the performance of upcoming learner $l_{c+1}$, we proposed early starting (ES) strategy that assists to predict the termination of $l_c$. More clearly, the random initial values of learning parameters $\theta_{c+1}$ which are going to be used by $l_{c+1}$ have not been optimized for the current task $t_c$. To prevent this issue, we propose the ES strategy where the remaining PB, i.e., $(PB_\ell \equiv GPB - PB_{t_c})$ of the $l_c$ is compared with the required PB of $t_{c+1}$ ($PB_{c+1}$), and the $l_c$ continues if and only if $PB_\ell > PB_{c+1}$. To estimate $PB_{c+1}$, since the noise magnitude and the sampling probability (Gaussian probability) is equal during the training process of each learner, it is trivial to calculate the consumed PB of next iterations or the required PB for the next task (i.e., $PB_\ell = PB_{c+1}$). Doing this, $l_c$ will not be halted in the middle of training a task, and each learner starts its training procedure from the beginning of a task.

2. **How to add DP while subsets of data are used repeatedly (Addressing R2)?**

A serious impediment to deploy either of the proposed solutions in [23] or [24] in a CL process, is the data coming from the stream as well as samples stored in the EM to avoid catastrophic forgetting (CF). Note that, although the learners observe most of the data coming from the stream just once, a small portion stored in the EM is observed several times. For each observation the corresponding learner consumes the PBs associated with a portion of the sample stored in EM. Thus, if the spent PB of each stored sample in EM ($PB_{t_i}$) exceeds the GPB, the privacy is compromised. In the following, we elaborate this further.

The samples in EM that have been observed repeatedly, might be observed in different iterations of the learners’ training process. Depending on the privacy loss of the $l_c$ used at each iteration, a portion of the sample’s PB will be consumed and can be stored in $PB_{t_i} = \{PB_{m_{i1}k}, ..., PB_{m_{i}j_{ik}}\}$, where $PB_{m_{i1}k}$ stands for the consumed PB of $i$th iteration of the learner $l_k$. By doing so, we can calculate the total consumed PB for the sample via feeding $PB_{t_i}$ to the Block Composition Theorem (BCT) [23]. Tracking the behavior of $PB_{t_i}$, if it exceeds the GPB, we no longer use that sample in our CL procedure.

Remarkably, to avoid the CF, EM should include some samples for each task. Thus, ZPB samples will be randomly replaced from the stream with ones that are removed at each iteration. We also proposed other different strategies for replacing new samples described in subsection 4.3 (c) to make the DP-CL process more robust. By following this strategy, we can use a subset of data (those stored in EM and their consumed PB is less than GPB) multiple times. Therefore, since there is a limitless of data in real world CL scenarios, the halting of $l_c$ will not occur because of limitation in data PB.

3. **Adaptivity in the choice of DP parameters during the CL process. (Addressing R3.)**

To address the privacy-utility tradeoff, the proposed DP-CL process benefits from an adaptive training procedure such that controls the utility of DP-CL models by using new data and/or changing DP parameters. The block composition theorem allows us to train the used CL algorithms with different PB. For those tasks that have high number of samples in their training set, we will be able to adjust small PB leading to decrease privacy leakage and vice versa. If a model does not reach the pre-defined quality criteria (e.g., an accuracy target) until specific iteration and $PB < GPB$, the model can decrease the added noise ($\sigma$) to its weights, results in expediting increasing accuracy, although PB reaches GPB earlier. On the other hand, if a model reaches the pre-defined quality criteria in a specific iteration and $PB < GPB$, then the model can increase the added noise to its weights to increase the privacy of the model.

### B. DP-CL ARCHITECTURE

The proposed ($\epsilon_p, \delta_p$)-DP-CL Architecture includes three main modules called Learners’ Managing Unit (LMU), Privacy Meter Unit (PMU), and Data Managing unit (DMU). The detailed procedure of our proposed DP-CL method is shown in Algorithm 1, and is conceptually described in the following.
Algorithm 1: Proposed DP-CL

Procedure DP-CL:
// Learners’ Managing Unit (LMU):
While $b_{t_{n}}$ ← ask data from DMU
1. If $l_{c} = \emptyset$ then
2. Initialize $l_{c}$
3. train $l_{c}$ with $b_{t_{n}}$
4. $SPB_{l_{c}} ← PMU(l_{c}, b_{t_{n}})$
5. $DMU(b_{t_{n}}, SPB_{l_{c}})$ // save part of $b_{t_{n}}$ in EM
6. If $SPB_{l_{c}} ≥ GPB$ then
7. $C ← \{l_{c}, (t_{j}, ..., t_{n})\}$
8. $l_{c} ← Initialize$ a new learner
9. // Privacy Meter Unit (PMU):
10. If $l_{c}, b_{t_{n}}$ ← receive request from LMU then
11. LMU ← Calculate the spent PB of $l_{c}$ using MAA
12. $DMU ←$ Calculate the spent PB of each training sample in $b_{t_{n}}$ using BCT
// Data Managing Unit (DMU):
13. If receive data from Stream then
14. Save the data temporary
15. If receive request from LMU then
16. Fetch a batch of data from EM/Stream and send to LMU
17. If receive data from LMU then
18. Store data in EM
19. Store spent PB for each $b_{i}$
20. Update EM by replacing those samples which run out their PB with ZPB samples (Procedure UpdateEpsMem, Alg. 2)
end procedure

The LMU is composed of two sub-modules, called Training Controller (TC) and Data Controller (DC). The TC is responsible for adding new learners to the process, adjusting the $l_{c}$ parameters, saving the $l_{c}$’s parameters, and collecting the information about the tasks corresponding to each learner. Moreover, TC also receives the information related to halting time of a learner from the PMU. Besides, the DC also receives the training data from the DMU and feed them to the $l_{c}$. Additionally, the DC specifies which samples should be saved in the EM and send them to the DMU.

The PMU is responsible for measuring the spent PB of learners and training samples respectively by two sub-modules of Trainer PB Meter Unit (TPBMU) and Data PB Meter Unit (DPBMU). For each training iteration of $l_{c}$, the spent PB will be calculated by TPBMU so that if it exceeds the GPB, the TC will be notified to halt the $l_{c}$. The DPBMU calculates the spent PB for those samples, used in the current iteration of $l_{c}$ and send this information to the DMU. It should be noted that, the PB for all samples will be stored in a sub-module of DMU namely EM Data Controller (EDC), since we may need to remove some samples from EM and replace them by samples whose spent PB is less than GPB.

The DMU is responsible for managing the data and is composed of three sub-modules of Data Controller (DC), EDC, and Stream Data Controller (SDC). The DC fetches the data from the stream or EM by sending a request to EDC or SDC. It also collects the spent PB of the training samples stored at EM or coming from the stream. SDC also stores the received data from the stream into a temporary database. Upon receiving a request from DC or EDC, the SDC will deliver the requested data to those modules. The EDC is responsible for adding/removing the samples having spent PB more than GPB. When the privacy loss for a sample reaches to GPB, the sample will be removed.

C. ADDING ROBUSTNESS To DP-CL

To combat the CF and mitigate the effect of attacks, we incorporate a robust procedure into the proposed DP-CL scheme, including three steps 1) modifying the DL architecture, 2) refining the OF of the A-GEM algorithm, and 3) filling the EM more effectively. In what follows we elaborate each of these steps separately. The first method has the aim of reducing the attacks success rate by making the CL parameters noisy, and the other two methods assist to prevent CF. However, our experiments show that the last two proposed methods can also decrease the attacks success rates to some extent

1) MODIFYING THE DP-CL ARCHITECTURE

To provide a robust DP-CL architecture, we change each learner’s architecture by adding a DP noise layer, that provide $(\epsilon, \delta)$-DP guarantees, after the first layer of each learner. Adding the DP noisy layer can be considered as a certified defense against $p$-norm bounded adversarial example attacks proved by [33]. More explicitly, in accordance with the sensitivity ($\Delta$) and size of the first layer ($b_{t_{1}}$), a noise with zero mean and standard deviation $\sigma = \sqrt{2\ln\left(\frac{1.25}{\delta}\right)} \Delta_{p, 2} L / \epsilon$ is produced by Gaussian mechanism (noise($\Delta, L, \epsilon, \delta$), line 6, Algorithm 2).

2) REFINING THE OBJECTIVE FUNCTION OF THE A-GEM ALGORITHM

Furthermore, to prevent CF, we incorporate a robustness condition into the training stage (called robust-A-GEM hereafter). In this regard, it should be noted that, the expected output of the randomization mechanism $A$ for class $j$ during the training of current task $t$ should be greater or equal to that of the previous task, i.e., $\overline{E}_{t}(A_j(x)) - E_{t-1}(A_j(x)) \geq 0$ where $E_{t}(A_j(x)) = \frac{1}{n} \sum_{i} a_{j,n}(x)$ and $n$ denotes the number of invocation of $A(x)$ and $a_{j,n}(x)$ demonstrates the $n^{th}$ draw from the distribution of the randomized function $A$ on the $j^{th}$ label. To meet this condition, the computed angle between the gradient of $l_{c}$ for $t_{c}$ with respect to label $j$ ($\hat{g}_{j}$) and the gradient of $l_{c}$ for the previous tasks ($\forall k < t$) for label $j$ ($g_{j,k}$) should be greater than zero ($\langle \hat{g}_{j}, g_{j,k} \rangle \geq 0$).

Moreover, instead of $n$ times invoking $A(x)$ for a specific sample $x$, to calculate $E_{t}(A_j(x))$, we use $n$ samples belonging to the $j^{th}$ class within the current batch, and for calculating $E_{t-1}(A_j(x))$, $n$ samples having label $j$ are chosen from the EM. This notion assists to incorporate this condition to the training process by changing the constraint
Algorithm 2 Robust DP-CL

**Procedure** Train($f_\theta, D_{train}, D_{test}$)

1. **Input:** Datasets $D_{train}$ and $D_{test}$, batch size $m$, learning rate for each task $\zeta$, gradient norm band $C$, privacy budget $\epsilon$, broken probability $\Delta$, robustness parameters $\sigma_\gamma$, $\epsilon_\gamma$, $\delta_\gamma$, $\Delta_\gamma$, size of first hidden layer $|h_1|$, $f$ includes $z$ hidden layers $\{h_1, \ldots, h_z\}$, EM depicted by $M$.
2. **Initialize** $\theta$ randomly
3. $y \leftarrow N(0, \sigma^2|h_1|)$
4. for $t = \{1, \ldots, T\}$ do
   5. for $(x, y) \in D_{train}$ do
   6. for $(x, y) \in D_{train}$ do
   7. $h_1 \leftarrow W_1^t x + \gamma$
   8. $(x_{ref}, y_{ref}) \leftarrow M(y)$
   9. $g_{\text{ref}} \leftarrow \nabla_{\theta} l(f_{\theta}(x_{ref}, t), y_{ref})$
   10. $g \leftarrow \nabla_{\theta} l(f_{\theta}(x, t), y)$
   11. **Clipping gradient and adding noise**
      $g_{\text{ref}} \leftarrow \frac{1}{m} \left( g_{\text{ref}}^\top + N(0, \sigma^2 C^2 I) \right)$
   12. $g' \leftarrow \frac{1}{m} \left( g_{\text{ref}}^\top + N(0, \sigma^2 C^2 I) \right)$
   13. If $g' g_{\text{ref}} \geq 0$ then
      14. $\bar{g} \leftarrow g'$
   15. else
      16. $\bar{g} \leftarrow g' - \frac{g^\top g_{\text{ref}}}{g_{\text{ref}}^\top \bar{g}_{\text{ref}}} g_{\text{ref}}$
   17. end if
   18. $\theta \leftarrow \theta - \zeta \bar{g}$
   19. end for
   20. UpdateEpsMem($M, D_{train}, T$)
21. end for

end procedure

**Procedure** UpdateEpsMem ($M, D_{train}, T, GPB$)

22. // remove stored samples in $|M|$ with high PB
23. for $i = \{1, \ldots, |M|\}$ do
24. if spent $P_B > GPB$ do
25. remove $(x_i, y_i, t_i) \leftarrow M_i$
26. $(x) \leftarrow D_{train}^s$, which meet robustness condition
27. $M_i \leftarrow (x)$
28. end for
29. // Add a few samples from current task
30. $s \leftarrow \frac{|M|}{T}$
31. for $i = \{1, \ldots, s\}$ do
32. $(x, y) \leftarrow D^s_{\text{train}}$
33. **If** $(x, y)$ meet the robustness condition then
34. $M \leftarrow (x, y)$
35. end for
36. return $M$

end procedure

**Procedure** Evaluation ($f_\theta, D_{test}$)

37. $a \leftarrow 0 \in \mathbb{R}^2$
38. for $t = \{1, \ldots, T\}$ do
39. $a_t \leftarrow 0$
40. for $(x, y) \in D_{test}$ do
41. $a_t \leftarrow a_t + \text{ACCURACY}(f_{\theta}(x, t), y)$
42. end for
43. $a_t \leftarrow \frac{a_t}{\text{len}(D_{test})}$
44. end for
45. return $a$

end procedure

A-GEM objective function. Therefore, we modify the optimization function as below:

$$\min_{\tilde{g}} \frac{1}{2} \| g - \tilde{g} \|_2^2 \quad \text{s.t.} \quad \langle \tilde{g}_j, g_{jk} \rangle \geq 0 \quad \forall k < t$$

where $g_{jk}$ will be the average gradient from the previous tasks with respect to $j$th class. By doing that, the new updated rule will be obtained as follows:

$$\tilde{g} \leftarrow g - \frac{g^\top g_{\text{ref}}}{g_{\text{ref}}^\top g_{\text{ref}}} g_{\text{ref}}$$

(5)

The proof of this update rule is given in Appendix A.

3) FILLING THE EM EFFICIENTLY

The easy-to-forget samples (which worsen CF) which classify correctly with small robust boundary during the training process have a chance to enter EM. Therefore, having such samples which are not a good representative of their corresponding classes in EM leads to CF during the learning of next tasks. Particularly this issue happens if the computed angle between the gradient vector of the samples extracted for class $j$ from EM $(\tilde{g}_j)$ and the proposed gradient $(g)$ at current iteration is larger than zero. Here, we propose a robustness condition by which if a sample meets this condition, then it will be added to the EM (called efficient-EM). For sample $x_2$ located in a batch including $n$ samples, the robustness condition calculated as follow:

$$E^b_t \left( f_j(x_2) \right) - \max_{i \neq j} E^{ub}_t \left( f_i(x_2) \right) \geq \frac{1}{n} \sum_{i=1}^{n} E^b_t \left( f_j(x_2) \right) - \max_{i \neq j} E^{ub}_t \left( f_i(x_2) \right)$$

$$+ e - E^b_t \left( f_j(x_2) \right)$$

$E^b_t \left( f_j(x_2) \right)$ and $E^{ub}_t \left( f_j(x_2) \right)$ are the $\eta$-confidence lower and upper bound, respectively. We estimate these bounds using Hoeffding’s inequality with probability $\eta$,

$$E^b_t \left( f(x) \right) \leq E \left( f(x) \right) - \frac{1}{2n} \ln \left( \frac{2y}{1-\eta} \right)$$

$$E \left( f(x) \right) + \frac{1}{2n} \ln \left( \frac{2y}{1-\eta} \right) \leq E^{ub}_t \left( f(x) \right)$$

for $y^{th}$ label (Lines 26, 33, Algorithm 2).
D. THE PROPOSED ROBUST DP-CL ALGORITHM

The proposed robust DP-CL algorithm (shown in Algorithm 2) includes three procedures called Train, UpdateEpsMem, and Evaluation. The Train procedure takes the train and test data, as well as the $l_c$’s parameters. Considering the size of first hidden layer, a generated random Gaussian noise (line 3), is added to the first hidden layer (line 6). By wisely sampling from the EM (considering the notion presented at section 4.3b; line 7), the gradient for the current batch (line 9) and the sampled batch (line 8) have been calculated. Then, $g$ and $g_{ref}$ are clipped so that its l2-norm is bounded by a predefined gradient clipping bound $C$ and subsequently, a random Gaussian noise $N(0, \sigma^2C^2)$ with a predefined noise scale $\sigma$ is added (Line 11 and 12). Depending on the computed angle between $g'$ and $g_{ref}'$, the new gradient will be applied (lines 13-18). After feeding each batch and updating the $l_c$, the EM will be updated by executing the UpdateEpsMem procedure. During this procedure, we first replace the samples that run out their PB with ZPB ones from $D_{y_{k+1}}^{train}$. Then some samples from the current task which meet the proposed robustness condition (presented at section 4.3c) will be added to the EM. Finally, the Evaluation procedure measures the effectiveness of the training procedure by calculating the accuracy

V. EVALUATION

We have carried out extensive experiments on two benchmark datasets (permuted MNIST and split CIFAR) and evaluate our proposed robust DP-CL process by answering the following questions:

**Q1**: How does the added DP mechanism affect the accuracy of the A-GEM algorithm?
**Q2**: What is the impact of using several learners on the accuracy of the DP-CL process?
**Q3**: How can the ES deal with the performance degradation in the training process?
**Q4**: How the proposed robust DP-CL acts against attacks?
**Q5**: How much data the DP-CL process will need?

Before answering these questions, we will briefly describe the used datasets description, the used DL architectures, the evaluation metrics, and observe the behavior analysis of DP’s parameters in the following subsections.

A. DATASET DESCRIPTION

Two datasets have been considered to train and test the proposed robust DP-enabled CL process. First, Permutated MNIST (PMNIST) [34] is a variant of MNIST dataset including handwritten digits. It consists of 20 tasks each of which is composed of 10 classes, 60,000 training and 10,000 test samples. Each task describes a certain random permutation of the input pixels, applied to the entire images of that task. Split CIFAR (SCIFAR) [35] devides of dividing the original CIFAR-100 dataset [36] into 20 disjoint subsets, each of which is generated through random sampling of 5 classes without replacement from the total number of 100 classes. The whole number of training samples for each task is 2500 whose 20% are allocated for testing. In general, there are two streams of tasks, described by the sequences of datasets $D^{CV} = \{D_1, ..., D_{Tcv}\}$ and $D^{EV} = \{D_{Tcv+1}, ..., D_T\}$ where $D_k = \{(k, l_i, k_i, k_i, p_{k_i})_{i=1}^{\%}\}$ is the dataset of $k$th task. Notably, we have $T_{cv} < T$ and set $T_{cv} = 3$ while $T = 20$ in all our experiments. $D^{CV}$ represents the stream of datasets allocated for cross-validation; this stream allows the learner to replay all samples several times aimed at model hyper-parameters selection as well as system adjustment. By contrast, $D^{EV}$ stands for the actual dataset used for final training and evaluation on the test set. Actually, this means that the model sees the training examples from $D^{EV}$ just one time.

B. NETWORK ARCHITECTURE

Shallow and a deep DL architectures including a fully-connected network with two hidden layers of 256 units each (Figure 2, a) for PMNIST dataset, a reduced ResNet18 (Figure 2 (b)) for SCIFAR dataset like in [37], will be used in our experiments. While the models are randomly initialized, the stochastic gradient descent (SGD) with mini-batch size 10 is used to optimize the network parameters. Similar to the approach in [26], in order to tune the hyper-parameters, the data of the first three tasks is fed into the first learner several times.

C. EVALUATION METRICS

We have used three metrics called Average Accuracy [38], Average Forgetting [38], and Certified Accuracy [33] to evaluate our proposed robust DP-CL model. In the following we briefly define these metrics. The training dataset of each task, $D_{t_{i}}$, consists of a total $B_{t_{i}}$ mini-batches. After each observation of $B_{t_{i}}$, the performance of the learner is examined over the whole tasks on the associated test sets. Let $a_{t_{i,j}} \in [0, 1]$ expresses the accuracy obtained using the test set of task $j$, after the model has been trained with $i$th mini-batch...
of task $t$.

**Average Accuracy** [38], varying between $[0,1]$, can be calculated after completing the continually learning procedure of the A-GEM model with all the mini-batches corresponding to the $t^{th}$ task and is defined as:

$$AA_t = \frac{1}{k} \sum_{j=1}^{k} a_{k,B_j}. j.$$  

**Average Forgetting** [38], varying between $[-1,1]$, is computed after the model has been trained for the tasks $1, 2, ..., t-1$. This metric is defined as $F_k = \frac{1}{k} \sum_{j=1}^{k} f_j$, where $f_j$ is the forgetting measure on task $j$ after the model is trained for the tasks $1, 2, ..., t-1$, obtained as: $f_j = \max_{i \in \{1,2,...,t-1\}}(a_{i,B_j} - a_{k,B_j})$. AF is crucial to be measured after learning the entire tasks for a two-fold reason. On one hand, AF quantifies the accuracy degradation on the earlier tasks, while on the other hand it specifies how fast a model learns a new task.

**Certified Accuracy** [33], varying between $[0,1]$ is defined as $CF = \frac{|\text{test}|}{|\text{test}|}[\text{isCorrect}(x_t) \& \text{isRobust}(x_t)]$, where $|\text{test}|$ is the size of testing set and isCorrect($x_t$) denotes a function returning 1 if the prediction on test sample $x_t$ returns the correct label, and 0 otherwise, and isRobust($x_t$) returns 1 if the robustness size is larger than a given attack bound $\mu_a$ and 0 otherwise.

### D. BEHAVIOR ANALYSIS OF DP’s PARAMETERS

In this section, we aim for observing the behavior of DP parameters $(\epsilon, \delta)$ for two abovementioned DL architectures. To generate the figures, we have exploited the MAA for two various dataset MNIST(a) and CIFAR(b). Figure 3 shows six plots where $\delta$ is calculated for a given $\epsilon = 2$, while $\sigma \in \{2, 4, 8\}$, and $t \in \{1 ... 17\}$. As it can be seen from Figure 3 (a), for the cases $\sigma = 2$ as well as $\sigma = 4$ the value of $\delta$ smoothly increases during the first tasks, while sharply grows for the 4 last tasks. While, in case of $\sigma = 8$ $\delta$ values are smoothly rising for the entire tasks. As it would be expected, the more noise we add to the classifier, the smaller values of $\delta$ would be resulted.

Figure 4 depicts six other plots where $\epsilon$ is calculated for a given $\delta = \frac{1}{10000}$, while $\sigma$, and $t$ are opted same as what we used to generate Figure 3. As it would be expected, the more noise we add to the classifier, the smaller values of $\epsilon$ would be resulted.

### E. PERFORMANCE EVALUATION

Now in the following we respond to Q1-Q5 separately.

**Q1: How does the added DP mechanism affect the accuracy of the A-GEM algorithm?**

To answer this question, we compare the accuracy of A-GEM with or without adding DP. To do so, we execute the DP-A-GEM with different level of noise $\sigma \in \{2, 4, 8\}$. We further consider the high GPB assumption where $\epsilon=2$ for PMNIST while for SCIFAR we have $\epsilon=4$. By doing so, the spent PB will no longer reach to the GPB, and thus no additional learner is needed to be added (i.e., $L = 1$). Figure 5 shows the average accuracy after 5 executions for each configuration on PMNIST (Figure 5 (a)) and SCIFAR (Figure 5 (b)) datasets. As it can be observed, upon increasing the level of noise, the accuracy is reduced so that in case of $\sigma = 8$, a CF phenomenon has been happened, i.e., this can be interpreted from the negative slope of this curve. As another important observation, the results of DP-A-GEM method have less fluctuations and more stable accuracy as compared to the A-GEM method in which DP has been eliminated.

**Q2: What is the impact of having several learners on the accuracy of the DP-CL process?**

To observe the accuracy of our proposed DP-A-GEM method we need to include several learners in the process. In this regard, aimed at involving two learners, three various small GPB values ($\epsilon = \{0.41, 0.19, 0.12\}$ for PMNIST dataset and $\epsilon = \{2.2, 1.22, 0.5\}$ for SCIFAR dataset) are considered for different levels of noise $\sigma \in \{2, 4, 8\}$. As observed in Figure 4, the value of $\sigma$ affects the spent PB for each iteration of each learner. To perceive how these two
learners are subsequently involved, we get into one of our experiments shown in Figure 6(a). To do so, using PMNIST dataset, three different configurations of \((\epsilon, \delta)\) including \((0.41, 0.0005)\), \((0.19, 0.0005)\), and \((0.12, 0.0005)\) have been utilized. As it is witnessed, once the PB of first learner reaches to GPB at the end of task 9, the second learner comes into play to proceed the training process. From Figure 6, a sudden drop in accuracy can be observed when a new learner starts its learning process. For example, in Figure 6 (a), the accuracy of three abovementioned configurations is decreased about 40 percent. However, by

Figure 5: The accuracy of A-GEM method and the proposed DP-A-GEM method for PMNIST (a) and SCIFAR (b) datasets. Various levels of noise \(\sigma \in \{2, 4, 8\}\) have been adjusted.

Figure 6: The results of DP-CL algorithm for PMNIST and SCIFAR datasets, when two learners are used during the process, show in a and c respectively. For having two learners (a), GPB adjusted such that the first learner will finish its PB at the end of task 9. Since three levels of noise \(\sigma = \{2, 4, 8\}\) have been considered during our experiments, three values of \((\epsilon, \delta)\) adjusted such that the PB of first learner reaches to GPB after training task 6, and PB of second learner reaches to GPB after training task 12 \((0.35, 0.00001), (0.16, 0.00001), \) and \((0.115, 0.00001)\) for PMNIST (b), and \((1.8, 0.00001), (0.81, 0.00001), \) and \((0.4, 0.00001)\) for SCIFAR (d)).

Figure 7: Using FT and ES, the accuracy of DP-CL algorithm increases when two or three learners have been used. For each experiment, we investigate the effect of ES and start the training process for \(l_{1,1}\) one task earlier. At each plot, the light color shows the result without using ES \((e.g., \text{DP-A-GEM} (\sigma = 2)), \text{FT})\) and the dark color shows when ES uses along with FT \((e.g., \text{DP-A-GEM} (\sigma = 2)), \text{FT + ES})\). The DP parameters \((\epsilon, \delta)\) have been adjusted the same as previous step.
insisting the training process via the second learner, the accuracy gradually returns to the previous value. There are two main reasons for this issue. First, the noise generated by Gaussian mechanism starts to be added to the weights from the first iteration of second learner. Whilst for the first Gaussian mechanism starts to be added to the weights from the first iteration of second learner. The second learner does not exploit $D_{CV}$ for hyper-parameters fine-tuning. To circumvent this performance degradation, besides using the $D_{CV}$ for fine-tuning, we start earlier the training process for new learners, i.e., ES.

**Q3: How the ES can deal with the performance degradation in the training process?**

ES here means that the training process of the new learner commences one task earlier than the task where the spent PB reaches to the GPB. For instance, in case of having two learners, the second learner initiates its training at the beginning of task 9, whilst the first learner runs out its PB at the end of this task. During this time, both learners are involved in learning task 9, concurrently. Notably, this will be performed only during the training process where we aimed to fine-tuning the learners. Thus, during the inference time, the entire samples belonging to task 9 are fed to $l_1$.

Now, we re-execute all our experiments for two various cases. In the first one, namely FT, only $D_{CV}$ is used for fine-tuning. For the second case, called FT-ES, ES is involved, as well. The curves with transparent colors illustrated in Figure 7, correspond to the accuracies corresponding to the FT, dark colors are associated with the accuracies corresponding to the FT-ES. If we have only FT, the accuracy respectively increases 37%, 18% for PMNIST and SCIFAR datasets, as compared to their counterparts when even FT is not performed. Moreover, in case of FT-ES the accuracy respectively improves to 4%, 6% for PMNIST and SCIFAR datasets, as compared to their counterparts when only FT is performed. In addition to the accuracy, the forgetting score is evaluated for different levels of noise, when one/two learners are utilized in the process (See Figure 8).

**F. THE IMPACT OF ATTACKS ON THE ROBUST DP-CL PROCESS**

**Q4: How the proposed robust DP-CL acts against attacks?**

In this regard, we first apply the four white-box attacks mentioned in Section 3.3 on 5 different scenarios, comprised of: 1) A-GEM algorithm, 2) DP-A-GEM, 3) PixelDP-A-GEM, 4) RAGEM-PixelDP-A-GEM, and 5) EEM- RAGEM-PixelDP-A-GEM. Before that, we applied the attacks on A-GEM and DP-A-GEM algorithms. Figure 9 shows the impact of attack on A-GEM algorithm, after learning of each task, the four attack algorithms have been applied on test set and the accuracy has been measured (Figure 9, light colors). Compared to A-GEM algorithm, the DP-A-GEM algorithms obtained better accuracy by 9.3 percent and 4.6 percent for PMNIST and SCIFAR datasets respectively. Finally, by measuring forgetting average and certified accuracy metrics, we evaluated the effect of the proposed robust solutions (PixelDP, robust-A-GEM (RAGEM), and efficient-EM(EEM)) by applying the white-box attacks on two PMNIST (Figure 11 (a)) and SCIFAR (Figure 11 (b)) datasets.

**G. DATA CONSUMPTION**

**Q5: How much data the DP-CL process will need?**

The number of replaced samples in EM has been observed during the training process for both datasets, which helps to have a good estimation of number of necessary training data in DP-CL training process. Figure 11 shows the number of replaced samples for different levels of noise $\sigma = \{2, 4, 8\}$ for the two PMNIST (Figure 11 (a, b)) and SCIFAR (Figure 11 (c, d)) datasets when one or two learners have been used in the process. As it can be observed from Figure 11, when there is one learner, and $\sigma = 2$, the training process needs more training samples. Therefore, the more we add noise, the less data the DP-CL process will need.

**VI. DISCUSSION**

There are two DL networks in our experiments, a shallow with 2 hidden layers including $\cong 269,000$ trainable parameters and another one with a deeper architecture including 18 hidden layers including 11 million trainable parameters. By measuring the DP parameters, it can be observed that, the deeper the network, the more noise will be added to the network, and consequently the DP parameters increase more quickly. For instance, at the end of training Resnet18, the value of $\delta$ is more than 10 times higher than the shallow network for all levels of noise. A similar effect is observed for $\epsilon$ such that for different levels of noise $\sigma \in \{2, 4, 8\}$, its value is 5.45, 5.38, and 5 times larger than shallow network respectively. Notably, although the deeper network has 40 times more parameters than the shallow network, the DP parameters do not increase linearly with respect to number of networks’ parameters.

To increase the privacy of both networks, we raised the noise level from 2 to 8 ($\sigma \in \{2, 4, 8\}$). Although, the accuracy of both networks constantly increases for $\sigma \in \{2, 4\}$, it decreases by about 6% and 20% for FC2 and Resnet18 networks respectively, when $\sigma = 8$. Interestingly,
The number of replaced training samples have been monitored for the two PMNIST (a and b) and SCIFAR (c and d) datasets when one (a,c) or two (b,d) learners have been used in the training process.

The results of DP-A-GEM method have less fluctuations and more stable accuracy compared to simple A-GEM method specially for $\sigma \in \{2, 4\}$. In the next step, we decreased the GPB to evaluate the performance of DP-A-GEM when there are several learners. Depending on the noise level, the accuracy of second and third learner has a sudden drop between 35-45% for PMNIST dataset, and between 20-30% for SCIFAR dataset. But by using the fine-tuning and ES strategies, the performance increases about 43% and 23% for FC2 and Resnet18 networks respectively. To accurately measure the degradation, when there are several learners, we calculate the forgetting accuracy. Notably, when there are two learners, the forgetting accuracy of each learner is less than when there is one learner in the process. For instance, the forgetting accuracy for first and second learner are 0.071 and 0.103 respectively (Figure 8 (a), $\sigma = 4$) and less than 0.124 which is the forgetting score of when there is just one learner. In other words, the long training process with just one learner leads to high forgetting score and the CF will finally happen.

Furthermore, the proposed three methods to robustize the DP-CL process are effective against the applied four white-box attacks. First, we applied the attacks on simple A-GEM algorithm and DP-A-GEM algorithm to investigate the effect of adding DP against attacks. As shown in Figure 9, almost in all cases the DP-enabled version of A-GEM increases the accuracy compared to simple A-GEM which is about 7 percent for PMNIST, and 4 percent for SCIFAR on average. Then, by adding the robust methods one-after-another, the attacks have been applied. As shown in Figure 10, each of the proposed methods has positive effect on the accuracy of the DP-A-GEM algorithm under attacks. On average, PixelDP improved the accuracy by 3.3 percent for PMNIST and 3.8 percent for SCIFAR dataset. Robust-A-GEM, which is applied after adding PixelDP method, improved the accuracy by 1.65 for PMNIST, and 4.1 percent for SCIFAR dataset.
dataset. Finally, the efficient-EM increased the accuracy by 3.6 and 2.3 percent for PMNIST and SCIFAR datasets respectively. Therefore, the robustness methods increased the accuracy of DP-A-GEM algorithms by 8.55 and 10.2 percent for PMNIST and SCIFAR datasets respectively. Moreover, by adding the robustness methods, the forgetting average decreased from 0.124 to 0.075 for PMNIST and from 0.155 to 0.123 for SCIFAR dataset.

VII. CONCLUSION
The major contribution of this paper is adding differential privacy (DP) into continual learning (CL) procedures, aimed at protecting against adversarial examples. In CL processes, the model learns sequentially and endlessly from time-varying data streams which makes the task of adding DP to CL challenging. More explicitly, the added noise due to DP together with the endless learning feature of CL leads to CF which is a serious obstacle. To address this concern, we have proposed an innovative approach by which we cannot only strike a tradeoff between privacy and utility, but also mitigate the CF. We continually control the instantaneous spent PB to not exceed the available GPB. Besides, a three-step robust procedure is also included in our approach to mitigate the negative impact of CF, as much as possible. We also assessed the proposed approach against four well-recognized adversarial attacks comprised of: 1) FGSM, 2) I-FGSM, 3) MIM, and 4) the attack by Madry et al. [31]. Our simulation results validated the effectiveness of the proposed method in facing such strong attacks so that we could improve the criteria of both the certified accuracy and the forgetting measure, simultaneously.

REFERENCES
[1] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521, no. 7553 (2015): 436-444.
[2] Su, Chang, Zhenxing Xu, Jyotishman Pathak, and Fei Wang. "Deep learning in mental health outcome research: a scoping review." Translational Psychiatry 10, no. 1 (2020): 1-26.
[3] Károly, Artür István, Péter Galambos, József Kuti, and Imre J. Rudas. "Deep learning in robotics: Survey on model structures and training strategies." IEEE Transactions on Systems, Man, and Cybernetics: Systems 51, no. 1 (2020): 266-279.
[4] Lavecchia, Antonio. "Deep learning in drug discovery: opportunities, challenges and future prospects." Drug discovery today 24, no. 10 (2019): 2017-2032.
[5] Gomez, Luis F., Aythami Morales, Juan R. Orozco-Arroyave, Roberto Daza, and Julian Fierrez. "Improving Parkinson Detection Using Dynamic Features From Evoked Expressions in Video." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1562-1570. 2021.
[6] Delange, Matthias, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Timme Tuytelaars. "A continual learning survey: Defying forgetting in classification tasks." IEEE Transactions on Pattern Analysis and Machine Intelligence (2021).
[7] Akhtar, Naveed, and Ajmal Mian. "Threat of adversarial attacks on deep learning in computer vision: A survey." IEEE Access 6 (2018): 14410-14430.
[8] Huang, Xiaowei, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. "A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defense, and interpretability." Computer Science Review 37 (2020): 100270.
[9] Younsuf, Hana, Michael Lahzi, Said A. Salloum, and Khaled Shaalan. "Systematic review on fully homomorphic encryption scheme and its application." Recent Advances in Intelligent Systems and Smart Applications (2021): 537-551.
[10] Li, Jing, Xiaohui Kuang, Shujie Lin, Xu Ma, and Yi Tang. "Privacy preservation for machine learning training and classification based on homomorphic encryption schemes." Information Sciences 526 (2020): 166-179.
[11] Sweeney, Latanya. "k-anonymity: A model for protecting privacy." International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10, no. 05 (2002): 557-570.
[12] Agarwal, Charu C. "On k-anonymity and the curse of dimensionality." In FLDB, vol. 5, pp. 901-909. 2005.
[13] Brickell, Justin, and Vitaly Shmatikov. "The cost of privacy: destruction of data-mining utility in anonymized data publishing." In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 70-78. 2008.
[14] Dwork, Cynthia, and Aaron Roth. "The algorithmic foundations of differential privacy." Found. Trends Theor. Comput. Sci. 9, no. 3-4 (2014): 211-407.
[15] Abadi, Martin, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep learning with differential privacy." In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp. 308-318. 2016.
[16] Zhao, Jingwen, Yunfang Chen, and Wei Zhang. "Differential privacy preservation in deep learning: Challenges, opportunities and solutions." IEEE Access 7 (2019): 48901-48911.
[17] Chen, Yan, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. "Pegasus: Data-adaptive differentially private stream processing." In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp. 1375-1388. 2017.
[18] Dwork, C., Naor, M., Pitassi, T. and Rothblum, G.N., 2010. June. Differential privacy under continual observation. In Proceedings of the forty-second ACM symposium on Theory of computing (pp. 715-724).
[19] Bu, Zhiqi, Jinshuo Dong, Qi Long, and Weijie J. Su. "Deep learning with Gaussian differential privacy." Harvard data science review data science review 2020, no. 23 (2020).
[20] Arachchige, Pathum Chamikara Mahawaga, Peter Bertok, Ibrahim Khalil, Dongxi Liu, Seyit Camtepe, and Mohammed Atiquzzaman. "Local differential privacy for deep learning." IEEE Internet of Things Journal 7, no. 7 (2019): 5827-5842.
[21] Phan, NhatHai, Xintao Wu, Han Hu, and Dejing Dou. "Adaptive Laplace mechanism: Differential privacy preservation in deep learning." In 2017 IEEE International Conference on Data Mining (ICDM), pp. 385-394. IEEE, 2017.
[22] Lécuyer, Matthias, Riley Spahn, Kiran Vodrahalli, Roxana Geambasu, and Daniel Hsu. "Privacy accounting and quality control in the sage differentially private ML platform." In Proceedings of the 27th ACM Symposium on Operating Systems Principles, pp. 181-195. 2019.
[23] Cummings, Rachel, Sara Kreibiel, Kevin A. Lai, and Uthaipton Tantipongpipat. "Differential privacy for growing databases." arXiv preprint arXiv:1803.06416 (2018).
[24] Wei, Kang, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H. Vincent Poor. "Federated learning with differential privacy: Algorithms and performance analysis." IEEE Transactions on Information Forensics and Security 15 (2020): 3454-3469.
[25] Chaudhry, Arslan, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. "Efficient lifelong learning with a-gem." arXiv preprint arXiv:1812.00420 (2018).
[26] Shokri, Reza, and Vitaly Shmatikov. "Privacy-preserving deep learning." In Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, pp. 1310-1321. 2015.
[27] Dwork, Cynthia, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating noise to sensitivity in private data analysis." In Theory of cryptography conference, pp. 265-284. Springer, Berlin, Heidelberg, 2006.
[28] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv preprint arXiv:1412.6572 (2014).
[29] Dong, Yinpeng, Fangzhao Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. "Boosting adversarial attacks with momentum."
In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 9185-9193. 2018.
[31] Madry, Aleksander, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. "Towards deep learning models resistant to adversarial attacks." arXiv preprint arXiv:1706.06083 (2017).
[32] Kurakin, Alexey, Ian Goodfellow, and Samy Bengio. "Adversarial examples in the physical world." (2016).
[33] Lecuyer, Mathias, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. "Certified robustness to adversarial examples with differential privacy." In 2019 IEEE Symposium on Security and Privacy (SP), pp. 656-672. IEEE, 2019.
[34] Kirkpatrick, James, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan et al. "Overcoming catastrophic forgetting in neural networks." Proceedings of the national academy of sciences 114, no. 13 (2017): 3521-3526.
[35] Zenke, Friedemann, Ben Poole, and Surya Ganguli. "Continual learning through synaptic intelligence." In International Conference on Machine Learning, pp. 3987-3995. PMLR, 2017.
[36] Krizhevsky, Alex, and Geoffrey Hinton. "Learning multiple layers of features from tiny images." (2009): 7.
[37] Lopez-Paz, David, and Marc'Aurelio Ranzato. "Gradient episodic memory for continual learning." Advances in neural information processing systems 30 (2017): 6467-6476.
[38] Chaudry, Arslan, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. "Riemannian walk for incremental learning: Understanding forgetting and intransigence." In Proceedings of the European Conference on Computer Vision (ECCV), pp. 532-547. 2018.
[39] Gomez-Barrero, Marta, Emanuele Maiorana, Javier Galbally, Patrizio Campisi, and Julian Fierrez. "Multi-biometric template protection based on homomorphic encryption." Pattern Recognition 67 (2017): 149-163.
[40] Jiang, Bin, Jianqiang Li, Guanghui Yue, and Houbing Song. "Differential privacy for industrial internet of things: opportunities, applications and challenges." IEEE Internet of Things Journal (2021).
[41] Zhao, Jingwen, Yunfang Chen, and Wei Zhang. "Differential privacy preservation in deep learning: Challenges, opportunities and solutions." IEEE Access 7 (2019): 48901-48911.

Ahmad Hassanpour is currently a Ph.D candidate at the Department of Information Security and Communication Technology (IIT), Norwegian University of Science and Technology (NTNU), Norway. He is also a Marie Skłodowska-Curie fellow and part of an EU project called Privacy Matters. Previously, he received the M.S. degree in computer engineering from Shiraz University of Technology. His research areas are deep learning, computer vision, and privacy.

Majid Moradkia was born in 1986. He received the Ph.D. degree in telecommunication system engineering from the Department of Electrical and Electronics Engineering, Shiraz University of Technology, Shiraz, Iran. He currently works as a Postdoctoral Research Fellow with the Engineering Technology Department, University of Houston, TX, USA. His main research interests lie within the area of physical-layer security, Internet of Things (IoT), Privacy, Machine Imaging, and Signal Processing.

Bian Yang got his B.S. (2000), M.S.(2002), and Ph.D (2006) from Harbin Institute of Technology in electronic engineering and information security. He visited Fraunhofer IGD in Darmstadt during 2003-2005 for research on data hiding and media security, worked as a lecture at the School of Computer Science and Technology at Harbin Institute of Technology during 2005-2007, and as a research engineer at Thomson Corporate Research in Beijing during 2007-2008. He joined the Norwegian Information Security Laboratory (NISlab) at the Gjøvik University College working on privacy-preserving biometrics during 2008-2015, and founded in 2016 and has since been coordinating the eHealth and Welfare Security (eHWS) group at the Department of Information Security and Communication Technology of the Norwegian University of Science and Technology (NTNU). His research interests include cybersecurity and privacy for e-health and welfare technologies and services, privacy modelling and enhancing technologies, security biometrics and identity management, and security practice and human factors.

Ahmed Abdelhadi (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from The University of Texas at Austin in 2011. He is currently an Assistant Professor with the University of Houston. Before joining UH, he was a Research Assistant Professor with Virginia Tech. He was a member of the Wireless Networking and Communications Group (WNCG) and the Laboratory of Informatics, Networks and Communications (LINC) Group during his Ph.D. In 2012, he joined the Bradley Department of Electrical and Computer Engineering and the Hume Center for National Security and Technology, Virginia Tech. He was a Faculty Member of Wireless @ Virginia Tech. His research interests are in the areas of wireless communications and networks, artificial intelligence, cyber physical systems, and security. He has coauthored more than 80 journal and conference papers, and seven books in these research topics. His book Cellular Communications Systems in Congested Environments is bookplated as the Virginia Tech Provost’s Honor Book and his book Resource Allocation with Carrier Aggregation in Cellular Networks is featured in the 13th Annual Virginia Tech Authors Recognition Event. He received the Silver Contribution Award from the IEEE International Conference on Computing, Networking and Communications (ICNC), the Best Paper Award from the IEEE International Symposium on Systems Engineering (ISSE), and the Outstanding Paper Award from the IEEE International Conference of Advanced Communications Technology (ICACT).

Christoph Busch (Senior Member, IEEE) is currently a member of the Department of Information Security and Communication Technology (IIT), Norwegian University of Science and Technology (NTNU), Norway. He also holds a joint appointment with the Faculty of Computer Science, Hochschule Darmstadt (HDA), Germany. Furthermore, he has been a Lecturer of biometric systems with the Technical University of Denmark (DTU), since 2007. He has coauthored more than 400 technical articles and has been a speaker at international conferences. He is also a Convener of WG3 in ISO/IEC JTC1 SC37 on biometrics and an Active Member of CEN TC 224 WG18. Furthermore, on behalf of Fraunhofer, he chairs the Biometrics Working Group of the TeleTrusT Association and the German Standardization Body on Biometrics (DIN-NIA37). He served for various program committees, such as NIST IBPC, ICB, ICHB, BSI-Congress, GI-Congress, DACH, WEDELMUSIC, and EUROGRAPHICS, and served for several conferences, journals, and magazines as a Reviewer, such as ACM-SIGGRAPH, ACM-TISSEC, the IEEE COMPUTER GRAPHICS AND APPLICATIONS, the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, and the Computers & Security journal (Elsevier). He is also an Appointed Member of the Editorial Board of the IET Biometrics journal and the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY journal.
Julian Fierrez (Member, IEEE) received the M.Sc. and Ph.D. degrees in telecommunications engineering from the Universidad Politécnica de Madrid, Spain, in 2001 and 2006, respectively. Since 2004 he has been at Universidad Autonoma de Madrid, where he is currently an Associate Professor. From 2007 to 2009, he was a Visiting Researcher with Michigan State University, USA, under a Marie Curie postdoc. His research is on signal and image processing, HCI, responsible AI, and biometrics for security and human behavior analysis. He is actively involved in large EU projects in these topics (e.g., TABULA RASA and BEAT in the past, now IDEA-FAST and TRESPASS-ETN), and has attracted notable impact for his research. He was a recipient of a number of distinctions, including the EAB Industry Award 2006, the EURASIP Best Ph.D. Award 2012, and the 2017 IAPR Young Biometrics Investigator Award. He has received best paper awards at ICB and ICPR. He is Associate Editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and the IEEE TRANSACTIONS ON IMAGE PROCESSING. He is member of the ELLIS Society

APPENDIX A: MODIFYING A-GEM UPDATE RULE (EQ. (5))

Here we provide the proof DP–A-GEM’ update rule, stated in Section IV (C2) , Eq. 5. To proof, we first invoke the DP–A-GEM problem in Eq. 4 as follows:

\[
\min_{\bar{g}_j} \frac{1}{2} \| g - \bar{g}_j \|_2^2 \\
\text{s.t. } \langle \bar{g}_j, g_{jk} \rangle \geq 0 \ \forall k < t
\]  

(A.1)

Replacing \( \bar{g}_j \) with \( z \) and rewriting Eq. A.1 yields:

\[
\min_{z} \frac{1}{2} z^T z - g^T z \text{ s.t. } -z^T g_{j,ref} \leq 0
\]  

(A.2)

Note that we removed the term \( g^T > g \) from the OF and change the direction of the inequality constraint. The Lagrangian function can be acquired as:

\[
L(z, \alpha) = \frac{1}{2} z^T z - g^T z - \alpha z^T g_{j,ref}
\]  

(A.3)

Now, we pose the dual of Eq. A.3 as:

\[
\theta_{D}(\alpha) = \min_{z} L(z, \alpha)
\]  

(A.4)

Let's find the value \( z^* \) that minimizes the \( L(z, \alpha) \) by setting the derivatives of \( L(z, \alpha) \) w.r.t. to \( z \) to zero:

\[
\nabla_z L(z, \alpha) = 0
\]  

(A.5)

The simplified dual after putting the value of \( z^* \) in Eq. A.4 can be written as:

\[
\theta_{D}(\alpha) = \frac{1}{2} \left( g^T g + 2\alpha g^T g_{j,ref} + \alpha^2 g_{j,ref}^T g_{j,ref} \right) - g^T g - 2\alpha g^T g_{j,ref} - \alpha^2 g_{j,ref}^T g_{j,ref} = -\frac{1}{2} g^T g - \alpha g^T g_{j,ref} - \frac{1}{2} \alpha^2 g_{j,ref}^T g_{j,ref}
\]  

(A.6)

This solution \( \alpha^* \) = max_{\alpha, \alpha > 0} \theta_{D}(\alpha) to dual is given by:

\[
\nabla_{\alpha} \theta_{D}(\alpha) = 0
\]  

(A.7)

By putting \( \alpha^* \) in Eq. A.5, we recover the A - GEM update rule:

\[
z^* = g - \frac{g^T g_{j,ref}}{\theta_{j,ref}^T g_{j,ref}} g_{j,ref} = \bar{g}
\]  

(A.8)