IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 15, 2021, accepted February 21, 2022, date of publication February 25, 2022, date of current version March 8, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3154826

Differential Privacy Preservation in Robust Continual Learning

AHMAD HASSANPOUR[1], MAJID MORADIKIA[2], BIAN YANG[1], AHMED ABDELHADI[2], CHRISTOPH BUSCH[1], AND JULIAN FIERREZ[3], (Member, IEEE)

[1]Department of Information Security and Communication Technology, Norwegian University of Science and Technology (NTNU), 2815 Gjøvik, Norway
[2]Engineering Technology Department, University of Houston, Houston, TX 77004, USA
[3]School of Engineering, Universidad Autonoma de Madrid, 28049 Madrid, Spain

Corresponding author: Ahmad Hassanpour (ahmad.hassanpour@ntnu.no)

This work was supported by the Project Privacy Matters (PRIMA) under Grant H2020-MSCA-ITN-2019-860315. The work of Julian Fierrez was supported by the Project Biometrics and Behavior for Unbiased and Trusted AI with Applications (BBforTAI) under Grant PID2021-127641OB-I00 MICINN/FEDER.

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 (*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 completely access to the architecture and weights of the model, comprised of countermeasure methods.

The associate editor coordinating the review of this manuscript and approving it for publication was Aasia Khanum.

Gray-box attacks also presume that the attacker knows everything about the network and defense, except the parameters.

To confront with such attacks, three well known methods have been broadly used in several literature: fully homomorphic encryption (HE) [10], [11], [37], 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 data set 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 [17]. 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], [38]. 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. Dwork and Roth [15] 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] 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) [30], 3) Momentum Iterative Method (MIM) [28], and 4) the attack proposed by Madry *et al.* [29]. 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], [19]–[21]. 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 [22], [23] 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 [23]. 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 [24]. 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 [22], have proposed the so-called block composition theorem by which the DL algorithms are executed till the PB consumption of each block [1] 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.**

[1]We can interchangeably use the word of "block" and "sample" throughout the paper.

III. PRELIMINARIES

A. CONTINUAL LEARNING

A typical CL process, e.g., A-GEM [25], 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 $x_i \in X_t$ is a feature vector, $t_i \in T$ is a task descriptor, and $y_i \in Y$ is a target vector. In general, CL algorithms aim to learn a predictor $f_\theta : X \times T \rightarrow Y$ in which θ denotes the relevant tunable parameters of predictor f.

To get more insight, in the following we succinctly explain A-GEM [25]. 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., $k \leq T$, the abovementioned explanations can be mathematically formulated as the following constrained optimization problem

$$\min_{\theta} \mathcal{L}_{AG}\left(f_\theta, D_t\right), s.t. \mathcal{L}_{AG}\left(f_\theta, M_k\right) \leq \mathcal{L}_{AG}\left(f_\theta^{t-1}, M_k\right) \forall k < t \tag{1}$$

where the objective function $\mathcal{L}_{AG}\left(f_\theta, D_t\right)$ 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 $A : B \rightarrow R$. The DP is often characterized by the parameters (ϵ, δ) where the privacy budget (PB) $\epsilon > 0$ and the broken probability $\delta \in [0, 1]$ 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:

$$P\left[A\left(D\right) \in O\right] \leq e^{\epsilon} P\left[A\left(D^{'}\right) \in O\right] + \delta \tag{2}$$

where $\left\{D, D^{'}\right\} \in B$ are two neighboring inputs and $O \subseteq R$ represents any subsets of outputs. Besides, $P\left[\cdot\right]$ denotes the 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:

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

2) **Gaussian mechanism** [15]: This mechanism will be used in this paper. Using this kind of mechanism the white Gaussian noise $\mathcal{N}\left(0, \sigma^2\right)$ is added to the output entries. Given $\epsilon \in (0, 1]$, the Gaussian mechanism with $\sigma \geq \sqrt{2ln\left(\frac{1.25}{\delta}\right)}.\frac{\Delta_A}{\epsilon}$ is (ϵ, δ) –DP and the l_2 sensitivity parameter Δ_A 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" [26], considering ℓ subroutines each of which is $(\epsilon, 0)$-differentially private, the privacy of an algorithm including a combination of these subroutines is degraded up to the bound of $(\epsilon\ell, 0)$ as compared to the single subroutine.

C. ADVERSERIAL EXAMPLES

Adversarial examples are a kind of attack against ML models, where the attacker add a small perturbation $\alpha \triangleq \{a_i\}_{i=1}^{I} \in R^I$ to the given input $x \triangleq \{x_i\}_{i=1}^{I} \in R^I$ of the DL model, resulting in a considerable change at the output $y \triangleq \{y_i\}_{i=1}^{c} \in R^c$. The perturbation is usually specified by a l_p-norm ball of radius μ, i.e., $\Theta_\mu \triangleq \left\{\alpha : \|\alpha\|_p \leq \mu\right\}$ where $p \in \{1, 2, \infty\}$ [27]. To evaluate the robustness of the proposed method, particularly four well-known white box[2] attack algorithms are utilized to generate the adversarial samples: i) Fast Gradient Sign Method (FGSM) [27], ii) Iterative-FGSM (I-FGSM) [30], iii) Momentum Iterative Method (MIM) [28], and iv) the attack proposed by Madry *et al.*, [29], 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 the 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 effect 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 (ϵ, δ). As per requirements of a CL process these variables need to be updated and thus a CL-based composition theory must satisfy the three 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 [22] or [23]. 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 $L \triangleq \{l_1, l_2, \ldots\}$, 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 $M = \left\{b_i, b_j, \ldots\right\}$, where b_i shows the ith 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 (l_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., $L \triangleq \{l_1, l_2, \ldots, l_{c-1}, l_c\}$ 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* (ϵ_g, δ_g) : More explicitly, 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 GPB 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 θ_{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., $(PB_r \triangleq GPB - PB_{l_c})$ 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 $PB_r > PB_{c+1}$. 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., $PB_c = PB_{c+1}$). 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 [22] or [23] 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_{b_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 $PB_{b_i} = \{PB_{m_i,l_k}, \ldots, PB_{m_j,l_h}\}$, where PB_{m_i,l_k} stands for the consumed PB of ith iteration of the learner l_k. By doing so, we can calculate the total consumed PB for the sample via feeding PB_{b_i} to the Block Composition Theorem (BCT) [22]. Tracking the behavior of PB_{b_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 (σ) 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 (ϵ_g, δ_g)-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.

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

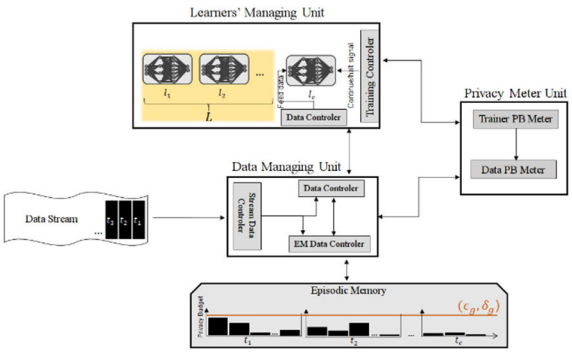

FIGURE 1. The proposed DP-CL architecture.

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 (ϵ, δ)-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 [31]. More explicitly, in accordance with the sensitivity (Δ) and size of the first layer ($|h_1|$), a noise with zero mean and standard deviation $\sigma = \sqrt{2\ln(\frac{1.25}{\delta})}\Delta_{p,2}L/\epsilon$ is produced by Gaussian mechanism (*noise*$(\Delta, L, \epsilon, \delta)$, *line 6, Algorithm 2*).

Algorithm 1 Proposed DP-CL.

Procedure DP-CL:
// Learners' Managing Unit (LMU):
While $b_{i,t_i} \leftarrow$ ask data from *DMU*
  If $l_c = \emptyset$ **then**
    Initialize l_c
  train l_c with b_{i,t_i}
  $SPB_{l_c} \leftarrow PMU\,(l_c, b_{i,t_i})$
  $DMU\,(b_{i,t_i}, SPB_{l_c})$// *save part of* b_{i,t_i} *in EM*
  If$SPB_{l_c} \geq GPB$ **then**
    $C \leftarrow [l_c, (t_j, \ldots, t_i)]$
    $l_c \leftarrow$ Initialize a new learner
// Privacy Meter Unit (PMU):
If $l_c, b_{i,t_i} \leftarrow$ receive request from *LMU* **then**
  LMU $\leftarrow$ Calculate the spent PB of l_c using MAA
  DMU $\leftarrow$ Calculate the spent PB of each training sample in b_{i,t_i} using BCT
//Data Managing Unit (DMU):
If receive data from *Stream* **then**
  Save the data temporary
If receive request from *LMU* **then**
  Fetch a batch of data from EM/Stream and send to *LMU*
If receive data from *LMU* **then**
  Store data in EM
  Store spent PB for each b_i
  Update EM by replacing those samples which run out their PB with ZPB samples (*Procedure UpdateEpsMem, Alg.2*)
end procedure

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., $\mathbb{E}_t\left(A_j\left(x\right)\right) - \mathbb{E}_{t-1}\left(A_j\left(x\right)\right) \geq 0$ where $\mathbb{E}_t\left(A_j\left(x\right)\right) = \frac{1}{n}\sum_n a_{j,n}(x)$ 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 ($\tilde{g}_j$) and the gradient of l_c for the previous tasks ($\forall k < t$) for label j ($g_{j,k}$) should be greater than zero ($\langle \tilde{g}_j, g_{j,k} \rangle \geq 0$).

Moreover, instead of n times invoking $A(x)$ for a specific sample x, to calculate $\mathbb{E}_t\left(A_j\left(x\right)\right)$, we use n samples belonging to the j^{th} class within the current batch, and for calculating $\mathbb{E}_{t-1}\left(A_j\left(x\right)\right)$, 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 A-GEM objective function. Therefore, we modify the optimization function as below:

$$\min_{\tilde{g}} \frac{1}{2} \parallel g - \tilde{g}_j \parallel_2^2 \; s.t. \langle \tilde{g}_j, g_{j,k} \rangle \geq 0 \forall \quad k < t \tag{4}$$

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

$$\tilde{g} \leftarrow g - \frac{g^\top g_{j,ref}}{g_{j,ref}^\top g_{j,ref}} g_{j,ref} \tag{5}$$

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 that if a sample meets this condition, then it will be added to the EM (called efficient-EM). For sample x_z located in a batch including n samples, the robustness condition calculated as follow:

$$\begin{aligned} &\mathbb{E}_t^{lb}\left(f_j\left(x_z\right)\right) - max_{i:i\neq j}\mathbb{E}_t^{ub}\left(f_j\left(x_z\right)\right) \\ &\geq \frac{1}{1+e^{\frac{\sum_{s=1}^{n}\mathbb{E}^{lb}\left(f_j(x_s)\right)-max_{i:i\neq j}\mathbb{E}^{ub}\left(f_j(x_s)\right)}{n}}} \end{aligned} \tag{6}$$

$\mathbb{E}_t^{lb}\left(f_j\left(x_z\right)\right)$ and $\mathbb{E}_t^{ub}\left(f_j\left(x_z\right)\right)$ are the η-confidence lower and upper bound, respectively. We estimate these bounds using Hoeffding's inequality with probability η, $\mathbb{E}^{lb}\left(f\left(x\right)\right) \triangleq \mathbb{E}\left(f\left(x\right)\right) - \sqrt{\frac{1}{2n} ln\left(\frac{2y}{1-\eta}\right)} \leq E\left(f\left(x\right)\right) \leq \mathbb{E}\left(f\left(x\right)\right) + \sqrt{\frac{1}{2n} ln\left(\frac{2y}{1-\eta}\right)} \triangleq \mathbb{E}^{ub}\left(f\left(x\right)\right)$ 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 $\mathcal{C}$ and subsequently, a random Gaussian noise $N\left(0,\sigma^2\mathcal{C}^2I\right)$ with

Algorithm 2 Robust DP-CL.

Procedure Train(f_θ, D^{train}, D^{test})
1 **Input**: Datasets D^{train} and D^{test}, batch size m, learning rate for each task ζ_t, gradient norm band C, privacy budget ϵ, broken probability δ, robustness parameters:σ_r, ϵ_r, δ_r, Δ_r, size of first hidden layer $|h_1|, f$ includes z hidden layers $\{h_1, \ldots, h_z\}$, EM depicted by M.
2 **Initialize** θ randomly
3 $\gamma \leftarrow N(0, \sigma^2 |h_1|)$
4 **for** $t = \{1, \ldots, T\}$ **do**
5 **for** $(x, y) \in D_t^{train}$ **do**
6 $h_1 \leftarrow W_1^T x + \gamma$
7 $(x_{ref}, y_{ref}) \sim M(y)$
8 $g_{ref} \leftarrow \nabla_\theta l(f_\theta\left(x_{ref}, t\right), y_{ref})$
9 $g \leftarrow \nabla_\theta l(f_\theta\left(x, t\right), y)$
10 **Clipping gradient and adding noise**
11 $g'_{ref} \leftarrow \frac{1}{m}(\frac{g_{ref}}{max\left(1, \frac{\|g_{ref}\|}{C}\right)} + N\left(0, \sigma^2 C^2 I\right))$
12 $g' \leftarrow \frac{1}{m}(\frac{g_{ref}}{max\left(1, \frac{\|g\|}{C}\right)} + N\left(0, \sigma^2 C^2 I\right))$
13 **If** $g^{'} g'_{ref} \geq 0$ **then**
14 $\tilde{g} \leftarrow g'$
15 **else**
16 $\tilde{g} \leftarrow g' - \frac{g'^\top g'_{j,ref}}{g'^\top_{j,ref} g'_{j,ref}} g_{j,ref}$
17 **end if**
18 $\theta \leftarrow \theta - \zeta_t \tilde{g}$
19 **end for**
20 UpdateEpsMem (M, D_t^{train}, T)
21 **end for**
end procedure

Procedure UpdateEpsMem (M, D_t^{train}, T, GPB)
22 // remove stored samples in $|M|$ with high PB
23 **for** $i = \{1, \ldots, |M|\}$ **do**
24 **if** $spent\_PB_i > GPB$ **do**
25 remove $(x, y_i, t_i) \leftarrow M_i$
26 $(x) \leftarrow D_{y_i,t_i}^{train}$ 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 $(x, y) \leftarrow D_t^{train}$
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_θ, D^{test})
37 $a \leftarrow 0 \in R^T$
38 **for** $t = \{1, \ldots, T$ **do**
39 $a_t \leftarrow 0$
40 **for** $(x, y) \in D_t^{test}$ **do**
41 $a_t \leftarrow a_t + ACCURACY(f_\theta\left(x, t\right), y)$
42 **end for**
43 $a_t \leftarrow \frac{a_t}{len(D_t^{test})}$
44 **end for**
45 **return** a
end procedure

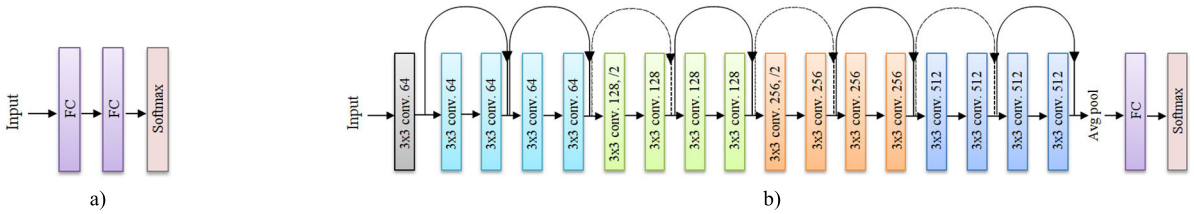

FIGURE 2. a) Fully connected network with two hidden layers used for PMNIST dataset. b) Reduced ResNet18 for SCIFAR dataset.

a predefined noise scale σ 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 $D^{train}_{y_i,t_i}$. 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, Permuted MNIST (PMNIST) [32] 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) [33] devides of dividing the original CIFAR-100 dataset [34] 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 $D^{CV} = \{D_1, \ldots, D_{T_{cv}}\}$ and $D^{EV} = \{D_{T_{cv}+1}, \ldots, D_T\}$ where $D_k = \{(x^k_i, t^k_i, y^k_i)^{n_k}_{i=1}\}$ is the dataset of kth 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 [35], 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 [25], 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 [36], Average Forgetting [36], and Certified Accuracy [31] to evaluate our proposed robust DP-CL model. In the following we briefly define these metrics. The training dataset of each task, D_t, consists of a total B_t mini-batches. After each observation of B_t, 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 ith mini-batch of task t.

Average Accuracy [36], 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: $AA_t = \frac{1}{k}\sum_{j=1}^{t} a_{k,B_k,j}$.

Average Forgetting [36], varying between $[-1, 1]$, is computed after the model has been trained for the tasks $1, 2, \ldots, t-1$. This metric is defined as $F_k = \frac{1}{k}\sum_{j=1}^{t-1} f^t_j$ where f^t_j is the forgetting measure on task j after the model is trained for the tasks $1, 2, \ldots, t-1$, obtained as:

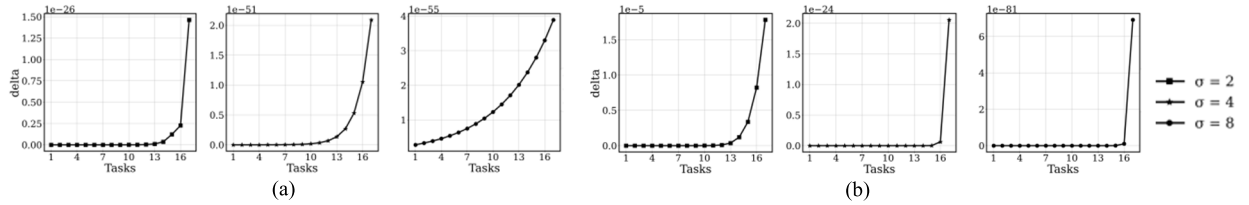

FIGURE 3. Behavior of δ vs. different tasks for $\epsilon = 2$, as well as different level of noise $\sigma \in \{2, 4, 8\}$. (a) PMNIST dataset, (b) SCIFAR dataset.

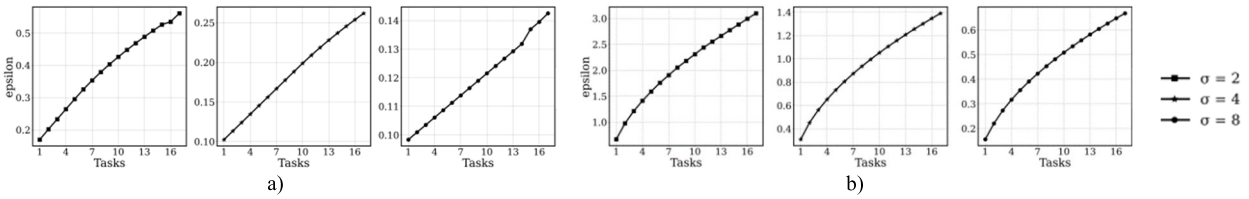

FIGURE 4. Behavior of ϵ vs. different tasks for $\delta = \frac{1}{10000}$, as well as different level of noise $\sigma \in \{2, 4, 8\}$. (a) PMNIST dataset, (b) SCIFAR dataset

$f_j^t = max_{l\in\{1,\ldots,k-1\}} a_{l,B_l,j} - a_{k,B_t,j}$. 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 [31],varying between [0, 1] is defined as $CF \triangleq \frac{\sum_{i=1}^{|test|} is\ Correct(x_i) \& is\ Robust(x_i)}{|test|}$ where $|test|$ is the size of testing set and $isCorrect\ (x_i)$ denotes a function returning 1 if the prediction on test sample x_i returns the correct label, and 0 otherwise, and $isRobust(x_i)$ returns 1 if the robustness size is larger than a given attack bound μ_a and 0 otherwise.

D. BEHAVIOR ANALYSIS OF DP's PARAMETERS

In this section, we aim for observing the behavior of DP parameters (ϵ, δ) 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 δ is calculated for a given $\epsilon = 2$, while $\sigma \in \{2, 4, 8\}$, and $t \in \{1 \ldots 17\}$. As it can be seen from Figure 3 (a), for the cases $\sigma = 2$ as well as $\sigma = 4$ the value of δ smoothly increases during the first tasks, while sharply grows for the 4 last tasks. While, in case of $\sigma = 8$ δ 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 δ would be resulted.

Figure 4 depicts six other plots where ϵ is calculated for a given $\delta = \frac{1}{10000}$, while σ, 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 ϵ 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 ϵ =2 for PMNIST while for SCIFAR we have ϵ =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 $\sigma = \{2, 4, 8\}$. As observed in Figure 4, the value of σ 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 (ϵ, δ) 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 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

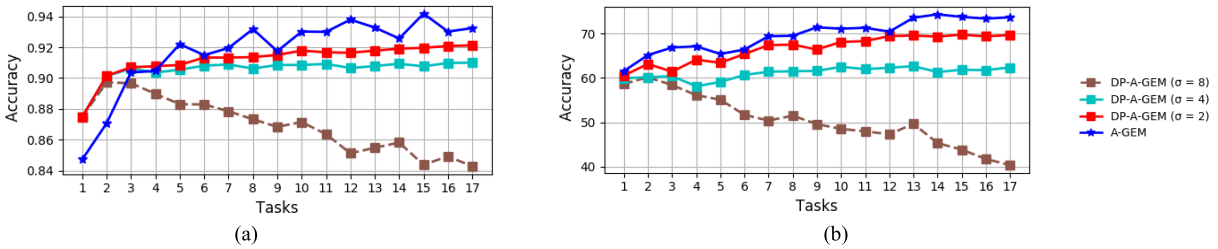

FIGURE 5. The accuracy of A-GEM method and the proposed DP-A-GEM method for PMNIST (a) and SCIFAR (b) datasets. Various levels of noise $\sigma \in \{2, 4, 8\}$ have been adjusted.

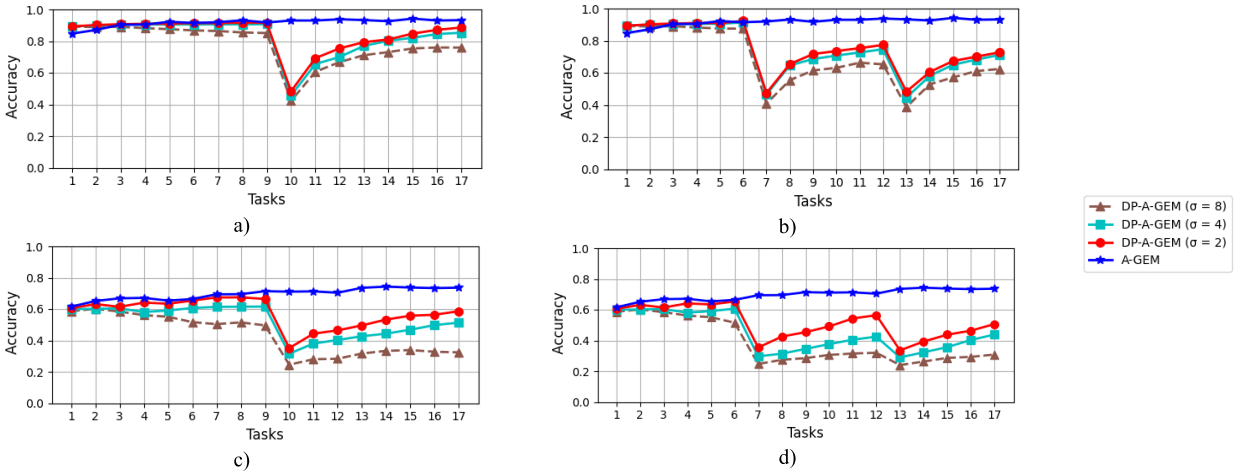

FIGURE 6. The results of DP-CL algorithm for PMNIST and SCIFAR datasets, when two learners are used during the process, show in a and c respectively. For having two learners (a), GPB adjusted such that the first learner will finish its PB at the end of task 9. Since three levels of noise $\sigma = \{2, 4, 8\}$ have been considered during our experiments, three values of (ϵ, δ) including (0.41, 0.00001), (0.19, 0. 00001), and (0.12, 0. 00001) used for PMNIST dataset, and (2.2, 0. 00001), (1.22, 0. 00001), and (0.5, 0. 00001) used for SCIFAR dataset. Moreover, for having three learners, the GPB values (ϵ, δ) 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)).

by Gaussian mechanism starts to be added to the weights from the first iteration of second learner. Whilst for the first learner this noise is added after the fine-tuning step, mitigating the impact of noise on accuracy. Second, the second learner does not exploit D^{CV} for hyper-parameters' fine-tuning. To circumvent this performance degradation, besides of 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

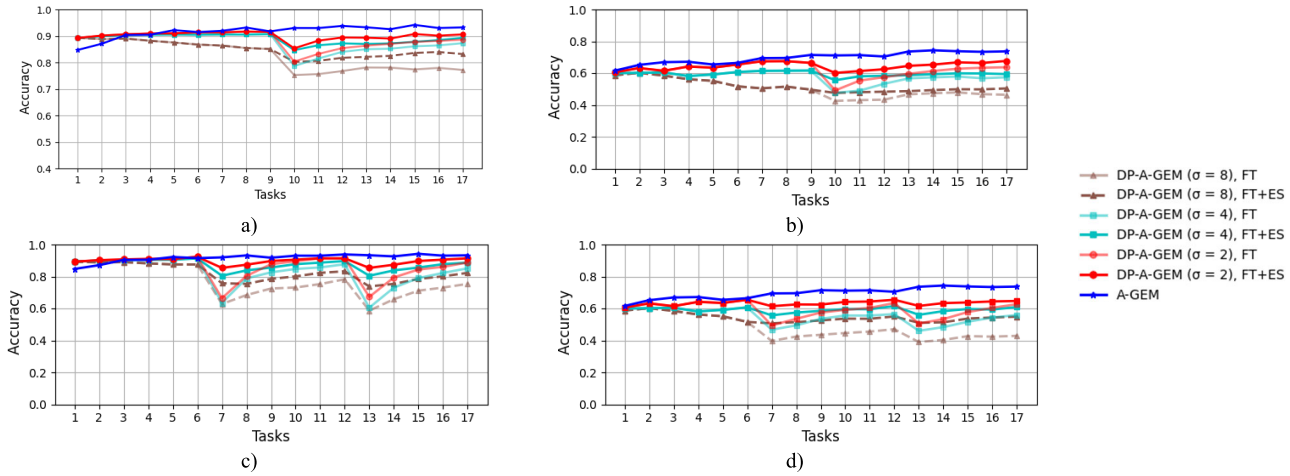

FIGURE 7. Using FT and ES, the accuracy of DP-CL algorithm increases when two or three learners have been used. For each experiment, we investigate the effect of ES and start the training process for l_{c+1} one task earlier. At each plot, the light color shows the result without using ES (e.g., $DP-A-GEM\sigma=2$), FT.) and the dark color shows when ES uses along with FT (e.g., $DP-A-GEM(\sigma=2)$, $FT+ES$). The DP parameters $((\epsilon,\delta))$ have been adjusted the same as previous step.

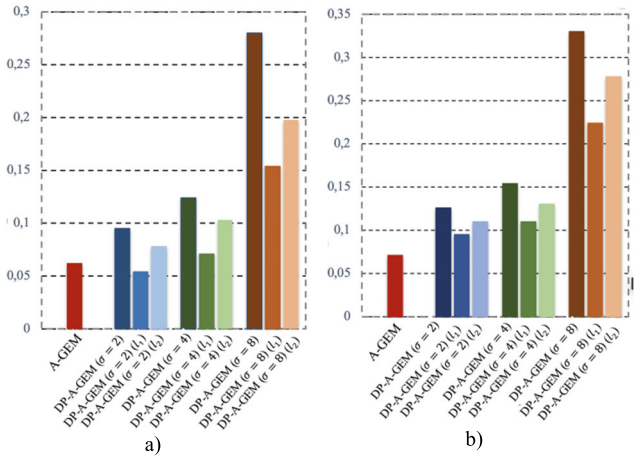

FIGURE 8. The forgetting average has been calculated for two PMNIST (a), and SCIFAR (b) datasets. For each one, when there are one or two learners in the process and $\sigma=\{2,4,8\}$, the forgetting measure has been calculated. When two learners have been used, the forgetting score has been measured for each of which (i.e., l_1, l_2).

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 δ is more than 20 times higher than the shallow network for all levels of noise. A similar effect is observed for ϵ 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 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

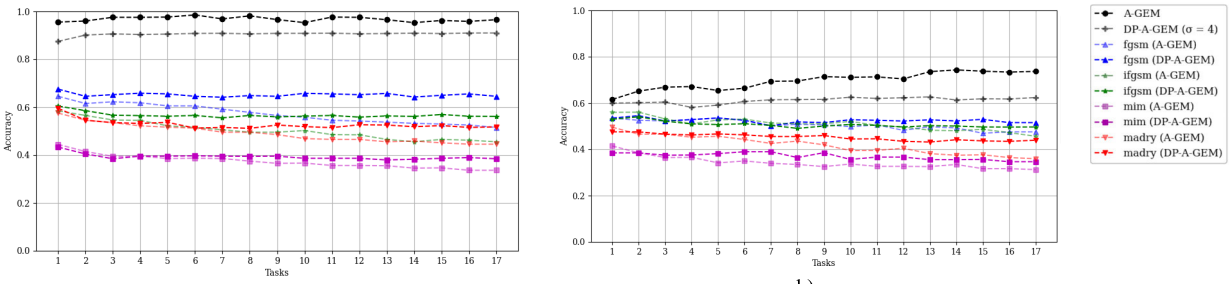

FIGURE 9. The (certified) accuracy of A-GEM algorithm (light colors) and DP-A-GEM algorithm (dark colors) after applying attacks on PMNIST (a) and SCIFAR (b) datasets has been measured (i.e., $l_\infty\ (\mu = 0.1)\ , \sigma = 4$).

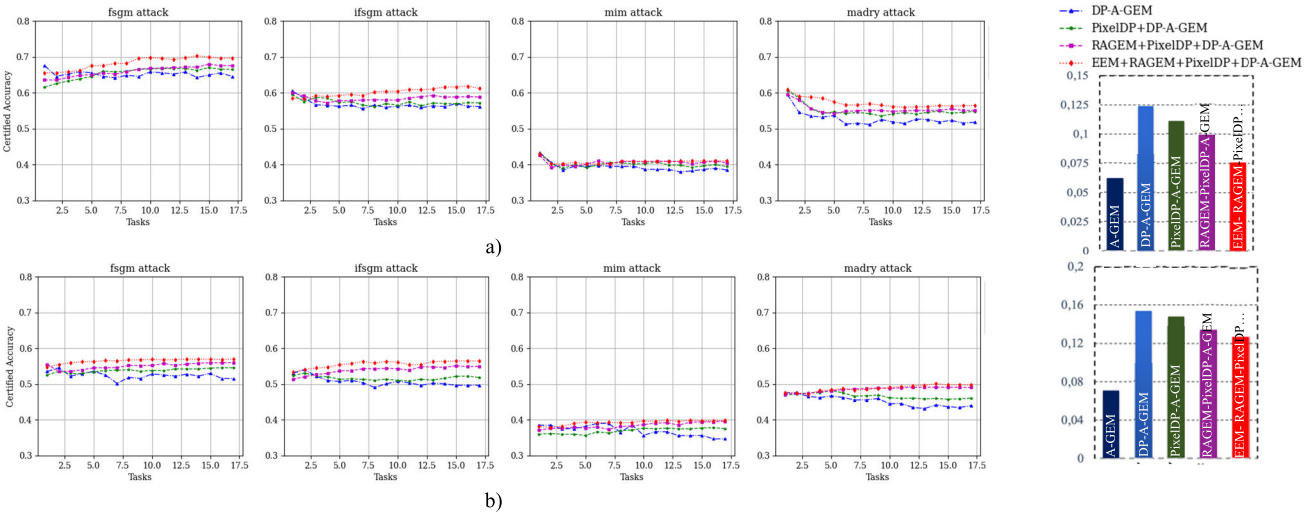

FIGURE 10. By adding the three robustness methods to the DP-CL process, the certified accuracy and forgetting average have been measured after applying four attacks (i.e., fgsm, ifgsm, mim, and madry) on both datasets, (a) PMNIST dataset, (b) SCIFAR dataset (i.e., $l_\infty\ (\mu = 0.1)\ , \sigma = 4$).

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

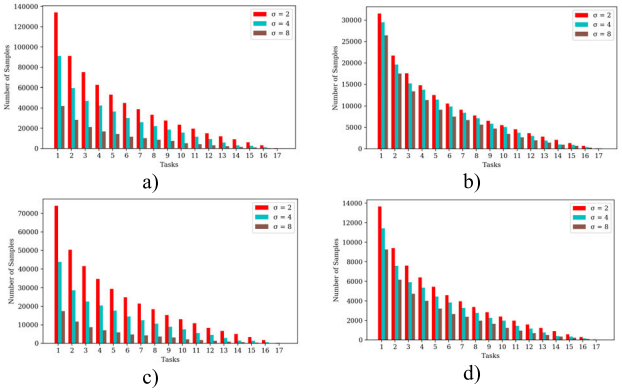

FIGURE 11. The number of replaced training samples have been monitored for the two PMNIST (a and b) and SCIFAR (c and d) datasets when one (a,c) or two (b,d) learners have been used in the training process.

the 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. 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.* [29]. 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.

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:

$$\min_{\tilde{g}_j} \frac{1}{2} \parallel g - \tilde{g}_j \parallel_2^2 \quad s.t. \ \langle \tilde{g}_j, g_{j,k} \rangle \geq 0 \quad \forall k < t \tag{A.1}$$

Replacing $\tilde{g}_j$ with z and rewriting Eq. A.1 yields:

$$\min_{z} \frac{1}{2} z^\top z - g^\top z \ \ s.t. - z^\top g_{j,ref} \leq 0 \tag{A.2}$$

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

$$L(z, \alpha) = \frac{1}{2} z^\top z - g^\top z - \alpha z^\top g_{j,ref} \tag{A.3}$$

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

$$\theta_D(\alpha) = \min_{z} L(z, \alpha) \tag{A.4}$$

Lets 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:

$$\nabla_z L(z, \alpha) = 0$$
$$z^* = g + \alpha g_{j,ref} \tag{A.5}$$

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

$$\begin{aligned} \theta_D(\alpha) &= \frac{1}{2}\left(g^\top g + 2\alpha g^\top g_{j,ref} + \alpha^2 g_{j,ref}^\top g_{j,ref}\right) \\ &\quad - g^\top g - 2\alpha g^\top g_{j,ref} - \alpha^2 g_{j,ref}^\top g_{j,ref} \\ &= -\frac{1}{2} g^\top g - \alpha g^\top g_{j,ref} - \frac{1}{2}\alpha^2 g_{j,ref}^\top g_{j,ref} \end{aligned} \tag{A.6}$$

This solution $\alpha^* = max_{\alpha;\alpha>0}\theta_D(\alpha)$ to dual is given by:

$$\nabla_\alpha \theta_D(\alpha) = 0$$
$$\alpha^* = -\frac{g^\top g_{ref}}{g_{ref}^\top g_{ref}} \tag{A.7}$$

By putting α^* in Eq. A.5, we recover the A - GEM update rule:

$$z^* = g - \frac{g^\top g_{j,ref}}{g_{j,ref}^\top g_{j,ref}} g_{j,ref} = \tilde{g} \tag{A.8}$$

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, Dec. 2015.
[2] C. Su, Z. Xu, J. Pathak, and F. Wang, "Deep learning in mental health outcome research: A scoping review," *Transl. Psychiatry*, vol. 10, no. 1, pp. 1–26, Dec. 2020.
[3] A. I. Karoly, P. Galambos, J. Kuti, and I. J. Rudas, "Deep learning in robotics: Survey on model structures and training strategies," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 266–279, Jan. 2021.
[4] A. Lavecchia, "Deep learning in drug discovery: Opportunities, challenges and future prospects," *Drug Discovery Today*, vol. 24, no. 10, pp. 2017–2032, Oct. 2019.
[5] L. F. Gomez, A. Morales, J. R. Orozco-Arroyave, R. Daza, and J. Fierrez, "Improving Parkinson detection using dynamic features from evoked expressions in video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 1562–1570.
[6] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Netw.*, vol. 113, pp. 54–71, May 2019.
[7] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Feb. 5, 2021, doi: 10.1109/TPAMI.2021.3057446.
[8] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.
[9] X. Huang, D. Kroening, W. Ruan, J. Sharp, Y. Sun, E. Thamo, M. Wu, and X. Yi, "A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability," *Comput. Sci. Rev.*, vol. 37, Aug. 2020, Art. no. 100270.
[10] H. Yousuf, M. Lahzi, S. A. Salloum, and K. Shaalan, "Systematic review on fully homomorphic encryption scheme and its application," in *Recent Advances in Intelligent Systems and Smart Applications*. Springer, 2021, pp. 537–551.
[11] J. Li, X. Kuang, S. Lin, X. Ma, and Y. Tang, "Privacy preservation for machine learning training and classification based on homomorphic encryption schemes," *Inf. Sci.*, vol. 526, pp. 166–179, Jul. 2020.

[12] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, 2002.
[13] C. C. Aggarwal, "August. On K-anonymity and the curse of dimensionality," in *Proc. VLDB*, vol. 5, 2005, pp. 901–909.
[14] J. Brickell and V. Shmatiko, "The cost of privacy: Destruction of data-mining utility in anonymized data publishing," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 70–78.
[15] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.
[16] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 308–318.
[17] J. Zhao, Y. Chen, and W. Zhang, "Differential privacy preservation in deep learning: Challenges, opportunities and solutions," *IEEE Access*, vol. 7, pp. 48901–48911, 2019.
[18] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum, "Differential privacy under continual observation," in *Proc. 42nd ACM Symp. Theory Comput. (STOC)*, 2010, pp. 715–724.
[19] Z. Bu, J. Dong, Q. Long, and S. Weijie, "Deep learning with Gaussian differential privacy," *Harvard Data Sci. Rev.*, vol. 2020, no. 23, Sep. 2020, doi: 10.1162/99608f92.cfc5dd25.
[20] P. C. Mahawaga Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, and M. Atiquzzaman, "Local differential privacy for deep learning," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5827–5842, Jul. 2020.
[21] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive Laplace mechanism: Differential privacy preservation in deep learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 385–394.
[22] M. Lécuyer, R. Spahn, K. Vodrahalli, R. Geambasu, and D. Hsu, "Privacy accounting and quality control in the sage differentially private ML platform," in *Proc. 27th ACM Symp. Operating Syst. Princ.*, Oct. 2019, pp. 181–195.
[23] R. Cummings, S. Krehbiel, K. A. Lai, and U. Tantipongpipat, "Differential privacy for growing databases," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 8864–8873.
[24] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, Q. S. T. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, 2020.
[25] A. Chaudhry, M. A. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-gem," in *Proc. Int. Conf. Learn. Represent.*, 2019.
[26] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory Cryptogr. Conf.*, 2006, pp. 265–284.
[27] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.
[28] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," 2017, *arXiv:1710.06081*.
[29] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017, *arXiv:1706.06083*.
[30] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," 2016, *arXiv:1607.02533*.
[31] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 656–672.
[32] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, and D. Hassabis, "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, 2017.
[33] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3987–3995.
[34] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.
[35] D. Lopez-Paz and M. A. Ranzato, "Gradient episodic memory for continual learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6467–6476.
[36] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 532–547.
[37] M. Gomez-Barrero, E. Maiorana, J. Galbally, P. Campisi, and J. Fierrez, "Multi-biometric template protection based on homomorphic encryption," *Pattern Recognit.*, vol. 67, pp. 149–163, Jul. 2017.
[38] B. Jiang, J. Li, G. Yue, and H. Song, "Differential privacy for industrial Internet of Things: Opportunities, applications, and challenges," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10430–10451, Jul. 2021.

AHMAD HASSANPOUR received the M.S. degree in computer engineering from the Shiraz University of Technology. He is currently pursuing the Ph.D. degree with the Department of Information Security and Communication Technology (IIK), Norwegian University of Science and Technology (NTNU), Norway. He is also a Marie Skłodowska-Curie Fellow and a part of an EU Project called Privacy Matters. His research interests include deep learning, computer vision, and privacy.

MAJID MORADIKIA 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 is currently working as a Postdoctoral Research Fellow with the Engineering Technology Department, University of Houston, TX, USA. His research interests include physical-layer security, the Internet of Things (IoT), privacy, machine learning, computational imaging, and signal processing.

BIAN YANG received the B.S., M.S., and Ph.D. degrees in electronic engineering and information security from the Harbin Institute of Technology, in 2000, 2002, and 2006, respectively. From 2003 to 2005, he visited Fraunhofer IGD, Darmstadt, for research on data hiding and media security. He worked as a Lecturer with the School of Computer Science and Technology, Harbin Institute of Technology, from 2005 to 2007, and a Research Engineer at Thomson Corporate Research, Beijing, from 2007 to 2008. From 2008 to 2015, he was with the Norwegian Information Security Laboratory (NISlab), Gjøvik University College, where he is working on privacy-preserving biometrics, In 2016, he founded and has been coordinating the eHealth and Welfare Security (eHWS) Group, Department of Information Security and Communication Technology, Norwegian University of Science and Technology (NTNU). His research interests include cybersecurity and privacy for e-health and welfare technologies and services, privacy modeling 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 at Virginia Tech. His research interests include the areas of wireless communications and networks, artificial intelligence, cyber physical systems, and security. He has coauthored more than 80 journals 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 (IIK), 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 Convenor 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, 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 and 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 Politecnica de Madrid, Spain, in 2001 and 2006, respectively. Since 2004, he has been with the 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 Postdoctoral Researcher. His research interests include 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, such as TABULA RASA and BEAT in the past and now IDEA-FAST and TRESPASS-ETN, and has attracted notable impact for his research. He is a member of the ELLIS Society. He was a recipient of a number of distinctions, including the 2006 EAB Industry Award, the 2012 EURASIP Best Ph.D. Award, and the 2017 IAPR Young Biometrics Investigator Award. He has received best paper awards at ICB and ICPR. He is an Associate Editor of the IEEE Transactions on Information Forensics and Security and the IEEE Transactions on Image Processing.