Securing Collaborative Deep Learning in Industrial Applications Within Adversarial Scenarios

Christian Esposito, *Member, IEEE*, Xin Su, *Member, IEEE*, Shadi A. Aljawarneh, *Member, IEEE*, and Chang Choi, *Senior Member, IEEE*

***Abstract*—Several industries in many different domains are looking at deep learning as a way to take advantage of the insights in their data, to improve their competitiveness, to open up novel business possibilities, or to resolve the problem that thought to be impossible to tackle. The large scale of the systems where deep learning is applied and the need of preserving the privacy of the used data have imposed a shift from the traditional centralized deployment to a more collaborative one. However, this has opened up several vulnerabilities caused by compromised nodes and inputs, with traditional crypto primitives and access control models exploited to offer protection means. Providing security can be costly in terms of higher energy consumption, calling for a wise use of these protection means. This paper exploits game theory to model interactions among collaborative deep learning nodes and to decide when using actions to support security enhancements.**

***Index Terms*—Adversarial learning, collaborative learning, deep learning (DL), energy efficiency, game theory, privacy.**

I. Introduction

THE long dreamed achievement chased by scientists and technicians is to instruct a machine to perform a certain cognitive task, without having a human operator to directly and precisely program the machine. Such a dream has its foundation in the widely known Church–Turing thesis, and since the workshop at Dartmouth College in 1956 that enshrined the birth of the artificial intelligence (AI) research, multiple AI schemes appeared, many of them falling within the literature of machine learning (ML). With ML, a machine is basically able to underpin the hidden correlations within its inputs, to learn from the provided data, and to formulate a valid output conclusion out of it about a certain phenomenon. Pattern recognition or data classification is among the main uses of such schemes, which encompass neural networks (NN) as the widely known means to have smart machines. After the design of a multitude kinds of classifiers and NN, ML met a progressive cooling of the academia and/or industry interest. The main reason is the failure of these ML solutions to obtain the goals of general AI, and even narrow AI, i.e., to perform the full range of human cognitive abilities, or a limited portion of them. This is due to the fact that a high accuracy is achievable only if a large amount of labeled data is provided and a great effort of hand coding the classifier tuning and rules' definitions is spent, so as to let the ML to perform its job.

The recent proliferation of sensory networks, whose pinnacle is the advent of the Internet of Things (IoT), has called out for a rebirth of ML. In fact, it has posed again the need of providing a sort of intelligence to the machines so as to infer knowledge out of such data and to take the best decision in a smarter and autonomous manner. This is imposed by the envisioned applications aiming at realizing smart houses, smart factories, smart hospitals, etc. The answer to such a demand is represented by deep learning (DL), which represents an evolution of the traditional NN solutions where multiple hidden layers are added between the input and the output layers. The hype around DL has been promoted both by the wide availability of GPUs, offering a faster, cheaper, and more powerful support to the parallel computations underlying any DL execution, and the success of cloud computing, providing practically infinite storage and computation power so demanded by DL solutions.

A. Collaborative DL

The large scale of the target systems, expressed with respect to both the number of the composing nodes and the volume of the managed data, imposes multiple scalability issues on how DL should be performed, and demands a radical change of the ML architectures. Indeed, the traditional deployment of ML

Manuscript received March 19, 2018; revised May 8, 2018; accepted June 21, 2018. Date of publication July 6, 2018; date of current version November 1, 2018. This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education under Grant 2017R1A6A1A03015496 and in part by the National Research Foundation of Korea grant funded by the Government of South Korea (Ministry of Science and ICT) under Grant 2017R1E1A1A01077913. Paper no. TII-18-0706. *(Corresponding author: Chang Choi.)*

C. Esposito is with the Department of Computer Science, University of Salerno, Fisciano 84084, Italy (e-mail: christian.esposito@dia.unisa.it).

X. Su is with the College of IoT Engineering, Hohai University, Changzhou 213022, China (e-mail: leosu8622@163.com).

S. A. Aljawarneh is with the Software Engineering Department, Jordan University of Science and Technology, Irbid 22110, Jordan (e-mail: saaljawarneh@just.edu.jo).

C. Choi is with the IT Institute, Chosun University, Gwangju 61452, South Korea (e-mail: enduranceaura@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2018.2853676

consists in having a centralized computer program running an ML solution, receiving training data, and returning conclusions when tested. However, the large scale of the current applications makes such a traditional deployment nontractable despite the fact that, nowadays, we dispose of enormous computation power. In addition, various applications make use of private data that cannot leave the computing platform of its owner, such as the health information cannot leave the data center belonging to the hospital where they have been produced due to privacy reasons. Both the scalability and privacy issues can be approached by having a distributed deployment of DL, leading to what is currently known as collaborative DL (CDL), also referred in certain papers as federated learning. Specifically, such a term should not be confused with distributed or parallel learning where the overall DL solution is architected to be run on multiple modules running on computing nodes interacting by exchanging messages among themselves over the network (in the case of distributed learning), or by means of a shared memory (in the case of parallel learning). In both cases, there are multiple modules working on the overall available input data. On the contrary, within the context of CDL, we have that the overall input data are partitioned in fragments, and each of them is provided as input to a given, local module of the solution. Therefore, the training data are left where they have been produced or must reside, so as to train a local DL module. The output of the CDL solution is given by a single, global, module that properly aggregates the outputs obtained by the local modules, so as reach a decision close to the one that the overall process would have taken if it has been executed in a centralized manner. In such a way, private data are not propagated outside the administrative domains owning them, so privacy is enforced, and the workload of running DL is distributed among the involved computing infrastructures.

B. Security and Privacy in DL

The criticality of the operations supported by DL (e.g., controlling the maintenance of transportation infrastructures, diagnosing diseases to patients, or predicting critical failure of key components within an advanced ICT infrastructure) and the sensitivity of the handled data, which can contain personal information or allow adversaries to reconstruct the user habits, is calling out for the proper security and privacy provisions when DL is applied to these critical applications. Nowadays, this is particularly demanding since heavy ML computations are preferred to be outsourced to the cloud, realizing the so called "ML as a service" [1], so as to take advantage of its elastic resource provision. This is achieved at the cost of losing control over the data involved during the DL execution, and open up the possibility of several kinds of cyber-attacks. It is possible to have attacks, such as false data injection, denial of service, or man-in-the-middle, to compromise the DL engine so as to bring it to return erroneous outputs. Another adversary goal may be to steal data during the training or the testing of the DL module, or even the DL parameters after their tuning. Within the current literature, several cryptographic and authorization means have been applied to DL products so as to secure the communication of users, protect the data stored within the computing infrastructure hosting the DL engine or to avoid unauthorized use of the engine. The most adopted solution is to use encryption so as to protect the exchanged messages from eavesdropping, or to build a DL solution by running computations over encrypted data, or the introduction of randomness so as to cope with intentionally erroneous training sets. However, such means are only able to protect against attacks coming from outside the DL, but are ineffective if the attack comes from the inside of a DL solution, such as a compromised module violating the collaboration protocol. A compromised participant to the CDL or erroneous training data, i.e., directly compromised at its source and not when transmitted between two modules of the CDL solution, cannot be nullified by these traditional security-related solutions, calling out for a more effective protection measure. Moreover, the use of each of these solutions implies a cost in terms of higher response time from the CDL solution to a given user request or even a waste of energy. Typically, such security enhancements are executed at all times, even if it may be probable that an attacker or a threat is not active at a given time. This is a waste of energy or quality since the running protection is un-needed.

C. Our Contribution

This paper does not want to substitute the traditional security means based on crypto primitives and access control models, but to augment them with a way to detect byzantine local modules, i.e., those providing erroneous inputs to the global module since compromised or working on compromised training inputs, and to wisely use the traditional security means so as to reduce the waste of energy by keeping a high level of protection for the overall system. Our driving idea is to turn on and off the security protections in a wise manner, respectively, when it is highly probable that they are needed. To this aim, we have modeled the interactions among local and global modules by means of game theory so as to discriminate when using encryption, as an example, and/or when a given input must be accepted by the global module or when it should question its trustworthiness. To this aim, we exploit the game theory in order to model the interactions between the different modules within a CDL solution and to deal with an energy-efficient use of existing solutions for secure and private DL use. Specifically, our detailed contributions are the following ones:

1) modeling the actions between a provider and a requestor within a CDL architecture by expressing the pros and cons of encrypting the messages and/or using the gathered data to feed a module composing a CDL solution;
2) studying the solution concepts of such a model so as to determine under which conditions the best decisions are taken, i.e., the ones offering a high degree of security and privacy without wasting an excessive amount of energy;
3) assessing by means of simulations the quality of the decision rules obtained by studying the formulated game and its solution concept.

D. Roadmap

The rest of the paper is structured as follows. Section II introduces the needed notions of DL to support our discussion, and highlights the available work related to secure DL in this paper.

Section III describes our approach based on game theory, for defending the global module against compromised local modules. Section IV assessed the quality of the proposed solution by means of simulations. We conclude this paper with some final remarks and a plan for future work in Section V.

II. Background and Related Work

Recently, the need of achieving high performance and dealing with large-scale problems has been coming up beside the growing interest for the privacy of the training data. Due to the fact that many companies do not own the IT infrastructures required to run the current DL solutions, such as a cluster or cloud of CPUs/GPUs, the training data can be exposed to internal attacks run by the staff employed at the providers of these IT infrastructures, or external attacks toward the data exchanged by the users with the DL solution. Since it is possible that such data may contain sensitive information, which must be protected from unauthorized access to safeguard the privacy or security of an individual or organization, it is not possible to store or even forward them to the IT infrastructure so as to train the DL solution. To this aim, a series of collaborative architectures have emerged within the current literature, as described in [2], where the DL module is not divided into parallel modules so as to augment the performance and reduce the training efforts, but organized in interacting modules, each working with a fragment of the overall training and test data, so as to keep data sovereignty. Specifically, a given DL module, such as a stochastic gradient descent, is deployed at the premises of the owner of a specific piece of sensitive information, so that such information does not leave the allowed organizational boundaries. A centralized server, maybe hosted within the cloud, can run a proper aggregation function, such as a model averaging, on the received outputs from all the local modules.

A. Available Protection Means

If we consider only external attacks, the first vulnerability is represented by the data that users send to an ML model, or the outputs of the local models sent to the global one. An adversary may be able to intercept these data in order to access sensitive information or use the output of an ML solution in order to reconstruct sensitive information. The protection from stealing training data is to encrypt it [3] and to let the ML module to work on encrypted data, and also to provide encrypted output somehow [4]. A concrete example of this solution is the work presented in [5], where the training data are protected by using a scheme by Brakerski, Gentry, and Vaikuntanathan (indicated as BGV encryption), which is one of the possible fully homomorphic encryption (FHE) schemes [6], and the Taylor theorem to approximate the Sigmoid activation function typically used by the neurons in a DL solution (since this encryption scheme does not support the exponentiation operation). FHE allows us to compute arbitrary polynomials on encrypted data without ever having to decrypt it first, so that the disclosure of the training data is avoided and the ML solution is still able to compute its output. The idea of combining NN and FHE is at the basis of the framework proposed by Microsoft researchers, and called Crypto-nets [7], which has been further evolved in [8], so as to be efficiently applied to deeper NN as demands by DL. Also the work in [9] use cryptography, with linearly homomorphic encryption (LHE) [10], which is faster than the one in the previous work, combined with oblivious transfer [11], which is a cryptographic primitive to forward one of potentially many messages to a receiver, without knowing which specific message has been transferred, for privacy-preserving NN training. In [12], in addition to homomorphic encryption (HE), threshold secret sharing (TSS) [13], which is a primitive to securely distribute a secret towards multiple destinations. TSS is used with HE to construct a gradient exchange protocol among the local ML modules without a global one thanks to a secure multiparty computation, as also done in [9]. When data are owned by multiple entities and the ML module is global among them, there is a problem of multiple keys being used within the encryption, the work in [14] and [15] deals with it by using a multikey HE. It is important to impose nonmalleability of the used crypto-primitives for secure DL, especially when outsourced to the cloud [16]; however, such a point at the time has been scarcely investigated. Such a property consists of an adversary being able to transform a ciphertext, which decrypts m into another ciphertext, which decrypts the result of a function applied to M without knowing M.

An alternative approach is the one making use of data masking [17], as a way to protect the training data when transferred by hiding the original data with some randomly generated noise. Adding noise to the inputs, commonly based on Gaussian and Laplace-like distributions, is a common practice in ML, as presented in [18], not mainly for privacy purposes; but, when this is done in order to support the privacy of the training data, this must be carefully done in order to avoid compromising of the training process [19]. Within this class of solutions, we can find a technique that found quite some success in order to design privacy-preserving ML, namely differential privacy [20], whereas in [21], it is used to introduce randomness into the training process and make more difficult to extract part of the training data from a trained ML model, which is an attack called model inversion. Another kind of attack is membership inference, aiming to determine if a given piece of data has been used to train a certain ML model. Such an attack is conducted, as described in [22], with multiple shadow models, whose training dataset is known, that try to imitate the behavior of the target model. Differential privacy provides a way to protect against such an attack [23], however, if the adversary is strategic enough to mimic the behavior of the applied defense, then we have that the protection degree is very small, if not nullified. Final, when outsourced to a cluster or cloud, the ML module parameters can be easily stolen after its training [24]. If we consider that the training can take weeks or months and is based on data not easily obtainable, such parameters are valuable for a company or organization that have spent their resources and efforts in the ML creation and training. Such an attack can be made in a black-box approach, by the attacker learning a model that approximate well the one under attack, or in a white-box approach, by having the attacker accessing directly the ML module and taking its key parameters. For the first case, the proposed defensive solution is to round off

the confidence scores to some fixed precision and/or to apply differentially private learning. For the second case, a defense has not been yet found, and the hosting infrastructure has to be trusted.

If we focus on internal attacks, we have the last kind of possible attacks, also called causative, represented by poisoned samples provided as inputs to an ML module by an attacker [25], able to craft these samples so as to cause a misclassification, increase the complexity of the learning, or implying worse performances. A possible defense is to analyze the received samples before using them for training the ML module, as an example in [25], Papernot *et al.* have used a hardness measure by normalizing the average distortion of a set of samples or the adversarial distance of a given sample to a target class. Also in [26], this kind of attack is investigated and a series of countermeasures have been proposed: A data sanitization scheme, called reject on negative impact (RONI), is adopted by measuring the effect of considering a sample during the training and rejecting it if it has a negative impact on the classification, or a robust learning is employed by reducing the effects of outliers. Robust learning is provided in [27] by means of the so-called generative adversarial networks (GAN) where an adversarial discriminator is used so as to distinguish between real and poisoned samples. In [28], a different approach is adopted by means of a two-person sequential noncooperative Stackelberg game between the adversary providing poisoned data and the ML module for robust learning, so as to model when an attack can be performed and when the module ignores the received sample.

B. Problem Statement

The problem of ML, and similarly of DL, is to find a function h and a parameter vector ω so that the distance (in terms of misclassification) between the computed output $h(x;\omega)$ based on an input sample x and the right label to be assigned to the sample y is minimized

$$\min_{\omega\in\Omega} f(\omega) = \frac{1}{|Z|}\sum_{(x,y)\in Z} l(h(x;\omega),y) + \rho(\omega) \qquad (1)$$

where $\rho(\omega)$ is a regularizer to avoid overfitting of the ML model. When collaborative ML is used, such an equation must be partitioned among the multiple local modules, whose number is M, which may not share the same set of samples for training

$$\min_{\omega\in\Omega} f(\omega) = \min_{\omega\in\Omega} \frac{1}{M}\sum_{i=1}^{M} f_i(\omega)$$

$$= \min_{\omega\in\Omega} \frac{1}{M}\sum_{i=1}^{M} \frac{1}{|Z^i|}\sum_{(x,y)\in Z^i} l(h_i(x;\omega),y) + \rho(\omega). \qquad (2)$$

The role of the global module is to make an average of the outputs of the local modules by applying a proper weighting scheme. To this aim, it will periodically ask for the output of local modules on certain samples so as to tune the weights. To provide security and privacy, it is of pivotal importance to protect the interaction of the global module with the local one, mainly applying encryption, to discard the reply from a compromised module and also considering the resource-constrained nature of the involved nodes. In fact, CDL can be applied to local modules running on sensors, smartphones, or other devices where energy is of pivotal importance and cannot be wasted. In addition, it is also possible to have the global module running on these kinds of devices, as many DL applications can be related to IoT. Therefore, the problem is not only to provide security, as illustrated in Section II-A, but also to take a strategic decision to properly use these schemes when needed so as to have a low impact on the energy consumption of the involved devices.

III. Game Theoretic Defense Against Compromised Local ML Modules

The interaction by a local module and a global one, both indicated as players and, respectively, named as provider and requestor, during the training of the second one based on a partial or complete training of the first one is modeled in this paper as a signaling game, which is an example of a dynamic Bayesian game, where each player does not know the admissible strategies, payoff characterization, action mapping, or the true intentions of the other one, and the game is run sequentially by a player deciding its actions before the other one, which must react to such an action with the best response, and having an advantage over the player that has to response.

A game is composed by a set of players and, in this case, we have two players sequentially interacting among each other, i.e., the reputation requestor and the provider, where both do not have a complete information about the other one (i.e., if the reputation requestor can trust the provider or not). This privileges a Bayesian formulation of the game over the other ones available in the literature of game theory. Moreover, the two players do not act simultaneously by the provider replies to a request and the requestor has to determine what do to with the received reply. This implies that the game is not simultaneous, but sequential with a leader (i.e., the provider) and a follower (i.e., the requestor). Such a situation can be perfectly modeled as a signaling game, which consists of two players exchanging messages, where the first one is the one starting the game (i.e., how the provider reacts when receiving a request) and the second one acting after the reception of a message from the first one, which contains a reputation degree. Generally, in a signaling game, only one player has private information, i.e., the leader, whereas all the details of the second one are known to both. On the contrary, in our case, such a game has two privately informed players, as modeled in the courtship game [29]: The reputation requestor and provider are aware of their respective types (i.e., being honest or malicious), and behave according to it, however, a player is not aware of the type of the other one with which it is interacting. Both players select a proper strategy, each characterized by a payoff in terms of security and costs, mainly looking at the energy consumption as a cost. Such a game represents a framework for considering the effects of deception, where there is a possibility that the provider may take advantage of the information-asymmetric nature of the game so as to deceive the requestor and to make it to take nonbeneficial actions; but, even the opposite is possible. Final, our envisioned

game is not an instance of the cheap talk games [30], since sending a message is not costless, especially if encryption is adopted to protect the communications or guaranteeing the integrity and nonrepudiation of the exchanged messages, and the achievable payoff by the provider is strongly affected by the costs of transmitting information toward the requestor. Also, the payoff of the requestor is based on the correctness in the trust estimation and the cost of the performed actions.

The provider will, during its interaction with the requestor, communicate a message containing the computed reputation degree for the requested node in the network, which is defined as a signal in this specific game, and the content of the exchanged messages depends on the type of the provider. More formally, a reputation requestor and provider are characterized by a type $\mathcal{T}$, whose value can be $\mathcal{T}_H$ and $\mathcal{T}_M$, respectively, to indicate an honest player and a malicious one. On the one hand, in the case of a reputation requestor, the honesty means that the collected reputation is used for computing the trust degree of a given node of the network, whereas the maliciousness indicates that the collected reputation is used to perform cyber-attacks to the network. On the other hand, in the case of a reputation provider, an honest player always sends correct information (namely by means of c signals), marginally altered by a zero-mean white noise; on the contrary, a malicious one tries to deceive the requestor with false information on the reputation of the requested node (namely by means of d signals). Both signals can be sent as plain-text or coded by using a specific encryption scheme. The types $\mathcal{T}_\mathcal{P}$ and $\mathcal{T}_\mathcal{R}$, respectively, for the provider and the requestor are not decided during the game, but by an exogenous force or player, named nature, which is external and the source of the asymmetric information characterizing the game. With two given probabilities θ_P and θ_R (which may assume the same value), nature assigns $\mathcal{T}_M$ to the two players (θ_P is the probability of the provider being malicious, whereas θ_R is the one of the requestor being malicious), while with the reciprocal probability $1-\theta_P$ or $1-\theta_R$, the players have type $\mathcal{T}_H$: $\theta_P = P(\mathcal{T}_\mathcal{P} = \mathcal{T}_M)$. Therefore, the provider, depending on its type, can send two kinds of messages or signals, which are as follows:

1) The signal c contains the real computed reputation degree: $\rho_{\text{prov}} = \rho_{\text{real}} + \epsilon$, where ϵ is the zero-mean white noise.
2) The signal d contains the altered reputation degree: $\rho_{\text{prov}} = \rho_{\text{real}} + f$, where f is the altering factor chosen in the $[-0.5, 0.5]$ interval, and $f \gg \epsilon$.

In both cases, ρ_{prov} cannot exceed 1 or being lower than 0 so as to keep it within the $[0, 1]$ interval; therefore, firstly, if the sum in the computation of ρ_{prov} exceeds 1, then we have that $\rho_{\text{prov}} = 1$; secondly, if the sum is lower than 0, then $\rho_{\text{prov}} = 0$. When the provider is of type $\mathcal{T}_M$, we indicate the probability that the provider sends the signal d with p_M, whereas we indicate the probability that a provider of type $\mathcal{T}_H$ forward the signal c with p_H. These probabilities allow us to map from the provider types to their actions. Moreover, the provider has two ways of emitting its signals: The first one consists in a plain-text messages, and the second one is made of encrypted messages. The first one is prone to falsification and use of the messages for impersonation and cyber-attacks to the wireless sensor network, whereas the second one is more secure, removing the possibility of commencing attacks with these messages. However, the first communication means it is less expensive for a sensor to perform, whereas the second one has additional costs of executing the encryption and exchanging an additional number of bytes. Also, for the receiver, an encrypted message is more costly since decryption must be run and the node receives more bytes than needed. The provider is not obliged to respond to a request, but it is possible that it will not emit any signals. Similarly to the decision of which signal to send, we have a couple of probabilities disciplining when a plain and an encrypted signal should be emitted by the provider. When the provider is of type $\mathcal{T}_M$, we indicate the probability that the provider sends a generic signal x in an encrypted manner with $p_{M,\text{enc}}$, whereas we indicate the probability that a provider of type $\mathcal{T}_H$ forward a generic signal x in a plain manner with $p_{H,\text{enc}}$. These probabilities allow us to map from the provider types to their actions.

The requestor, upon the reception of the signal, is not aware of the true nature of the provider, neither has information useful to infer it, but must take an action only based on the received information, i.e., the provided message. One possibility for the requestor consists in completely trust the received reputation degree, and to use it, together with the other collected ones, to estimate the trustworthiness of the node of interest. Alternatively, the sensor does not trust the received information, and may commence a challenge with the provider in order to check the received reputation, with the intent of revealing its true nature. This is possible with one of the above-mentioned means to detect outliers by means of thresholds. This implies higher costs than the first option, but is able to achieve a higher resilience to threats. In addition, the requestor can receive plain-text messages and ones with encrypted content. In the first case, the requestor is vulnerable to possible manumissions occurred along the channel or to camouflaged providers, whereas in the second case, the requestor has more security that the message comes from the claimed sender and with an integrity assurance of the content. Also, the requestor can be malicious and honest; in the first case, the received messages can be used to commence attacks to the network, whereas in the second case, the message contents are only used in the proper trust evaluation. Therefore, a malicious requestor is negatively affected by the reception of encrypted messages, since it cannot use them in its eventual attacks.

Based on the emitted signals from the provider and the consequent actions of the requestor, the two players of our game obtain a payoff from taking part to it. Fig. 1 shows the time structure of the game in the so-called extensive form, which indicates the sequencing of the possible moves between the provider and the requestor, a collective of decision points where a player should decide its move, and the payoffs obtained by the two players in the game for all possible game outcomes. In the figure, the exogenous actions determining the type of the two players, which represent the incomplete information in the game, are depicted as "moves by nature." Specifically, for each outcome of the game, i.e., represented at the lowest level of the figure, a pair of payoff are indicated, where one is advocated to the provider and one for the requestor. The payoff charac-

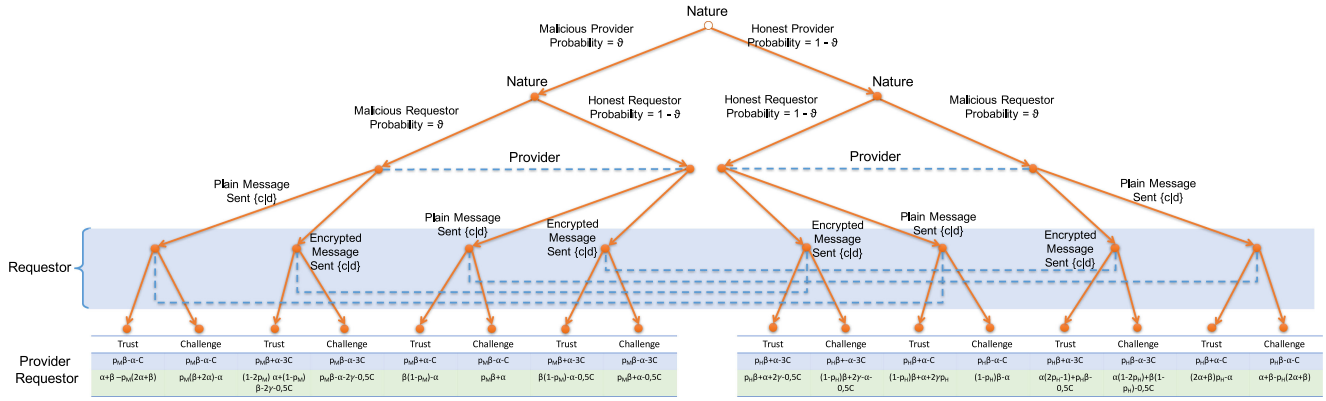


Fig. 1. Simplified game between provider and requestor.

TABLE I
PAYOFF CHARACTERIZATION FOR THE HONEST PROVIDER

Provider		Requestor		Payoff Functions	
Type	Signal	Type	Action	(Provider's Payoff, Requestor's Payoff)	Simplification
$\mathcal{T}_H$	c	$\mathcal{T}_H$	T	(D+E-C, A+F)	(α+β-C, α+γ)
			C	(E-D-C, -A)	(β-α-C, -α)
		$\mathcal{T}_M$	T	(D+E-C, A+F)	(α+β-C, α+β)
			C	(E-D-C, -A)	(β-α-C, -α)
	d	$\mathcal{T}_H$	T	(D-C, A+B)	(α-C, α+β)
			C	(-D-C, F-A)	(-α-C, β-α)
		$\mathcal{T}_M$	T	(D-C, -A)	(α-C, -α)
			C	(-D-C, A+F)	(-α-C, α+β)
	E(c)	$\mathcal{T}_H$	T	(D+E-(C+δ_P), A+F+B-δ_R+G)	(α+β-3C, α+β+2γ-0,5C)
			C	(E-D-(C+δ_P), B-A-δ_R+G)	(β-α-3C, 2γ-α-0,5C)
		$\mathcal{T}_M$	T	(D+E-(C+δ_P), A-B-δ_R+F+G)	(α+β-3C, α+β-0,5C)
			C	(E-D-(C+δ_P), -A-B-δ_R+G)	(β-α-3C), -α-0,5C)
	E(d)	$\mathcal{T}_H$	T	(D-(C+δ_P), A+B-δ_R+G)	(α-3C, α+2γ-0,5C)
			C	(-D-(C+δ_P), B-A-δ_R+F+G)	(-α-3C, β+2γ-α-0,5C)
		$\mathcal{T}_M$	T	(D-(C+δ_P), -A-B-δ_R+G)	(α-3C, -α-0,5C)
			C	(-D-(C+δ_P), A-B-δ_P+F+G)	(-α-3C, α+β-0,5C)
	-	$\mathcal{T}_H$	-	(0,0)	(0,0)
		$\mathcal{T}_M$	-	(0,0)	(0,0)

TABLE II
PAYOFF CHARACTERIZATION FOR THE MALICIOUS PROVIDER

Provider		Requestor		Payoff Functions	
Type	Signal	Type	Action	(Provider's Payoff, Requestor's Payoff)	Simplification
$\mathcal{T}_M$	c	$\mathcal{T}_H$	T	(D-C, -A+F)	(α-C, -α+β)
			C	(-D-C, A)	(-α-C, α)
		$\mathcal{T}_M$	T	(D-C, A+F)	(α-C, α+β)
			C	(-D-C, -A)	(-α-C, -α)
	d	$\mathcal{T}_H$	T	(S+E-C, -A)	(α+β-C, -α)
			C	(E-S-C, A+F)	(β-α-C, α+β)
		$\mathcal{T}_M$	T	(S+E-C, -A)	(α+β-C, -α)
			C	(E-S-C, -A)	(β-α-C, -α)
	E(c)	$\mathcal{T}_H$	T	(D-(C+δ_P), F-A+B-δ_R-G)	(α-3C, β-α-0,5C)
			C	(-D-(C+δ_P), A+B-δ_R-G)	(-α-3C, α-0,5C)
		$\mathcal{T}_M$	T	(D-(C+δ_P), A+F-B-δ_R-G)	(α-3C, α+β-2γ-0,5C)
			C	(-D-(C+δ_P), -A-B-δ_R-G)	(-α-3C, -α-2γ-0,5C)
	E(d)	$\mathcal{T}_H$	T	(S+E-(C+δ_P), B-δ_R-G-A)	(α+β-3C, -α-0,5C)
			C	(E-S-(C+δ_P), F+B+A-δ_R-G)	(β-α-3C, β+α-0,5C)
		$\mathcal{T}_M$	T	(S+E-(C+δ_P), -A-B-δ_R-G)	(α+β-3C, -α-2γ-0,5C)
			C	(E-S-(C+δ_P), A+F-B-δ_R-G)	(β-α-3C, α+β-2γ-0,5C)
	-	$\mathcal{T}_H$	-	(0,0)	(0,0)
		$\mathcal{T}_M$	-	(0,0)	(0,0)

terization in the fourth column of Tables I and II is based on a proper sum of gains and costs expressed by positive integers, as follows.

1) A is the gain achieved by the requestor for trusting an honest signal c or challenging a malicious signal d, on the contrary, it becomes the cost to pay for trusting a malicious signal d or challenging an honest signal c.
2) B is the gain that the honest requestor gets when receiving an encrypted message, thanks to the additional security provided by the use of the encryption.
3) C is the cost that a provider has to pay when sending a message, despite of its content, and such a value is augmented by the δ_P factor in the case of encrypted messages, where $\delta > C$, and such a factor, namely δ_R, is also a cost for the requestor when receiving an encrypted message.
4) D represents the gain that the provider gets if the requestor accepts its honest signal c, but it is also the costs paid in the case the requestor challenges its honest signal c.
5) E represents the gain that the provider gets when follows its type, such as sending an honest signal c when its type is $\mathcal{T}_H$ or sending a malicious signal d when its type is $\mathcal{T}_M$.
6) S represents the gain that the provider gets if the requestor accepts its malicious signal d, but it is also the costs paid in the case the requestor challenges its malicious signal d.

7) F represents the gain that the requestor gets when it follows its type, such as trusting an honest signal c or challenging a malicious signal d.
8) G represents the cost that the requestor gets when receiving an encrypted message that cannot be used for its attacks when the requestor's type is $\mathcal{T}_M$ or is the gain that an honest requestor gets for receiving an encrypted message.

Without loss of generality, we can introduce a simplification of such factors as follows, so as to obtain a simplified payoff characterization in the fifth column of Tables I and II.

1) We can assume that A, D, and S factors can have the same value, indicated as α.
2) We can indicate δ_P is twice C, whereas δ_R is equal to $\frac{C}{2}$.
3) We can assume that E, F factors can have the same value, indicated as β.
4) We can assume that B, G factors can have the same value, indicated as γ.

Furthermore, we can notice that α and γ indicates the achievable security degree, whereas C is the energy consumption due to the message sending operations, so these three must have the same values. Only β is secondary, so we can assume that it holds a value that is half of the other one. We can further simplify our game by determining when a provider returns an honest signal or an altered one. Specifically, we can combine the payoff for the probabilistic outcomes of trusting or challenging a given signal from the provider. Moreover, we can consider that we can omit the case of the provider not replying to the request so as to not have partially filled matrixes. This is due to the fact that it is not difficult to prove that the best response functions in our game is always positive, making impossible for the provider to select the no message as a suitable action. This is due to the fact that the provider has always a better choice than being silent in any of its possible types, so that we have omitted such a case from our discussion. Based on these considerations, we have a simplified form of our game, where we also assume that the two probabilities θ_P and θ_R are equal to the value θ.

Let us notice that due to the asymmetry in the game information, the requestor is not able to distinguish between several information nodes joined by a dashed line in the figure. For a concrete example, when receiving a plain-text message and its type is $\mathcal{T}_M$, it cannot distinguish between the case of the message sent by a malicious provider and an honest one (similarly to other cases). In game theory, we distinguish between pure strategies and the mixed ones. In the first case, we have that players will always select a specific move or action in every possible attainable situation in a game, whereas in the second case, a probability is associated with each selection. Specifically, in our envisioned game, a requestor randomizes its actions based on the received signal with a given probability: firstly, x is the probability to trust a plain c signal, whereas y is the probability to challenge a plain d signal; secondly, x_{enc} is the probability to trust an encrypted c signal, whereas y_{enc} is the probability to challenge an encrypted d signal. These probabilities allow us to map from the provider types to their actions.

Differently to what is assumed for the providers, such probabilities does not depend on the type of the requestor. In addition, the requestor and the provider can build a belief on the type of the opposing player, thanks to collected feedback from direct interaction of the requestor with the provider, or from indirect reputation scores collected by the provider about the requested. Such beliefs are defined over the interval $[0, 1]$ and influence the decision of the action that the players have to take after having received a given signal from the provider or a request from the requestor. Specifically, we indicate the beliefs of the provider with q_{req} and p_{req} with respect to the requestor to be, respectively, of type $\mathcal{T}_M$ or $\mathcal{T}_H$, where q_{prov} and p_{prov} are the dual ones for the provider.

The widely known solution concept in game theory is represented by the Nash equilibrium (NE), which represents the situation where a player picks up the strategy able to return the maximum payoff, so that selecting any other strategy is not more profitable than the one at the equilibrium. Within the context of games with imperfect information, as the one that we have previously formalized, the solution concept provided by NE is re-formulated by making the players to consider in the strategy selection not only the achievable payoff but also their believes over opponent types, so as to obtain the solution concept named Bayesian Nash equilibrium (BNE). Although we have slightly simplified the model of our game, the precise identification of the pure strategy BNE is extremely complex due to the large number of possible strategies and the two-sided information asymmetry. To this aim, it is possible to introduce the concept of subgame perfect equilibrium, which represents an NE of every subgame of the original repeated game. By combining the BNE and the sequential rationality, we obtain the perfect Bayesian equilibrium (PBE) that indicates strategies that are sequentially rational for a given belief system, which is the intent of our analysis. In those games and the relative equilibria, we have two extreme situations: separating and pooling equilibria. In the first case, the signals are able to completely expose the type of the sender, i.e., the anchor in our case, since the player always sends a given signal based on its assigned type, as this is the dominant strategy for the sender. On the contrary, in the second case, the anchor sends the same signal for each of the possible types that it can assume, so the sensor is not able to distinguish among the sender types based on the received signal.

Our starting point is analyzing the strategy profiles that conveys information: The provider's signals conveys its type (i.e., the provider sends an encrypted signals only if he is honest and a plain-text one only if he is malicious), and the requestor acceptance conveys its type (i.e., the requestor trusts the encrypted signal from the provider and challenge any plain-text signal only if honest). Therefore, we will consider the separating PBE with the following scenario. The provider sends an encrypted message only if honest, containing the c signal, while it sends a plain message with a d signal if malicious. The requestor accepts the provider's message if and only if it is honest, and has the following believes: If the provider has sent an encrypted message, then it is honest with probability equal to 1, whereas if the provider has sent a plain-text message, then it is malicious with probability equal to 1. When the requestor received an encrypted message and it is honest, the strategy of trusting it is preferable to the one of challenging it when $p_H\beta + \alpha + 2\gamma - 0,5C$ is

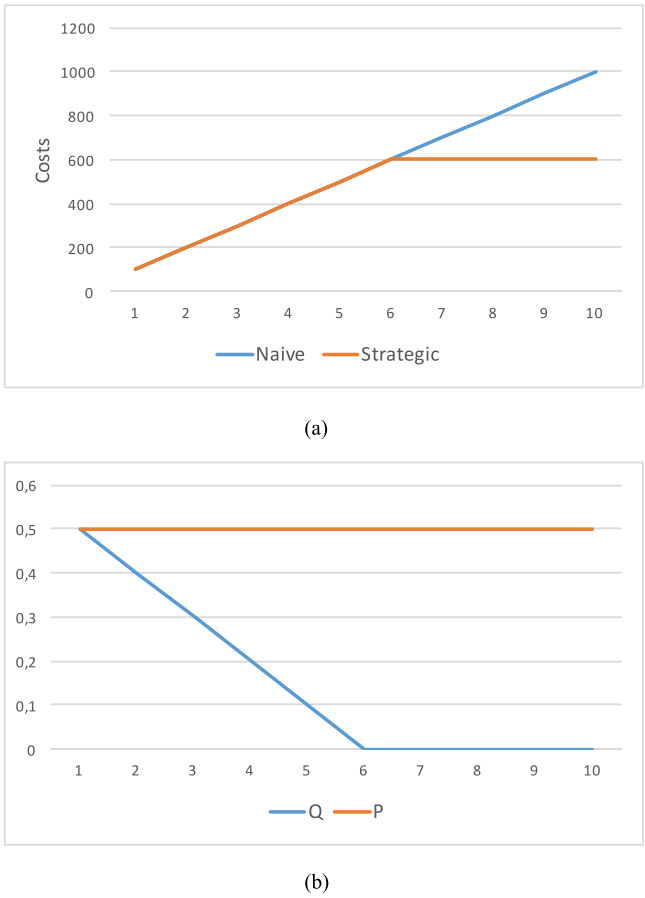


Fig. 2. Results when a semiseparating anchor with the d signal sent when honest.

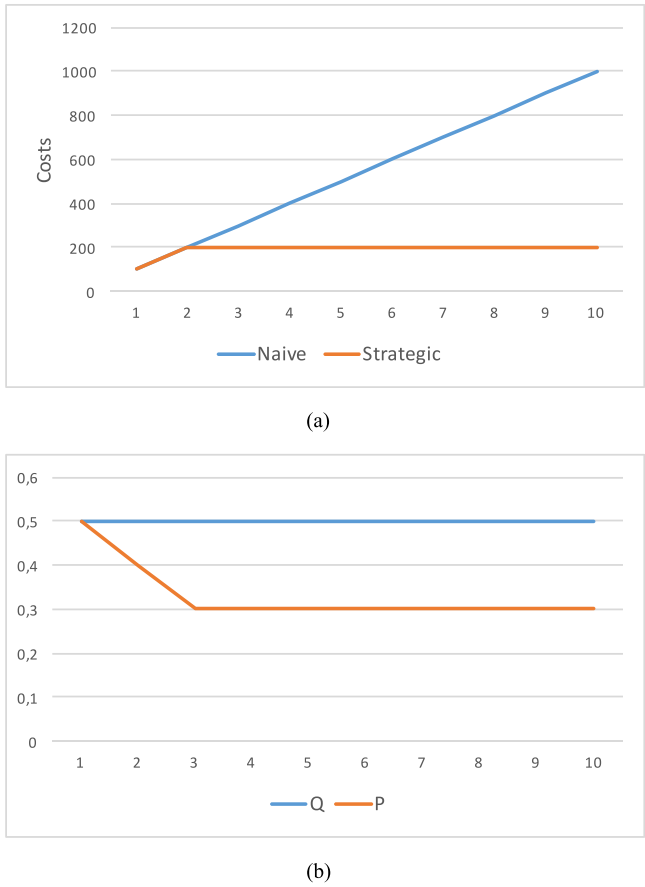


Fig. 3. Results when a separating anchor with the d signal sent when honest.

greater than $(1-p_H)\beta + 2\gamma - \alpha - 0,5C$, which holds when $\beta \geq -\frac{2}{2p_H - 1}\alpha$. Such an inequity is always true when we have that $p_H \geq \frac{0,5\beta - \alpha}{\beta}$, considering that p_H is a probability and cannot assume negative values, so we have that $\beta \geq 2\alpha$. This allows us to simplify our condition as $p_H \geq 0$. On the contrary, when the requestor is malicious and receives an encrypted message, it also accepts the signal if $(p_H - 0,5)\beta \geq (1-p_H)\alpha$ holds, which leads to $p_H \geq \frac{\alpha + 0,5\beta}{\alpha + \beta}$. If we consider the previous relation between α and β, we can say that $p_H \geq 2/3\alpha$. Let us now focus on the case of the reception of a plain-text message, revealing that the provider is not honest. In that case, if the requestor is malicious, it is more convenient to challenge the signal if $p_M \geq 0,5$, whereas in the case of an honest requestor, it is more convenient to challenge the signal if $p_M \geq \frac{\alpha - 0,5\beta}{\beta}$, which is valid if $\beta \geq 2\alpha$. Let us see if the best response is off the equilibrium path. If the requestor is malicious, the best response is to always trust an encrypted signal. If the requestor is honest, the best response is to challenge an encrypted signal emitted by a malicious provider when $p_M \geq \frac{0,5\beta - \alpha}{\beta}$. If the requestor is honest, the best response is to trust a plain signal emitted by an honest provider. If the requestor is malicious, the best response is to trust a plain signal emitted by an honest provider when $p_H \geq 0,5$. As evident, there is a certain deviation for the decision of the provider by seeing the payoff achievable by switching to the off the equilibrium path, making such a situation far from being a PBE.

Let us consider a pooling PBE, which can occur in the situation when the provider always replies with a plain signal, or always with an encrypted signal. In these cases, the analysis of the best response and possible deviations off the equilibrium path are similar to the ones we have seen when analyzing the separating PBE, and let us conclude that there is no PBE even for these two scenarios, since deviations are possible.

Our analysis of the separating and pooling PBE ignored the possibilities of mixed strategies where the two players behave based on their mutual beliefs. To this aim, we have to define a proper belief system, where beliefs are defined over the interval $[0, 1]$ and influence the decision of the action that the requestor has to take after having received a given signal from the provider. We define as $q = Pr(\mathcal{T}_M|\text{plain})$ the beliefs of the requestor that the provider is malicious when having received a plain signal, while $p = Pr(\mathcal{T}_M|\text{encrypted})$ the beliefs of the requestor that the provider is malicious when having received the signal encrypted. Such believes must be consistent to the expectations about the opponent's strategies and formulated by using the Bayes rules.

However, there are several possibilities for mixed strategies in our reputation provider–requestor signaling game that is tough to go through all of them looking for equilibrium points. The

suitable option is to consider those possibilities that are reasonable to be chosen by the two players. Specifically, we consider a semiseparating equilibrium where the provider selects a pure strategy when its type is $\mathcal{T}_M$, namely to send a plain signal, while it randomizes to send the plain or the encrypted signals, when it is of the opposing type. In fact, we have to note that sending the plain signals is a strictly dominant strategy for the provider when it is malicious; indeed, its payoff from following this strategy despite of the choice of the sensor is strictly higher than its payoff from sending the encrypted signals. However, the same does not hold for the provider when being honest: It prefers to send the plain signal when it anticipates that the requestor is honest, whereas it prefers to emit the encrypted signals if the requestor is malicious. Intuitively, the requestor has incentives to trust an encrypted signal emitted by the anchor that is honest, as a consequence, the anchor does not want to convey his type to the sensor by concealing it so as to bring the sensor to a move that is more favorable to the anchor. The belief named q is the one that supports the mixing behavior of the anchor so that the sensor is indifferent between its two actions and the anchor could not anticipate its moves, and can be determined by letting the expected payoff to be equal.

IV. Empirical Assessment

We have implemented the semiseparating PBE, and the naive one coming from the pure strategy equilibria formulation in a Java-based simulation, by having $A = 2$, $B = 1$, $C = 1$, and $\epsilon = 0.1$. Specifically, the behavior of the provider and requestor depends on the current belief values, the observed benefit of the taken decision, and the considerations of the introduced solution concepts. We have modeled all the possible behaviors of the local module, from the separating to the pooling and from the semiseparating to the mixed one. In our simulations, we have observed the divergence of the measured output due to the error caused by trusting a plain signal, and the waste of resources by challenging an honest local module. We have run several simulations when the local module executes the several possible envisioned behaviors, and our solution achieve similar results of the one based on pure strategy equilibria. A particular interesting result is achieved when the local module has a semiseparating behavior with the honest local module sending the encrypted signal. In this case, the naive solution is not optimal, as mentioned, but our strategic solution is able to reduce the costs by reaching PBE_E and stop challenging the honest provider, as illustrated in Fig. 2(a). The equilibrium point is obtained due to the progressive update of the q function, as shown in Fig. 2(b), leading to the equilibrium conditions. Another different experiment is depicted in Fig. 3 and is with a separating local module with the d signal sent when honest, also in this case, the requestor starts by challenging and thanks to an updated p function, it is able to determine that the local module is honest and to save resources by avoiding unnecessary challenges. In these two experiments, the convergence of our approach to the equilibrium depends on the used constant ϵ, which was equal to 0.1 in our experiments. A greater value of this constant is able to accelerate the convergence of the game.

V. Conclusion and Final Remarks

Security and privacy are considered extremely important as ML, especially its deep formulation, is being applied to industrial and critical context. The literature on this topic is starting to be characterized by several proposed solutions; however, it is equally important to have a wise use of the energy since many of these DL solutions are run on resource-constrained devices. Cryptographic primitives are typically used for this scope, but are costly. In this paper, we have modeled the interactions among the modules forming a CDL as a signaling game and used the results of our analysis for architecting the optimal behavior of honest global and local modules. As a future work, we have planned to have a more realistic implementation of our approach by using a representative example of CDL and applying our approach so as to quantitatively assess the suitability of the accuracy and precision of the CDL when using malicious datasets.

References

[1] M. Ribeiro, K. Grolinger, and M. A. M. Capretz, ''MLaaS: Machine learning as a service," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl.*, Dec. 2015, pp. 896–902.

[2] J. Konečnỳ, "Stochastic, distributed and federated optimization for machine learning," arXiv:1707.01155, 2017.

[3] C. Guo, R. Zhuang, Y. Jie, Y. Ren, T. Wu, and K.-K. R. Choo, "Fine-grained database field search using attribute-based encryption for e-healthcare clouds," *J. Med. Syst.*, vol. 40, no. 11, pp. 235:1–235:8, 2016.

[4] Z. Liu, K.-K. R. Choo, and M. Zhao, "Practical-oriented protocols for privacy-preserving outsourced big data analysis: Challenges and future research directions," *Comput. Secur.*, vol. 69, pp. 97–113, 2017.

[5] Q. Zhang, L. Yang, and Z. Chen, "Privacy preserving deep computation model on cloud for big data feature learning," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1351–1362, May 2016.

[6] M. V. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2010, pp. 24–43.

[7] P. Xie, M. Bilenko, T. Finley, R. Gilad-Bachrach, K. Lauter, and M. Naehrig, "Crypto-nets: Neural networks over encrypted data," arXiv:1412.6181, 2014.

[8] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network," *IACR Cryptology ePrint Arch.*, vol. 2017, p. 35, 2017.

[9] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Secur. Privacy*, May 2017, pp. 19–38.

[10] Z. Cao and L. Liu, "On the weakness of fully homomorphic encryption," arXiv:1511.05341, 2015.

[11] M. Naor and B. Pinkas, "Efficient oblivious transfer protocols," in *Proc. 12th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2001, pp. 448–457.

[12] X. Zhang, S. Ji, H. Wang, and T. Wang, "Private, yet practical, multiparty deep learning," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst.*, Jun. 2017, pp. 1442–1452.

[13] A. Beimel, "Secret-sharing schemes: A survey," in *Proceedings of the Third International Workshop on Coding and Cryptology*(Lecture Notes on Computer Science 6639). Berlin, Germany: Springer, 2011, pp. 11–46.

[14] P. Li *et al.*, "Multi-key privacy-preserving deep learning in cloud computing," *Future Gener. Comput. Syst.*, vol. 74, no. Supplement C, pp. 76–85, 2017.

[15] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," *Cluster Comput.*, Apr. 2017.

[16] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," *Inf. Sci.*, vol. 412–413, pp. 223–241, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025517302177

[17] R. Parameswaran and D. Blough, "Privacy preserving data obfuscation for inherently clustered data," *Int. J. Inf. Comput. Secur.*, vol. 2, no. 1, pp. 4–26, 2008.

[18] K. Matsuoka, "Noise injection into inputs in back-propagation learning," *IEEE Trans. Syst., Man, and Cybern.*, vol. 22, no. 3, pp. 436–440, May/Jun. 1992.
[19] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proc. 39th Annu. ACM Symp. Theory Comput.*, 2007, pp. 75–84.
[20] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. Models Comput.*, 2008, pp. 1–19.
[21] M. Abadi *et al.*, "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 308–318.
[22] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy*, 2017, pp. 3–18.
[23] A. Pyrgelis, C. Troncoso, and E. D. Cristofaro, "Knock knock, who's there? Membership inference on aggregate location data," arXiv:1708.06145, 2017.
[24] F. Tramèr, F. Zhang, A. Juels, M. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *Proc. USENIX Secur. Symp.*, 2016, pp. 601–618.
[25] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Secur. Privacy*, 2016, pp. 372–387.
[26] L. Huang, A. Joseph, B. Nelson, B. Rubinstein, and J. Tygar, "Adversarial machine learning," in *Proc. 4th ACM Workshop Secur. Artif. Intell.*, 2011, pp. 43–58.
[27] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[28] W. Liu and S. Chawla, "A game theoretical model for adversarial learning," in *Proc. IEEE Int. Conf. Data Mining Workshops*, 2009, pp. 25–30.
[29] T. Bergstrom and M. Bagnoli, "Courtship as a waiting game," *J. Political Economy*, vol. 101, no. 1, pp. 185–202, 1993.
[30] J. Farrell and M. Rabin, "Cheap talk," *J. Econ. Perspectives*, vol. 10, no. 3, pp. 103–118, 1996.

Christian Esposito (S'06–M'09) received the Ph.D. degree in computer engineering and automation from the University of Napoli "Federico II," Naples, Italy, in 2009.

He is currently an Assistant Professor with the University of Naples "Federico II". He serves as a Reviewer and the Guest Editor for several international journals and conferences (with about 200 reviews being done). He has been involved in the organization of about 40 international conferences/workshops. His research interests include reliable and secure communications, middleware, distributed systems, positioning systems, multiobjective optimization, and game theory.

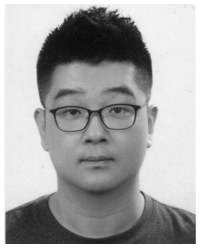
Xin Su (M'16) received the B.E. degree in computer engineering from the Kunming University of Science and Technology, Kunming, China, in 2008, the M.E. degree in computer engineering from Chosun University, Gwangju, South Korea, in 2010, and the Ph.D. degree in the program in IT and media convergence studies from Inha University, Incheon, South Korea, in 2015.

He is currently with the College of Internet of Things Engineering, Hohai University, Changzhou Campus, China. His research interests include 3GPP LTE(-A) systems, 5G nonorthogonal multiple access, MIMO beamforming, edge/fog computing, and mobile ad hoc networks.

Shadi A. Aljawarneh (M'16) received the B.Sc. degree in computer science from Jordan Yarmouk University, Irbid, Jordan, in 1999, the M.Sc. degree in computer science from University of Western Sydney, Sydney, NSW, Australia,in 2003, and the Ph.D. degree in software engineering from Northumbria University, Newcastle upon Tyne, U.K., in 2008.

He is currently an ACM Senior Member and Full Professor of software engineering with the Jordan University of Science and Technology.

Dr. Aljawarneh has presented at and been on the organizing committees for a number of international conferences and is a Board Member of the International Community for the IEEE, ACM, Jordan ACM Chapter, ACS, etc. A number of his papers have been selected as "Best Papers" at conferences and journals. Also, he is an Associate Editor for *Computers and Electrical Engineering* (Elsevier).



Chang Choi (SM'16) received the B.S., M.S., and the Ph.D. degrees in computer engineering from Chosun University, Gwangju, South Korea, in 2005, 2007, and 2012, respectively.

He is currently a Research Professor with Chosun University. His research interests include intelligent information processing, semantic web, smart IoT system, and intelligent system security.

Dr. Choi was the recipient of the academic awards from the Graduate School of Chosun University in 2012. He was also the recipient of Korean Government Scholarship for graduate students (Ph.D. course) in 2008. He has served or is currently serving on the organizing or program committees of international conferences and workshops, such as ACM RACS, EAI BDTA, IE, ACM SAC, and IEEE CCNC/SeCHID. He was also a Guest Editor for high-profile journals, such as *Future Generation Computer Systems*, *Applied Soft Computing*, *Multimedia Tools and Applications*, *Journal of Ambient Intelligence and Humanized Computing*, *Concurrency and Computation: Practice and Experience*, and *Autosoft*.