IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 29, 2020, accepted September 7, 2020, date of publication September 10, 2020, date of current version September 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023084

Privacy-Preserving Deep Learning on Machine Learning as a Service–A Comprehensive Survey

HARRY CHANDRA TANUWIDJAJA[1], RAKYONG CHOI[1], SEUNGGEUN BAEK[2], AND KWANGJO KIM[1], (Member, IEEE)

[1]School of Computing, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea
[2]Graduate School of Information Security, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea

Corresponding author: Harry Chandra Tanuwidjaja (elevantista@kaist.ac.kr)

This work was supported in part by the Indonesia Endowment Fund for Education, Lembaga Pengelola Dana Pendidikan (LPDP) and Institute for Information and communications Technology Planning and Evaluation (IITP), grant funded by the Korea government, Ministry of Science and ICT (MSIT) (No. 2017-0-00555, Towards Provable-secure Multi-party Authenticated Key Exchange Protocol based on Lattices in a Quantum World and <Q|Crypton> No.2019-0-00033, Study on Quantum Security Evaluation of Cryptography based on Computational Quantum Complexity).

ABSTRACT The exponential growth of big data and deep learning has increased the data exchange traffic in society. Machine Learning as a Service, (MLaaS) which leverages deep learning techniques for predictive analytics to enhance decision-making, has become a hot commodity. However, the adoption of MLaaS introduces data privacy challenges for data owners and security challenges for deep learning model owners. Data owners are concerned about the safety and privacy of their data on MLaaS platforms, while MLaaS platform owners worry that their models could be stolen by adversaries who pose as clients. Consequently, Privacy-Preserving Deep Learning (PPDL) arises as a possible solution to this problem. Recently, several papers about PPDL for MLaaS have been published. However, to the best of our knowledge, no previous paper has summarized the existing literature on PPDL and its specific applicability to the MLaaS environment. In this paper, we present a comprehensive survey of privacy-preserving techniques, starting from classical privacy-preserving techniques to well-known deep learning techniques. Additionally, we present a detailed description of PPDL and address the issue of using PPDL for MLaaS. Furthermore, we undertake detailed comparisons between state-of-the-art PPDL methods. Subsequently, we classify an adversarial model on PPDL by highlighting possible PPDL attacks and their potential solutions. Ultimately, our paper serves as a single point of reference for detailed knowledge on PPDL and its applicability to MLaaS environments for both new and experienced researchers.

INDEX TERMS Machine Learning as a Service (MLaaS), privacy-preserving deep learning (PPDL), using PPDL for MLaaS, adversarial model on PPDL, PPDL attacks and solutions.

I. INTRODUCTION

In a business environment, prediction and decision-making are two important processes that require careful consideration. Good judgement can lead to large profits, but bad decisions can ruin everything. There was a hypothesis that a computer could help a user predict something or decide what next step should be taken. As Artificial Intelligence (AI) has grown dramatically, this plan is no longer considered impossible. AI has the ability to sense, understand, learn, and respond [2]. This solves the weaknesses of computers without these four abilities. Prediction, on the other hand, is a process of learning available information and then using that knowledge to generate new information that is not yet available. A Deep Learning (DL) algorithm is a type of AI that has the ability to interpret data like a human brain and can learn and classify objects. By leveraging the ability of deep learning, we can predict the future and make decisions based on the currently available information, which becomes our training data when we train the DL model. After the training process is completed, a prediction model is produced. Based on this model, predictions based on clients' data will be performed. That is how Machine Learning as a Service (MLaaS), a promising business opportunity, was born.

MLaaS is a service, which usually runs on a cloud platform, with the purpose is to provide prediction service to clients by utilizing machine learning [3]. The service runs on a cloud environment so that clients do not need to build

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

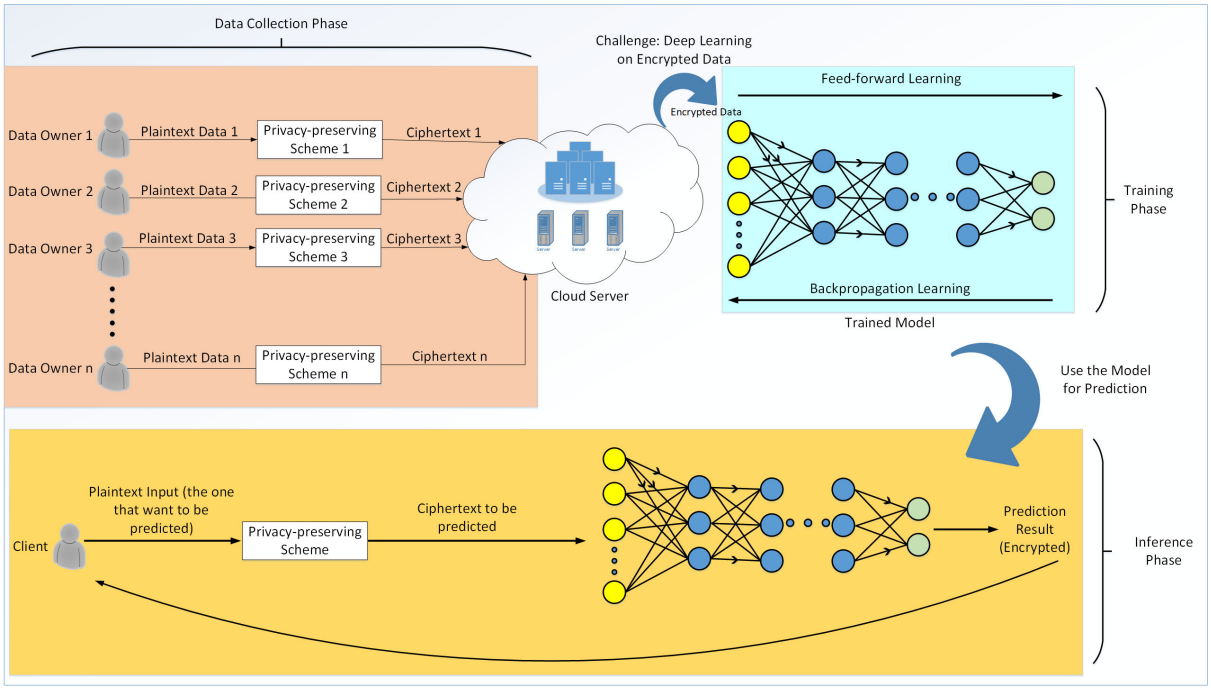

FIGURE 1. Illustration of PPDL Scheme.

their own machine learning model to do a prediction [4]. However, there is a problem. To perform predictions, a model owner needs to receive data from clients. The data may consist of sensitive information. Thus, clients are reluctant to provide their data. On the other hand, a model owner will also be worried that an adversary could be disguised as a client to try to steal the model. Furthermore, there is an issue about the privacy of the prediction result and whether will it be safe from access by unauthorized parties. In this scenario, Privacy-Preserving Deep Learning (PPDL) is needed as a solution. There are some previous works about Privacy-preserving (PP) surveys. However, none of them discusses PPDL for MLaaS.

- Mendes and Vilela [5] provided a comprehensive survey about privacy-preserving for data mining.
- Siduula *et al.* [6] performed a study on privacy-preserving for online social networks.
- Zhang *et al.* [7] discussed privacy-preserving in the edge computing paradigm.
- Domingo *et al.* [8] discussed privacy preserving in cloud computing.
- Rui and Yan [9] specifically discussed privacy-preserving in biometric authentication.
- Anand and Muthusamy [10] discussed issues of privacy-preserving in cloud computing.
- Tran and Hu [11] surveyed privacy-preserving for big data analytics.
- Zhen *et al.* [12] focused on the challenges of privacy-preserving machine learning in IoT.
- Riazi *et al.* [13] discussed deep learning in private data.
- Sultan *et al.* [14] discussed privacy-preserving metering in smart grid.
- Alvarez and Nojoumian [15] provided a survey on privacy-preserving protocol, specifically for sealed-bid auctions.

Compared with those papers, the main contribution of this paper can be summarized as follows:

- Our work covers the most recent and groundbreaking methods in PPDL, specifically for MLaaS.
- We propose a multi-scheme PPDL taxonomy that classifies adversarial models in PPDL, PP methods used in PPDL, and the challenges and weaknesses in state-of-the-art PPDL methods.
- We provide detailed comparisons of the surveyed PPDL works based on our defined metrics.
- We highlight the development of PPDL each year and summarize the main trend.
- To the best to our knowledge, this is the first PPDL survey paper to provides a complete discussion and analysis about PPDL on MLaaS.

This paper is an extension of our previous version [1] by improving the classification of PPDL into a wider scope, adding latest publications of 28 papers in total with our analysis of the weakness of each method. We present comparison tables (Tables 8 and 9), based on the privacy parameter and performance level, respectively. We also give more advanced discussion about adversarial model in PPDL, challenge and weakness, and also attacks on DL and PPDL as the possible solution.

In general, the PPDL scheme can be divided into three main phases: the data collection phase, training phase, and inference phase. Fig. 1 shows the illustration of PPDL as a whole. The data collection phase has the main

purpose of securely transporting data from the owner to the cloud. Data owner 1, data owner 2, data owner 3,... *etc.*, through data owner n encrypt their plaintext data using a privacy-preserving scheme into ciphertexts and send them to the cloud server. By doing this, the security of the data is guaranteed as only the data owner can see the true values of the data. After the data collection phase is completed, the next phase is the training phase. In the training phase, the encrypted data is used as the input of the deep learning model. The training process is divided into feed-forward learning and backpropagation learning. The feed-forward learning trains the model while backpropagation learning minimizes its error. After the training phase is completed, a prediction model is produced. This model will be used during the next phase, the inference phase. In the inference phase, clients who want to predict something send their queries to the cloud server. Using the prediction model, they obtain the prediction result. This concludes the general PPDL process.

The remainder of this paper is organized as follows: Section II discusses the classical privacy-preserving (PP) techniques, which include anonymization, cryptography, and differential privacy. Section III discusses deep learning in PP environments. Section IV discusses the limitations of applying novel DL techniques to PP. In Section V, we categorize the adversarial model in PPDL. The metrics used in this paper are described in Section VI. Then, in Section VII, we compare the state-of-the-art PPDL techniques. We present the possible attacks on DL model and the utilization of PPDL as the possible solution to overcome those attacks. In Section VIII, we discuss the challenge and weakness in utilizing PPDL for MLaaS. Finally, in Section IX, we present our conclusion and open issues for future research.

II. CLASSICAL PRIVACY-PRESERVING METHOD

We classify the classical PP method into three categories, as shown in Fig. 2. The three categories are group-based anonymity, cryptography method, and differential privacy.

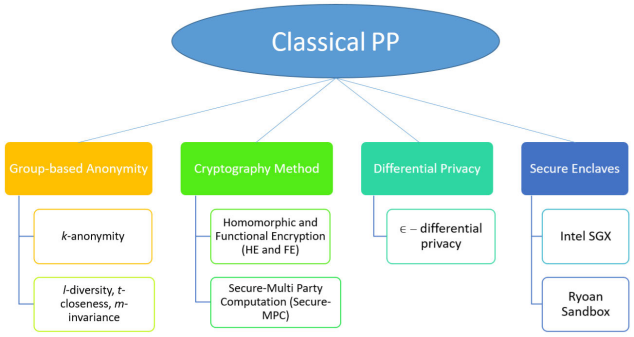

FIGURE 2. Classical PP Classification.

A. GROUP-BASED ANONYMITY

While homomorphic encryption, functional encryption, and secure multi-party computation techniques enable computation on encrypted data without revealing the original plaintext, we need to preserve the privacy of sensitive personal data such as medical and health data. One of the earliest milestones to preserving this privacy is to hide these sensitive personal data using data anonymization techniques.

The concept of k-anonymity was first introduced by Sweeney and Samarati [16] in 1998 to solve the problem: "Given sensitive personal data, produce the modified data which remains useful while the data cannot specify the corresponding person." Modified data are said to have k-anonymity if the information for any person whose information is in the modified data cannot be distinguished from at least $k-1$ individuals in the modified data. While k-anonymity is a simple and promising approach for group-based anonymization, it is susceptible to attacks such as a homogeneity attack or background knowledge attack [17] when background knowledge is available to an attacker. To overcome these issues, there are many privacy definitions, such as l-diversity, t-closeness, and m-invariance [17]–[19]. The concept of l-diversity means that each equivalent class has at least l distinct values for each sensitive attribute, and t-closeness is a further refinement of l-diversity created by also maintaining the distribution of sensitive attributes.

B. CRYPTOGRAPHY METHOD

1) HOMOMORPHIC AND FUNCTIONAL ENCRYPTION

In 1978, Rivest *et al.* [20] questioned whether any encryption scheme can support computation of encrypted data without knowledge of the encrypted information. If some encryption scheme supports an operation $\circ$ on encrypted data $Enc(m_1 \circ m_2)$, this scheme is called Homomorphic Encryption (HE) on an operation $\circ$. Depending on the computation type that HE supports, it is called partially HE when it supports the specific computation on encrypted data and Fully HE (FHE) when it supports arbitrary computation. For example, the well-known RSA encryption [21] supports multiplication on encrypted data without decryption, therefore RSA encryption is called multiplicative HE. Likewise, a scheme is additive HE if it supports addition on encrypted data without decryption.

The design of FHE remained as an interesting open problem in cryptography for decades, until Gentry suggested the first FHE in 2009 [22]. Afterwards, there have been a number of studies of HE schemes based on lattices with Learning With Errors (LWE) and Ring Learning With Errors (Ring-LWE) problems and schemes over integers with the approximate Greatest Common Divisor (GCD) problem [23]–[32]. Earlier works focused on HE were impractical for implementation; however, there are currently many cryptographic algorithm tools that support HE efficiently, such as HElib, FHEW, and HEEAN [33]–[35].

Functional Encryption (FE) was proposed by Sahai and Waters [36] in 2005 and formalized by Boneh *et al.* [37] in 2011. Let a functionality $F : K \times X \rightarrow \{0,1\}^*$. The functionality F is a deterministic function over (K, X) that outputs $(0, 1)^*$ where K is the key space and the set X is the

plaintext space. We say a scheme is FE for a functionality F over (K, X) if it can calculate $F(k, x)$ given a ciphertext of $x \in X$ and a secret key sk_k for $k \in K$.

Predicate encryption [38] is a subclass of FE scheme with a polynomial-time predicate $P : K \times I \rightarrow \{0, 1\}$ where K is the key space, I is the index set, and the plaintext $x \in X$ is defined as (ind, m); X is the plaintext space, ind is an index, and m is the payload message. As an example, we can define FE functionality $F_{\mathsf{FE}}(k \in K, (\mathsf{ind}, m) \in X) = m$ or $\perp$ depending on whether the predicate $P(k, \mathsf{ind})$ is 1 or 0, respectively. Depending on the choice of the predicate, Identity-based Encryption (IBE) [39]–[45] and Attribute-based Encryption (ABE) [36], [46] are well-known examples of predicate encryption schemes.

Both FE and HE enable computation over encrypted data. The difference is that the computation output of FE is a plaintext, while the output of HE remains encrypted as HE evaluates the encrypted data without decryption. There is no need for a trusted authority within HE systems. Additionally, HE enables the evaluation of any circuit g over the encrypted data if sk_g is given, but FE enables the computation of only some functions.

2) SECURE MULTI-PARTY COMPUTATION

The purpose of Multi-party Computation (MPC) is to solve the problem of collaborative computing that keeps the privacy of an honest/dishonest user in a group without using any trusted third party. Formally, in MPC, for a given number of participants, $p_1, p_2, \cdots, p_n$, each participant has private data, $d_1, d_2, \cdots, d_n$, respectively. Then, participants want to compute the value of a public function f on those private data, $f(d_1, d_2, \cdots, d_n)$, while keeping their own inputs secret.

The concept of secure computation was formally introduced as secure two-party computation in 1986 by Yao [47] with the invention of Garbled Circuit (GC). Yao's GC requires only a constant number of communication rounds, and all functions are described as a Boolean circuit. To transfer the information obliviously, Oblivious Transfer (OT) is used. The OT protocol allows a receiver P_R to obliviously select i and receive a message m_i from a set of messages M that belong to a sender party P_S. P_R does not know the other messages in M while P_S does not know the selected message.

Secret sharing is yet another building block for secure MPC protocols, *e.g.*, Goldreich *et al.* [48] suggested a simple and interactive secure MPC protocol using the secret-shared values to compute the value. Secret sharing is a cryptographic algorithm where a secret is parted and distributed to each participant. To reconstruct the original value, a minimum number of secret-shared values are required.

Compared with HE and FE schemes, in secure MPC, parties jointly compute a function on their inputs using a protocol instead of a single party. During the process, information about parties' secret must not be leaked. In secure MPC, each party has almost no computational cost with a huge communication cost, whereas the server has a huge computational cost with almost no communication cost in the HE scheme. The parties encrypt their data and send them to the server. The server computes the inner product between the data and the weight value of the first layer and sends the computation result back to the parties. Then, the parties decrypt the results and compute the non-linear transformation. The result is encrypted and transmitted again to the server. This process continues until the last layer has been computed. To apply secure MPC to deep learning, we must handle the communication cost as it requires many rounds of communication between the parties and the server, which is non-negligible.

C. DIFFERENTIAL PRIVACY

Differential privacy (DP) was first proposed by Dwork *et al.* [49] in 2006 as a strong standard to guarantee the privacy of the data. A randomized algorithm A gives ϵ-differential privacy if for all datasets D_1 and D_2 differ in at most one element, and for all subsets $S \in Range(imA)$, where imA denotes the image of A, such that

$$\Pr[\mathcal{A}(D_1) \in S] \leq \exp(\epsilon) \cdot \Pr[\mathcal{A}(D_2) \in S].$$

Differential privacy addresses when a trusted data manager wants to release some statistics on the data while the adversary cannot reveal whether some individual's information is used in the computation. Thus, differentially private algorithms probably resist identification and reidentification attacks.

An example of the latest implementation of differential privacy technique was proposed by Qi *et al.* [50]. They suggested a privacy-preserving method for a recommender system using Locality-Sensitive Hashing (LSH) technique, which is more likely to assign two neighboring points to the same label. As a result, sensitive data can be converted into less sensitive ones.

D. SECURE ENCLAVES

Secure enclaves, also known as Trusted Execution Environments (TEEs), are a secure hardware method that provides enclaves to protect code and data from another software on the related platform, including the operating system and hypervisor [51]. The concept of an enclave was firstly introduced by Intel [52], which introduced Software Guard Extensions (SGX), available on Intel processors starting from the Skylake generation [53]. Utilizing only SGX for privacy-preserving is not sufficient from the security and privacy perspectives because the code from the MLaaS provider is not trusted. SGX only protects the execution of trusted code on an untrusted platform. A code is called trusted if it is public and users can inspect the code. If the code is private, users cannot be assured that the code does not steal their data. Because of this, SGX needs to be confined to a sandbox to prevent data exfiltration. The most widely used sandbox for SGX is Ryoan [54]. The Ryoan sandbox also enables users to verify that the enclave executes standard ML code without seeing the model specifications. As a result, a combination of

the SGX and Ryoan sandboxes can guarantee the privacy of both clients and ML models.

III. DEEP LEARNING FOR PRIVACY-PRESERVING

PPDL is a development from the classical DL method. It combines the classical PP method with the emerging DL field. DL itself is a sub-class of machine learning the structure and functionality of that resemble a human brain. The structure of a deep learning model is modelled like a layered architecture. It starts from an input layer and ends with an output layer. Between an input layer and an output layer, there can be one or more hidden layers. The more hidden layers are used, the more accurate the DL model becomes. This is caused by the characteristic of a hidden layer. The output of one hidden layer will become the input of the next hidden layer. If we use more hidden layers, the deeper hidden layer will learn about more specific features. There are several DL methods that are widely used for PP. Based on our research, the most popular DL methods for PP are the Deep Neural Network (DNN), Convolutional Neural Network (CNN), and Generative Adversarial Network (GAN).

A. DEEP NEURAL NETWORK (DNN)

There are several commonly used layers in DNN, including the activation layer, pooling layer, fully connected layer, and dropout layer.

1) ACTIVATION LAYER

The activation layer, as shown in Fig. 3, decides whether the data is activated (value one) or not (value zero). The activation layer is a non-linear function that applies a mathematical process on the output of a convolutional layer. There are several well-known activation functions, such as Rectified Linear Unit (ReLU), sigmoid, and tanh. Because those functions are not linear, the complexity becomes really high if we use the functions to compute the homomorphically encrypted data. Hence, we need to find a replacement function that only contains multiplication and addition operations. The replacement function will be discussed later.

FIGURE 3. Activation Layer.

2) POOLING LAYER

A pooling layer, as shown in Fig. 4, is a sampling layer with the purpose of reducing the size of the data. There are two kinds of pooling: max and average pooling. In HE, we cannot use a max pooling function because we cannot search for the maximum value of encrypted data. As a result, average pooling is the solution to be implemented in HE.

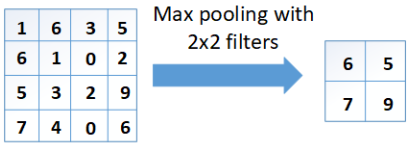

FIGURE 4. Pooling Layer.

Average pooling calculates the sum of values; thus, there is only the addition operation here, which can be used over homomorphically encrypted data.

3) FULLY CONNECTED LAYER

An illustration of a fully connected layer is shown in Fig. 5. Each neuron in this layer is connected to a neuron in the previous layer; thus, it is called a fully connected layer. The connection represents the weight of the feature like a complete binary graph. The operation in this layer is the dot product between the value of the output neuron from the previous layer and the weight of the neuron. This function is similar to a hidden layer in a Neural Network (NN). There is only a dot product function that consists of multiplication and addition function; thus, we can use it over homomorphically encrypted data.

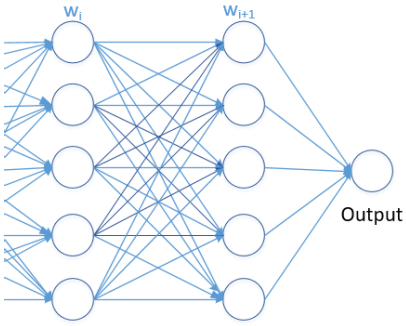

FIGURE 5. Fully Connected Layer.

4) DROPOUT LAYER

A dropout layer, shown in Fig. 6, is a layer created to solve an over-fitting problem. Sometimes, when we train our machine learning model, the classification result will be too good for some kind of data, showing bias based on the training set. This situation is not ideal, resulting in a large error during the testing period. The dropout layer will drop random data during training and set the data to zero. By doing this iteratively

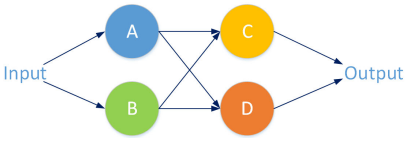

FIGURE 6. Dropout Layer.

during the training period, we can prevent over-fitting during the training phase.

B. CONVOLUTIONAL NEURAL NETWORK (CNN)

CNN [55] is a class of DNN usually used for image classification. The characteristic of CNN is a convolutional layer, as shown in Fig. 7, the purpose of which is to learn features extracted from the dataset. The convolutional layer has $n \times n$ size, on which we will perform a dot product between neighboring values to make a convolution. As a result, only addition and multiplication occurs in the convolutional layer. We do not need to modify this layer as it can be used for HE data, which are homomorphically encrypted. Table 1 shows the commonly used layers in DNN and CNN models.

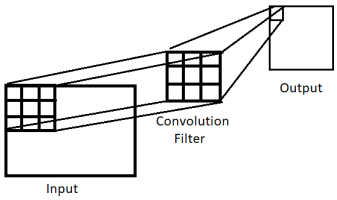

FIGURE 7. Convolutional Layer.

TABLE 1. Commonly Used Layers in DNN and CNN models.

Deep Learning Layer		Description	Function
Activation Function		ReLu	Maximum
		Sigmoid	Hyperbolic
		Tanh	Trigonometric
		Softmax	Hyperbolic
Pooling	Max Pooling	Computing the maximum value of overlapping region in the preceding layer	Maximum
	Mean Pooling	Computing the average value of non-overlapping region in the preceding layer	Mean
Fully Connected		Dot product between the value of output neuron from the previous layer and the weight of the neuron	Matrix-vector multiplication
Dropout		Set random data to zero during training to prevent overfitting	Drop
Convolutional		Dot product between neighbor values in order to make convolution, then sum up the result	Weighted Sum

C. GENERATIVE ADVERSARIAL NETWORK (GAN)

GAN [56] is a class of DNN usually used for unsupervised learning. GAN, as shown in Fig. 8, consists of two NNs that generate a candidate model and an evaluation model in a zero-sum game framework. The generative model will learn samples from a dataset until it reaches a certain accuracy. On the other hand, the evaluation model discriminates between the true data and the generated candidate model. GAN learns the process by modeling the distribution of individual classes.

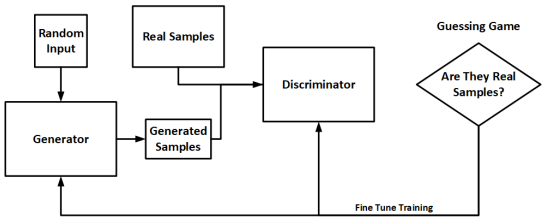

FIGURE 8. GAN Structure.

IV. LIMITATION OF IMPLEMENTING NOVEL DL TECHNIQUES TO PP

During our studies, we found some incompatibilities between DL structures and classical PP techniques. Modifications had to be made to combine the DL structures and PP techniques. We cover the three most widely required modifications, as shown in Fig. 9, including the batch normalization layer, an approximation of activation function, and the convolutional layer with increased stride.

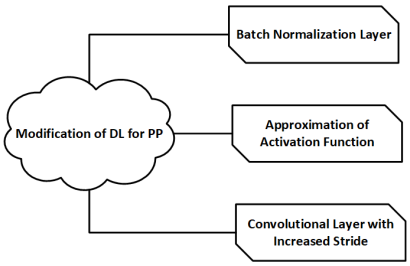

FIGURE 9. Required Modification for PPDL.

A. BATCH NORMALIZATION LAYER

The Batch Normalization (BN) layer was proposed by Ioffe and Szegedy [57]. The main purpose of the BN layer is to accelerate the training process by increasing the stability of the NN. This layer receives the output from the activation layer and then performs the re-scaling process, resulting in a value between zero and one. The BN layer computes the subtraction of each input with the batch mean value, and then divides it by the average value of the batch.

B. APPROXIMATION OF ACTIVATION FUNCTION

Several studies [58]–[60] have performed polynomial approximations for the activation function. Some well-known methods include numerical analysis, Taylor series, and establishing polynomials based on the derivative of the activation function. Numerical analysis generates some points from the ReLU function and then uses the points as the inputs of the approximation function. The Taylor series uses polynomials of different degrees to approximate the activation function.

C. CONVOLUTIONAL LAYER WITH INCREASED STRIDE

This architecture was proposed by Liu *et al.* [60] to replace the pooling layer. The architecture leverages a convolutional layer with increased stride as a substitution of the pooling layer. The BN layer is used between the fully connected layer and ReLU. By doing this, the depth of the data stays the same, but the dimension is reduced [57].

V. ADVERSARIAL MODEL IN PPDL

We categorize the adversarial model in PPDL based on the adversary's behavior, adversary's power, and adversary's corruption types as shown in Fig. 10.

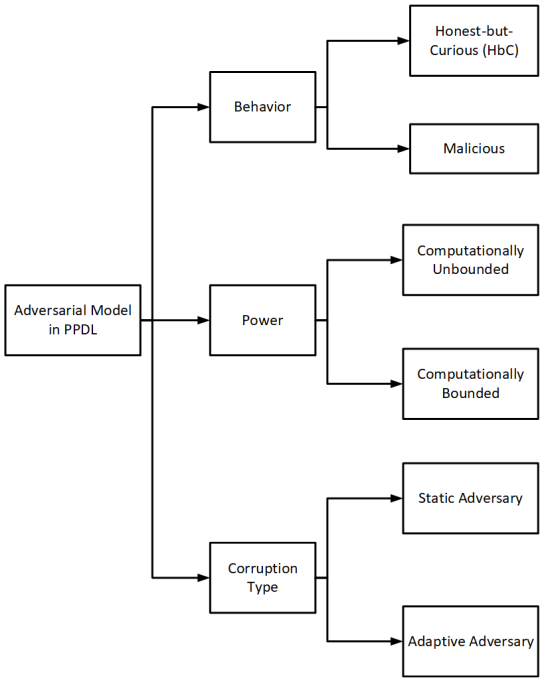

FIGURE 10. Adversarial Model in PPDL.

A. ADVERSARIAL MODEL BASED ON THE BEHAVIOR

We categorize the adversarial model based on the behavior into honest-but-curious and malicious.

1) HONEST-BUT-CURIOUS

In an Honest-but-Curious (HbC) adversary model, all parties, including the corrupted party, follow the security protocol honestly. They do not pursue any malicious activity toward the system or other participants. However, the corrupted party tries to perform a "curious action" to learn sensitive information from the model or from the other participants. This model is the one most commonly used in PPDL.

2) MALICIOUS

This scenario is also known as the active adversary model because the corrupted parties will actively try to attack even if they must deviate from the existing security protocol. If the corrupted parties can prematurely halt their attacking process, sometimes the model is also recognized as a fail stop model.

B. ADVERSARIAL MODEL BASED ON THE POWER

We categorize adversarial model based on the behavior into computationally unbounded and computationally bounded.

1) COMPUTATIONALLY UNBOUNDED

This means that the adversary has unlimited computational power. As a result, it is considered as the ideal adversary. It is usually used in theoretical information security field as it does not exist in real life.

2) COMPUTATIONALLY BOUNDED

This means that the adversary has limited computational power. Usually, it requires cryptographic assumption. The time assumption during the attack process is defined as the polynomial time.

C. ADVERSARIAL MODEL BASED ON CORRUPTION TYPE

We categorize the adversarial model based on the corruption type into static adversary and adaptive adversary.

1) STATIC ADVERSARY

In this model, the corrupted parties are defined before the protocol starts. An honest party will always stay honest, and a corrupted party will always stay corrupted.

2) ADAPTIVE ADVERSARY

In this model, an adversary will decide which party to corrupt based on the current situation. As a result, an honest party can become corrupted in the middle of protocol execution. However, in the adaptive model, an adversary can change the corrupted party such that the corrupted party can become honest again. This is classified as an adaptive-mobile adversary.

VI. COMPARISON METRICS

To compare the performances of each surveyed article, we used two kinds of metrics, qualitative metrics and quantitative metrics. Fig. 11 shows the metrics for the surveyed PPDL works in this paper. Qualitative metrics include Privacy of Client (PoC), Privacy of Model (PoM), and Privacy of Result (PoR). PoC means that neither the model owner nor

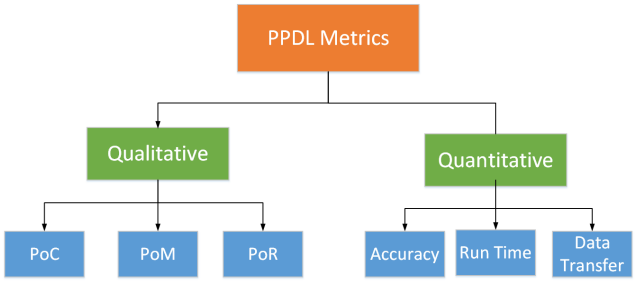

FIGURE 11. Metrics for Surveyed PPDL Works.

the cloud server or any other party knows about the client data. PoM means that neither the client nor the cloud server or any other party knows about the DL model. PoR means that neither the model owner nor the cloud server or any other party can obtain the information about the prediction result. Quantitative metrics include accuracy and inference time. Accuracy means the percentage of correct predictions made by a PPDL model. The inference time is the time needed by the model to perform encryption/decryption, send data from the client to the server, and execute the classification process. We measured the average accuracy and inference time of each method. Then, we set the average value as the relative evaluation. If the accuracy value is higher than average, the accuracy of the proposed method is good. Furthermore, if the run time and data transfer are lower than average, the run time and data transfer of the proposed methods are good. We used the comparison data from the respective papers as we believe they are the best result to be achieved. We did not re-execute their codes as not all of the codes are open to the public. We focused our paper on the Hybrid PPDL method, which combines classical privacy-preserving with various deep learning practices.

VII. STATE-OF-THE-ART PPDL METHODS

As shown in Fig. 12, we classified each PPDL method by its privacy-preserving techniques: HE-based PPDL, secure MPC-based PPDL, differential privacy-based PPDL, secure enclaves-based PPDL, and hybrid-based PPDL. Hybrid-based PPDL means that the PPDL method combines more than one privacy-preserving technique mentioned before.

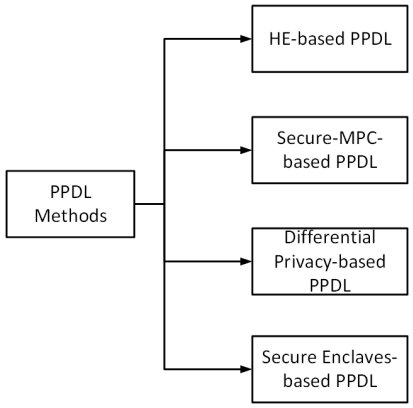

FIGURE 12. Classification of PPDL Methods by Its Privacy Preserving Techniques.

We have surveyed several key publications on PPDL per each year since 2016 as shown in Fig. 13.

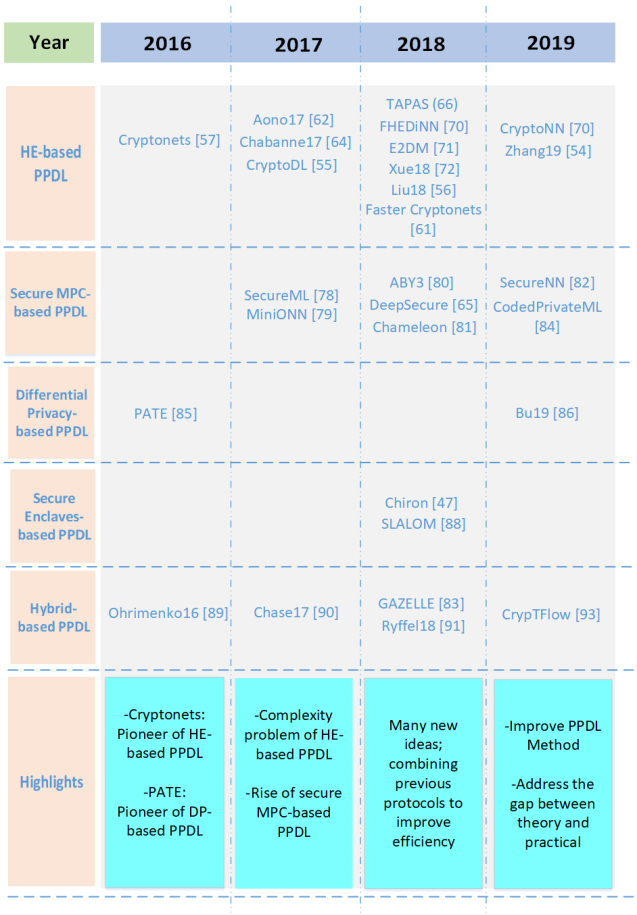

FIGURE 13. The Surveyed Paper of PPDL Since 2016.

A. HE-BASED PPDL

HE-based PPDL combines homomorphic encryption with deep learning. The structure of HE-based PPDL is shown in Fig. 14. Generally, there are three phases in HE-based PPDL: the training phase (T1-T2-T3-T4), inference phase (I1-I2-I3)), and result phase (R1-R2-R3). In the training phase, a client encrypts the training dataset using HE (T1) and sends the encrypted dataset to the cloud server (T2). In the cloud server, secure training is executed (T3), resulting in a trained model (T4). This is the end of the training phase. For the inference phase, the client sends the testing dataset to the cloud server (I1). The testing dataset becomes the input of the trained model (I2). Then, the prediction process is run using the trained model (I3), resulting in an encrypted computation result. This is the end of the inference phase. Next, the cloud server prepares to transport the encrypted computation result (R1) and sends it to the client (R2). The client finally decrypts it and obtains its computation result (R3).

1) YEAR 2016

Cryptonets was proposed by Gilad-Bachrach *et al.* [61] to address the privacy issue in Machine Learning as a Service (MLaaS). The author combined cryptography and machine learning to present a machine learning framework that can receive encrypted data as an input. Cryptonets improves the performance of ML Confidential [62] developed by Graepel *et al.*, a modified PPDL scheme based on Linear Means Classifier [63] and Fisher Linear Discriminant [64] that works on HE. ML Confidential uses polynomial

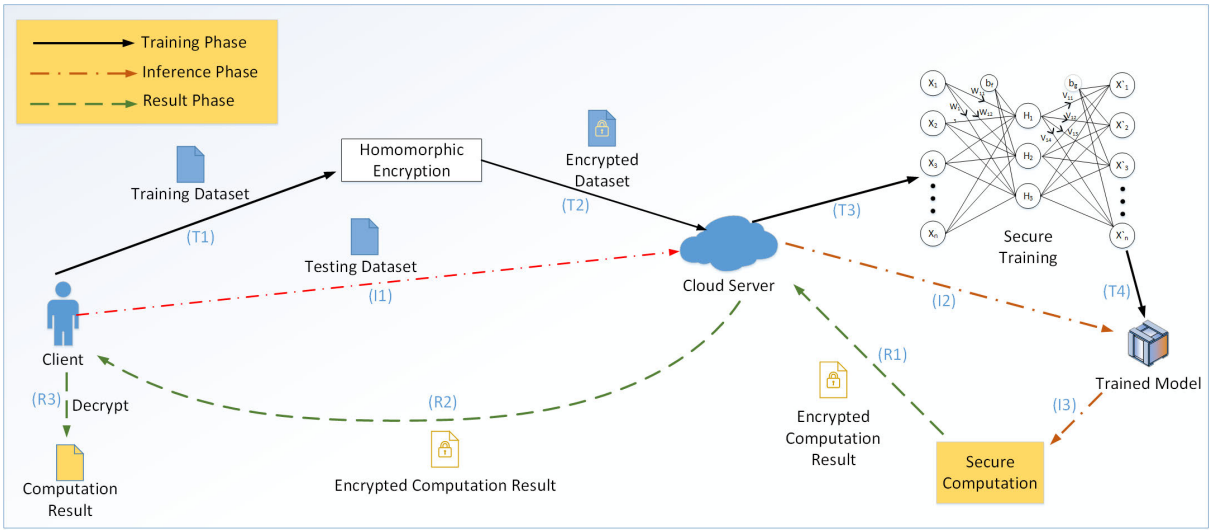

FIGURE 14. The Structure of HE-based PPDL.

approximation to substitute for the nonlinear activation function. In this case, the PoM is not guaranteed because the client must generate the encryption parameter based on the model. ML Confidential uses a cloud service-based scenario, and its main feature is ensuring the privacy of data during the transfer period between the client and the server. At first, the cloud server produces a public key and its private key for each client. Then, the client data are encrypted using HE and transferred to the server. The cloud server will perform the training process using the encrypted data and use the training model to perform classification on the testing dataset.

Cryptonets applies prediction based on encrypted data and then provides the prediction result, also in encrypted form, to users. Later, users can use their private key to decrypt the prediction result. By doing this, the privacy of the client and the privacy of the result are guaranteed. However, the privacy of model is not guaranteed because the client must generate an encryption parameter based on the model. The weakness of Cryptonets is the performance limitation because of the complexity issue. It does not work well on deeper NNs that have a large number of non-linear layers. In this case, the accuracy will decrease and the error rate will increase.

Cryptonets has trade-off between accuracy and privacy. This is caused by the utilization of activation function approximation using low-degree polynomial during the training phase. The neural network needs to be retrained again using plaintext with the same activation function in order to achieve good accuracy. Another weakness of Cryptonets is the limited number of neural network layer. The multiplicative leveled HE cannot be run on deep neural network with many layers. Faster Cryptonets [65] accelerates homomorphic evaluation in Cryptonets [61] by pruning network parameter such that many multiplication operations can be omitted. The main weakness of Faster Cryptonets is that it has vulnerability to membership inference attack [66] and model stealing attack [67].

2) YEAR 2017

Aono17 [68] is a PPDL system based on a simple NN structure. The author shows a weakness in the paper by Shokri and Shmatikov [69] that leaks client data during the training process. The weakness is called Gradients Leak Information. It is an adversarial method for obtaining input values by calculating the gradient of the corresponding truth function to weight and the gradient of the corresponding of truth function to bias. If we divide the two results, we obtain the input value. Because of that reason, Aono17 [68] proposes a revised PPDL method to overcome this weakness. The key idea is allowing the cloud server to update the deep learning model by accumulating gradient values from users. The author also utilized additively HE to protect gradient values against curious servers. However, a weakness actually remains in this approach because it does not prevent attacks between participants. Proper authentication of participants should be performed by the cloud server to prevent this vulnerability. This method is able to prevent data leakage by encrypting the gradient value. However, it has some limitations as the homomorphic encryption is compatible with parameter server only.

Chabanne17 [70] is a privacy-preserving scheme on DNN. The scheme is a combination of HE and CNN. The main idea is to combine Cryptonets [61] with polynominal approximation for the activation function and batch normalization layer proposed by Ioffe and Szegedy [57]. The scheme wants to improve the performance of Cryptonets, which is only good when the number of non-linear layers in the model is small. The main idea is to change the structure of the regular NN by adding a batch normalization layer between the pooling layer and activation layer. Max pooling is not a linear function. As a result, in pooling layers average pooling is used instead of max pooling to provide the homomorphic part with a linear function. The batch normalization layer contributes to restricting the input of each activation layer, resulting in

a stable distribution. Polynomial approximation with a low degree gives a small error, which is very suitable for use in this model. The training phase is performed using the regular activation function, and the testing phase is performed using the polynomial approximation as a substitution to non-linear activation function. Chabanne17 showed that their model achieved 99.30% accuracy, which is better than that of Cryptonets (98.95%). The pros of this model is its ability to work in a NN with a high number of non-linear layers while still providing higher than 99% accuracy, unlike Cryptonets which exhibits a decrease in accuracy when the number of non-linear layers is increased. Chabanne17's weakness is that the classification accuracy relies on the approximation of activation function. If the approximation function has high degree, it will be hard to get best approximation so that the accuracy will decrease.

CryptoDL [59], proposed by Hesamifard *et al.*, is a modified CNN for encrypted data. The activation function part of CNN is substituted with a low-degree polynomial. That paper showed that the polynomial approximation is indispensable for NN in HE environments. The authors tried to approximate three kinds of activation functions: ReLU, sigmoid, and tanh. The approximation technique is based on the derivative of the activation function. First, during the training phase, CNN with polynomial approximation is used. Then, the model produced during the training phase is used to perform classification over encrypted data. The authors applied the CryptoDL scheme to the MNIST dataset and achieved 99.52% accuracy. The weakness of this scheme is not covering privacy-preserving training in DNN. The privacy-preserving is only applied for the classification process. The advantage of this work is that it can classify many instances (8,192 or larger) for each prediction round, whereas DeepSecure [71] classifies one instance per round. Hence, we can say that CryptoDL works more effectively than DeepSecure. The weakness of CryptoDL is claimed to be the limited number of layers in DNN. Since as the number of layer increases, the complexity is also increased multiplicatively due to HE operations, reducing its performance like Cryptonets [61].

3) YEAR 2018

In TAPAS [72], the author addresses the weakness of Fully Homomorphic Encryption in PPDL, which requires a large amount of time to evaluate deep learning models for encrypted data [31]. The author developed a deep learning architecture that consists of a fully-connected layer, a convolutional layer, and a batch normalized layer [57] with sparsified encrypted computation to reduce the computation time. The main contribution here is a new algorithm to accelerate binary computation in the binary neural network [73], [74]. Another superiority of TAPAS is supporting parallel computing. The technique can be parallelized by evaluating gates in the same level at the same time. A serious limitation of TAPAS is that it only supports binary neural network. In order to overcome this limitation, a method to encrypt non-binary or real-valued neural network is required.

FHE DiNN [75] is a PPDL framework that combines FHE with a discretized neural network. It addresses the complexity problem of HE in PPDL. FHE-DiNN offers a NN with linear complexity with regard to the depth of the network. In other words, FHE-DiNN has the scale invariance property. Linearity is achieved by the bootstrapping procedure on a discretized NN with a weighted sum and a sign activation function that has a value between -1 and 1. The sign activation function will maintain linearity growth such that it will not be out of control. The computation of the activation function will be performed during the bootstrapping procedure to refresh the ciphertext, reducing its cumulative noise. When we compare the discretized neural network to a standard NN, there is one main difference: the weight, the bias value, and the domain of the activation function in FHE DiNN needs to be discretized. The sign activation function is used to limit the growth of the signal in the range of -1 and 1, showing its characteristic of linear scale invariance for linear complexity. Compared with Cryptonets [61], FHE DiNN successfully improves the speed and reduces the complexity of FHE but with a decrease in accuracy; thus, a trade-off exists. The weakness of this method happens in the discretization process, which uses sign activation function that leads to a decrease in accuracy. It gets better if the training process is directly executed in a discretized neural network, rather than by converting a regular network into a discretized one.

E2DM [76] converts an image dataset into matrices. The main purpose of doing this is to reduce the computational complexity. E2DM shows how to encrypt multiple matrices into a single ciphertext. It extends some basic matrix operations such as rectangular multiplication and transposition for advanced operations. Not only is the data encrypted; the model is also homomorphically encrypted. As a result, PoC and PoM are guaranteed. E2DM also fulfills the PoR as only the client can decrypt the prediction result. For the deep learning part, E2DM utilizes CNN with one convolutional layer, two fully connected layers, and a square activation function. The weakness of E2DM is that it can only support simple matrix operation. Extending the advanced matrix computation will be a promising future work.

Xue18 [77] tries to enhance the scalability of the current PPDL method. A PPDL framework with multi-key HE was proposed. Its main purpose [77] was to provide a service to classify large-scale distributed data. For example, in the case of predicting road conditions, the NN model must be trained from traffic information data from many drivers. For the deep learning structure, [77] modification to the conventional CNN architecture is necessary, such as changing max pooling into average pooling, adding a batch normalization layer before each activation function layer, and replacing The ReLU activation function with a low-degree approximation polynomial. PoC and PoR are guaranteed here. However, the privacy of the model is not guaranteed because the client must generate an encryption parameter based on the model. The weakness of this approach is that the neural network must be trained by using encrypted data during the training phase. So, privacy

leakage may happen if appropriate countermeasure is not deployed.

Liu18 [60] is a privacy-preserving technique for convolutional networks using HE. The technique uses an MNIST dataset that contains handwritten numbers. Liu18 [60] encrypts the data using HE and then uses the encrypted data to train CNN. Later, the classification and testing process is performed using the model from CNN. The idea is to add a batch normalization layer before each activation layer and approximate activation layer using Gaussian distribution and the Taylor series. The non-linear pooling layer is substituted for with the convolutional layer with increased stride. By doing this, the author successfully modified CNN to be compatible with HE, achieving 98.97% accuracy during the testing phase. The main difference between regular CNN and modified CNN in privacy-preserving technology is the addition of the batch normalization layer and the change of the non-linear function in the activation layer and the pooling layer into a linear function. The proposed approach has weakness from the point of complexity since the HE has massive computational overhead leading to huge memory overhead.

4) YEAR 2019

CryptoNN [78] is a privacy-preserving method that utilizes functional encryption for arithmetic computation over encrypted data. The FE scheme protects the data in the shape of a feature vector inside matrices. By doing this, the matrix computation for NN training can be performed in encrypted form. The training phase of CryptoNN comprises two main steps: a secure feed-forward step and a secure back-propagation step. The CNN model is adapted with five main functions: a dot-product function, weighted-sum function, pooling function, activation function, and cost function. During the feed-forward phase, the multiplication of the weight value and feature vector cannot be performed directly because the vector value is encrypted. As a result, a function-derived key is used to transform the weight value such that it can computed. However, the scalability of CryptoNN is still in question since the dataset used in their experiment is a simple one. It needs to be tested with more complex dataset and deeper neural network model.

Zhang19 [58] is a secure clustering method for preserving data privacy in cloud computing. The method combines a probabilistic C-Means algorithm [79] with a BGV encryption scheme [12] to produce HE-based big data clustering on a cloud environment. The main reason for choosing BGV in this scheme is its ability to ensure a correct result on the computation of encrypted data. The author also addresses the weakness of the probabilistic C-Means algorithm, which is very sensitive and needs to be initialized properly. To solve this problem, fuzzy clustering [80] and probabilistic clustering [81] are combined. During the training process, there are two main steps: calculating the weight value and updating the matrix. To this end, a Taylor approximation for the activation function is used as the function is polynomial with addition and multiplication operations only. The main weakness is that the computation cost will increase proportionally to the number of neural network layers due to characteristic of HE.

According to Boulemtafes *et al.* [82], based on its learning method, PPDL techniques can be classified into two kinds; server-based and server-assisted. Server-based means that the learning process is executed on the cloud server. On the other hand, server-assisted means that the learning process is performed collaboratively by the parties and the server. Table 2 shows the features of our surveyed HE-based PPDL.

TABLE 2. Features of Our Surveyed HE-based PPDL.

References	Key Concept	Learning Type	Dataset
Cryptonets [61]	Enables cloud-based NN training using polynomial approximation of activation function	Server-based	MNIST
Aono17 [68]	Enables collaborative learning between participants over combined datasets	Server-assisted	MNIST
Chabanne17 [70]	Applies FHE with low degree approximation of activation function	Server-based	MNIST
CryptoDL [59]	Applies leveled HE with approximation of activation function based on the derivative of the function	Server-based	MNIST CIFAR-10
TAPAS [72]	Proposes a sparsified encrypted computation to speed up the computation in binary neural network	Server-based	MNIST
FHEDiNN [75]	Applies FHE in discretized neural network with linear complexity, in regards to the network's depth	Server-based	MNIST
E2DM [76]	Proposes multiple matrices encryption into a single chipertext to reduce computational complexity	Server-based	MNIST
Xue18 [77]	Applies multi-key HE with batch normalization layer before each activation function layer	Server-based	MNIST
Liu18 [60]	Proposes the addition of batch normalization layer and approximates activation function using Gaussian distribution and Taylor series	Server-based	MNIST
Faster Cryptonets [65]	Accelerates homomorphic evaluation by pruning the network parameters	Server-based	MNIST
CryptoNN [78]	Utilizes FE for arithmetic computation over encrypted data	Server-based	MNIST
Zhang19 [58]	Combines probabilistic C-Means algorithm with BGV encryption for big data clustering in cloud environment	Server-based	eGSAD Swsn

B. SECURE MPC-BASED PPDL

Generally, the structure of a secure MPC-based PPDL is shown in Fig. 15. Firstly, users perform local training using their private data (1). Then, the gradient result from the training process is secret-shared (2). The shared gradient is transmitted to each server (3). After that, the server aggregates the shared gradient value from users (4). The aggregated gradient

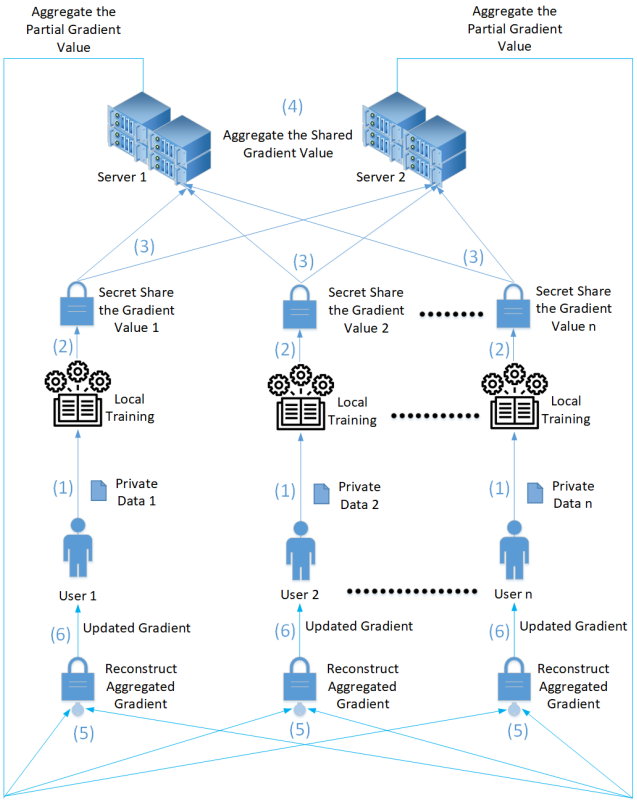

FIGURE 15. The Structure of Secure MPC-based PPDL.

value is transmitted from each server to each client (5). Each client reconstructs the aggregated gradient and updates the gradient value for the next training process (6). In the case of multi-party computation, secret sharing is used to preserve the data privacy. However, for specific secure two-party computation, a garbled circuit with secret sharing is widely used instead of secret sharing.

The structure of secure two-party computation is shown in Fig. 16. In secure two-party computation, a client uses garbled circuit to protect the data privacy. The communication between the client and the server is securely guaranteed by using oblivious transfer. At first, a client sends the private data input to the garbled circuit for the garbling process (1). Then, the next process is the data exchange between the client and the server using oblivious transfer (2). After the data exchange has been completed, the server runs the prediction process, using the data as an input in the deep learning model (3). The prediction result is sent back to the client. The client uses the garbled table to aggregate the result (4) and obtain the final output (5).

1) YEAR 2017

SecureML [83] is a new protocol for privacy-preserving machine learning. The protocol uses Oblivious Transfer (OT), Yao's GC, and Secret Sharing to ensure the privacy of the system. For the deep learning part, it leverages linear regression and logistic regression in a DNN environment.

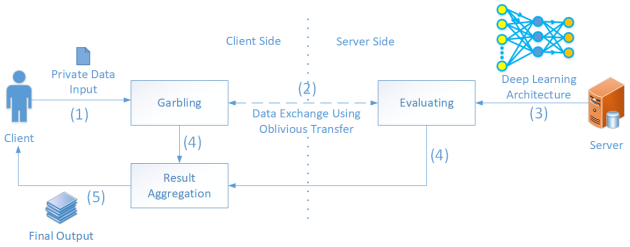

FIGURE 16. The Structure of Secure Two-party Computation-based PPDL.

The protocol proposes an addition and multiplication algorithm for secretly shared values in the linear regression. The Stochastic Gradient Descent (SGD) method is utilized to calculate the optimum value of regression. The weakness of this scheme is that it can only implement a simple NN without any convolutional layer; thus, the accuracy is quite low. The weakness of SecureML relies on the non-colluding assumption. In the two-servers model, the servers can be untrusted but not collude with each other. If the servers may collude, the privacy of participants can be compromised.

MiniONN [84] is a privacy-preserving framework for transforming a NN into an oblivious Neural Network. The transformation process in MiniONN includes the nonlinear functions, with a price of negligible accuracy lost. There are two kinds of transformation provided by MiniONN, including oblivious transformation for the piecewise linear activation function and oblivious transformation for the smooth activation function. A smooth function can be transformed into a continuous polynomial by splitting the function into several parts. Then, for each part, polynomial approximation is used for the approximation, resulting in a piecewise linear function. Hence, MiniONN supports all activation functions that have either a monotonic range or piecewise polynomial or can be approximated into a polynomial function. The experiment showed that MiniONN outperforms Cryptonets [61] and SecureML [83] in terms of message size and latency. The main weakness is that MiniONN does not support batch processing. MiniONN is also based on honest-but-curious adversary, so it has no countermeasure against malicious adversary.

2) YEAR 2018

ABY3 [85] proposed by Mohassel *et al.*, is a protocol for privacy-preserving machine learning based on three-party computation (3PC). The main contribution of this protocol is its ability to switch among arithmetic, binary, and Yao's 3PC depending on the processing needs. The main purpose of ABY3 is to solve the classic PPDL problem that requires switching back and forth between arithmetic (for example addition and multiplication) and non-arithmetic operations (such as activation function approximation). The usual machine learning process works on arithmetic operations. As a result, it cannot perform a polynomial approximation for activation function. ABY3 can be used to train linear

regression, logistic regression, and NN models. Arithmetic sharing is used when training linear regression models. On the other hand, for computing logistic regression and NN models, binary sharing on three-party GC is utilized. The author also introduced a new fixed-point multiplication method for more than three-party computation, extending the 3PC scenario. This multiplication method is used to solve the limitation of using MPC with machine learning. MPC is suitable for working over rings, unlike machine learning which works on decimal values. ABY3 provides a new framework that is secure against malicious adversaries; so it is not limited to honest-but-curious adversary. However, since the protocols are built in their own framework, it will be difficult to be implemented with other deep learning scheme.

DeepSecure [71] is a framework that enables the use of deep learning in privacy-preserving environments. The author used OT and Yao's GC protocol [47] with CNN to perform the learning process. DeepSecure enables a collaboration between client and server to perform the learning process on a cloud server using data from the client. The security of the system was proven using an honest-but-curious adversary model. The GC protocol successfully keeps the client data private during the data transfer period. The weakness of this method is its limitation of number of instances processed each round. The method can only classify one instance during each prediction round. DeepSecure offers a preprocessing phase that reduces the size of data. The strength of DeepSecure is that the preprocessing phase can be adopted easily because it is independent from any cryptographic protocol. Its main weakness is the inability to process batch processing.

Chameleon [86] is a PPDL method that combines Secure-MPC and CNN. For the privacy part, Chameleon uses Yao's GC which enables two parties to perform joint computation without disclosing their own input. There are two phases: an online phase and offline phase. During the online phase all parties are allowed to communicate, whereas during the offline phase the cryptographic operations are precomputed. Chameleon utilizes vector multiplication (dot product) of signed fixed-point representation which improves the efficiency of heavy matrix multiplication for encrypted data classification. It successfully achieves faster execution compared with CryptoNets [61] and MiniONN [84]. Chameleon requires two non-colluding servers to ensure the data privacy and security. For the private inference, it requires an independent third party or a secure hardware such as Intel SGX. Chameleon is based on honest-but-curious adversary, there is no countermeasure against malicious adversary. Chameleon's protocol is based on two party computation, so it is not efficient to implement in more than two-party scenario.

3) YEAR 2019

SecureNN [87] provides the first system that ensures the privacy and correctness against honest-but-curious adversaries and malicious adversaries for complex NN computation. The system is based on secure MPC combined with CNN. SecureNN was tested on an MNIST dataset and successfully achieved more than 99% prediction accuracy with execution times 2-4 times faster than other secure MPC based PPDL, such as SecureML [83], MiniONN [84], Chameleon [86], and GAZELLE [88]. Its main contribution is developing a new protocol for Boolean computation (ReLU, Maxpool, and its derivatives) that has less communication overhead than Yao GC. This is how SecureNN achieves a faster execution time than the other techniques mentioned above. The weakness of SecureNN is claimed to refine more communication overhead compared to ABY3 [86]. If the SecureNN protocol is modified so that it utilizes matrix multiplication like ABY3, the number of communication rounds will be reduced.

CodedPrivateML [89] distributes the training computation across several stations and proposes a new approach for secret sharing of the data and DL model parameter that significantly reduces the computation overhead and complexity. However, the accuracy of this method is only about 95%, which is not as high as other method such as GAZELLE [88] or Chameleon [86].

Table 3 shows the features of our surveyed secure MPC-based PPDL.

TABLE 3. Features of Our Surveyed Secure MPC-based PPDL.

References	Key Concept	Learning Type	Dataset
SecureML [83]	Proposes a combination of garbled circuit with oblivious transfer and secret sharing in a DNN environment	Server-assisted	MNIST CIFAR-10
MiniONN [84]	Transforms a NN into an oblivious NN	Server-assisted	MNIST
ABY3 [85]	Provides an ability to switch between arithmetic, binary, and three-party computation	Server-assisted	MNIST
DeepSecure [71]	Enables a collaboration between client and server to do a learning process on cloud server	Server-assisted	MNIST
Chameleon [86]	Enables a secure joint computation with two distinguished phases; online and offline	Server-assisted	MNIST
SecureNN [87]	Develops a new protocol for Boolean computation that has small overhead	Server-assisted	MNIST
Coded PrivateML [89]	Proposes a distributed training computation across clients with a new approach of secret sharing	Server-assisted	MNIST

C. DIFFERENTIAL PRIVACY-BASED PPDL

The structure of differential privacy-based PPDL is shown in Fig. 17. First, training data are used to train the teacher model (1). Then, the teacher model is used to train the student model. In this case we illustrated the student model as a GAN model that consists of a generator and discriminator (2). Random noise is added to the generator as it generates fake training data (3). On the other hand, the teacher model trains

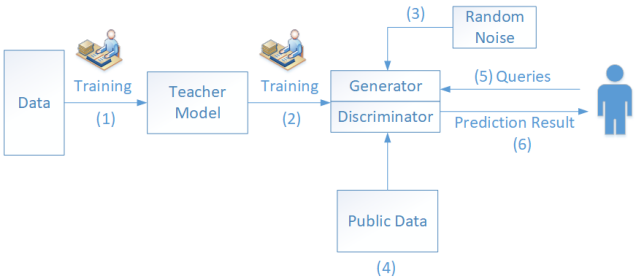

FIGURE 17. The Structure of Differential Privacy-based PPDL.

the student model using the public data (4). The student model runs a zero-sum game between the generator and the discriminator. Then, the student model is ready to be used for the prediction process. A client sends a query (5) to the student model. The student model runs the inference phase and returns the prediction result to the user (6).

Private Aggregation of Teacher Ensembles (PATE) [90] is a PPDL method for MLaaS that uses a differential privacy-based approach in Generative Adversarial Network (GAN). PATE is a black box approach that tries to ensure the privacy of data during training by using teacher-student models. During the training phase, the dataset is used to train the teacher models. Then, student models learn from the teacher models using a voting-based differential privacy method. By doing this, the teacher model is kept secretive, and the original data cannot be accessed by the student. The advantage of this model is due to the distinguished teacher model; when an adversary obtains a student model, the model will not give the adversary any confidential information. PATE has a serious weakness, which is not to provide good accuracy for complex data. If the data is too diverse, adding noise to the data will lower the performance of PATE. So, the performance of PATE depends on the type of input. It is only suitable for simple classification task. Furthermore, the computation cost is expensive due to many interactions between server and clients.

Another PPDL method that utilizes differential privacy is Bu19 [91]. Bu19 proposes Gaussian Differential Privacy (Gaussian DP) which formalizes the original DP technique as a hypothesis test from the adversaries' perspective.The concept of adding Gaussian noise is interesting. It must be evaluated in order to analyze the trade-off between the noise and the accuracy. The scalability issue implemented in the daily life remains in question.

Table 4 shows the features of our surveyed differential privacy-based PPDL.

TABLE 4. Features of Our Surveyed Differential Privacy-based PPDL.

References	Key Concept	Learning Type	Dataset
PATE [90]	Proposes a differentially private learning process by utilizing teacher models and student models	Server-based	MNIST SVHN
Bu19 [91]	Proposes a Gaussian differential privacy that formalizes the original differential privacy-based PPDL	Server-based	MNIST MovieLens

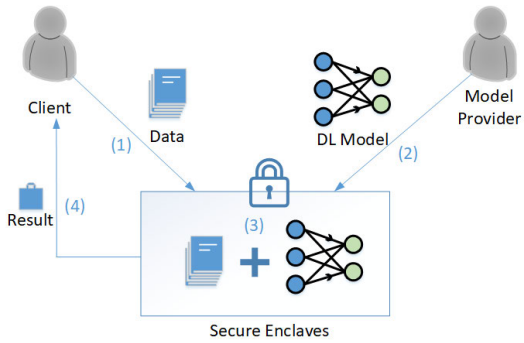

FIGURE 18. The Structure of Secure Enclaves-based PPDL.

D. SECURE ENCLAVES-BASED PPDL

The structure of secure enclaves-based PPDL is shown in Fig. 18. At first, a client sends data to the secure enclave environment (1). Then, the model provider sends the deep learning model to the enclaves (2). In the secure enclaves environment, the prediction process is executed using the client's data and the deep learning model (3). Then, the prediction result is sent to the client (4). The process in secure enclaves is guaranteed to be secure, and all of the data and models inside cannot be revealed to any other party outside the enclaves.

SLALOM [92] uses Trusted Execution Environments (TEEs), which isolate the computation process from untrusted software. The DNN computation is partitioned between trusted and untrusted parties. SLALOM runs DNN in the Intel SGX enclave which delegates the computation process to an untrusted GPU. The weakness of this approach is believed to limit CPU operation since the TEE does not allow to access GPU. A vulnerability by side channel attack may occur as shown by Van *et al.* [93].

Chiron [51] provides a black-box system for PPDL. The system conceals training data and model structure from the service provider. It utilizes SGX enclaves [94] and the Ryoan sandbox [54]. As SGX enclaves only protect the privacy of the model, the Ryoan sandbox is chosen here to ensure, even if the model tries to leak the data, that the data will be confined inside the sandbox, preventing the leakage. Chiron also supports a distributed training process by executing multiple enclaves that exchange model parameters through the server.

Chiron focuses on outsourced learning by using a secure enclave environment. The main difference between Chiron and Ohrimenko16 [95] is the code execution. Chiron allows the execution of untrusted code to update the model and implements protection by using sandboxes such that the code will not leak the data outside the enclave. On the other hand, Ohrimenko16 requires all codes inside the SGX enclave to be

public to ensure that the code is trusted. The main weakness relies on the assumption that the model is not exposed to other parties. As a result, if an adversary can get an access to the trained model, there will be leakage of data, as shown by Shokri *et al.* [96]. This leakage problem can be solved by using differential privacy-based training algorithm [90].

Table 5 shows the features of our surveyed secure enclaves-based PPDL.

TABLE 5. Features of Our Surveyed Secure Enclaves-based PPDL.

References	Key Concept	Learning Type	Dataset
Chiron [51]	Combines SGX enclaves and Ryoan sandbox to provide a black-box system for PPDL	Server-based	CIFAR ImageNet
SLALOM [92]	Utilizes a trusted execution environment to isolate computation processes	Server-based	-

E. HYBRID-BASED PPDL

Ohrimenko16 [95] proposes a secure enclave platform based on the SGX system for secure MPC. It focuses on collaborative learning, providing a prediction service in a cloud. Ohrimenko16 requires all codes inside the SGX enclave to be public to ensure that the code is trusted. The main weakness of this method is claimed to its inherent vulnerability to information leakage due to GAN attack as shown by Hitaj *et al.* [97].

Chase17 [98] wants to propose a private collaborative framework for machine learning. The main idea is to combine secure MPC with DP for the privacy part and leverage NN for the machine learning part. The weakness of this method is found to undergo a decrease in accuracy when implemented in a large network, exhibiting the scalability issue. In addition, its data privacy can only be guaranteed if the participants are non-colluding.

In GAZELLE [88], HE is combined with GC to ensure privacy and security in a MLaaS environment. For the HE library, it utilizes Single Instruction Multiple Data (SIMD) which includes addition and multiplication of ciphertext to improve the encryption speed. The Gazelle algorithm accelerates the convolutional and the matrix multiplication processes. An automatic switch between HE and GC is implemented such that encrypted data can be processed in NN. For the deep learning part, it leverages CNN comprising two convolutional layers, two ReLU layers as activation layers, one pooling layer, and one fully connected layer. The author used MNIST and CIFAR-10 datasets during the experiment and successfully showed that Gazelle outperforms several popular techniques such as MiniONN [84] and Cryptonets [61] in terms of run time. Furthermore, to prevent a linkage attack, Gazelle limits the number of classification queries from a client. The limitation of GAZELLE is claimed to support two-party computation scheme only since it utilizes garbled circuit for the secure exchange of two parties.

Ryffel18 [99] introduces a PPDL framework using federated learning built over PyTorch [100]. Federated learning requires multiple machines to train data in a decentralized environment. It enables clients to learn a shared prediction model using the data in their own device. The author combines secure MPC with DP to build a protocol enables federated learning. Overall, the proposed approach has overhead problem because of the bottleneck in the low-level library, compared to the high level python API. The proposed approach is vulnerable to collusion attack if the participants collude with each other.

CrypTFlow [101] combines secure enclaves with secret sharing in DNN to secure the learning process of the ImageNet dataset. The main weakness of CrypTFlow is believed not to support GPU processing. As a result, the computation overhead during the secure training is still high.

Table 6 shows the features of our surveyed hybrid-based PPDL.

TABLE 6. Features of Our Surveyed Hybrid-based PPDL.

References	Key Concept	Learning Type	Dataset
Ohrimenko16 [95]	Applies secure enclaves platform based for secure MPC	Server-based	MovieLens MNIST
Chase17 [98]	Proposes a private collaboration framework by combining secure MPC with DP	Server-assisted	MNIST
GAZELLE [88]	Combines HE with garbled circuit using SIMD	Server-based	MNIST CIFAR-10
Ryffel18 [99]	Builds a new protocol for federated learning in PPDL	Server-assisted	Boston Housing Pima Diabetes
CrypTFlow [101]	Combines secure enclaves with secret sharing in DNN	Server-assisted	ImageNet

We have summarized the general limitations of each PPDL method and our idea to overcome those limitations in table 7.

F. COMPARISON OF STATE-OF-THE-ART PPDL METHODS

We divided our comparison table into two types: performance comparison I and performance comparison II in Table 8 and Table 9, respectively. To compare the performance of each surveyed paper, we used the privacy metrics and performance metrics defined in section VI. The privacy metrics include Privacy of Client (PoC), Privacy of Model (PoM), and Privacy of Result (PoR). The performance metrics include accuracy, run time, and data transfer.

VIII. CHALLENGE AND WEAKNESS

In this section, we will discuss the challenges and weaknesses of utilizing PPDL for MLaaS from the papers that we surveyed. To analyze the limitations, we divided the PPDL approach into two main categories based on the type of transmission: the model parameter transmission approach and the data transmission approach. The model parameter transmission approach means that the model parameter is transmitted from the client to the server while the local data is kept by the client, and the training is performed on the

TABLE 7. Summary of Weakness and How to Overcome It.

Method	Main Limitation	How to Overcome
HE-based PPDL	More layer leads to more complexity because of the property of HE.	Adding batch normalization layer between pooling layer and activation layer.
	Accuracy highly dependents on the approximation of the activation function.	Using the polynomial with lowest possible degree as the activation function.
SecureMPC-based PPDL	Most of the current publications only guarantee privacy of clients based on honest-but-curious adversary, but they do not have protection against malicious adversary.	Designing a protocol that each participant shares their secret data, then the server runs secure computation to generate the output. By doing this, even if the server is malicious, it cannot see either the input or output value.
Differential Privacy-based PPDL	Differential privacy is computationally very expensive because it requires many interactions between server and client when adding the noise.	Distributed differential privacy can be a good solution to reduce the interactions. Since the architecture of previous differential privacy-based PPDL techniques are centralized, distributed differential privacy will be an interesting option.
Secure Enclaves-based PPDL	There is a data leakage issue due to side channel attack.	Utilizing differential privacy so that the adversary cannot get the real data, even if the side channel attack is successful.

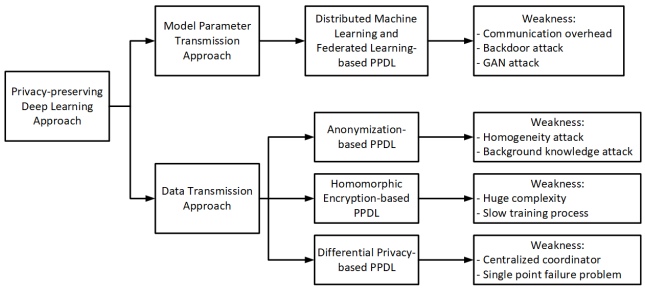

FIGURE 19. The Challenges and Weaknesses of State-of-the-Art PPDL Methods.

client side. On the other hand, the data transmission approach means that the client data itself is transferred to the server for the training process. In short, the challenges and weaknesses of state-of-the-art PPDL methods are shown in Fig. 19.

A. MODEL PARAMETER TRANSMISSION APPROACH

In this approach, during the training process, a model parameter is transmitted instead of the training data. PPDLs based on distributed machine learning and federated learning are included in this scenario. In distributed learning [89], [102]–[104], data owners keep their own data secret without revealing this data to another party. During each training stage, participants send their locally computed parameters to the server. By doing this, the participants can learn collaboratively without revealing their own data [105]. On the other hand, in federated learning [106]–[108], model provider sends the model to participants. Then, each participant executes the training process using their local data, resulting in an updated model. After that, the updated model is sent back to the model provider. The model provider will measure the average value of the gradient descent and update the model. We can see that the model parameter transmission approach reduces the communication overhead but increases the local computation overhead. This learning approach is vulnerable to backdoor attacks [109] and GAN-based attacks [110].

B. DATA TRANSMISSION APPROACH

In this approach, the participants send their data to the training server. Some PPDL methods that belong to this class are anonymization, HE, and DP-based PPDL. The main purpose of the anonymization technique is to remove the correlation between the data owner and the data entries. However, it requires a trusted coordinator to perform the anonymization process and distribute the result to the participants. It is also vulnerable to a single point of failure as a trusted proxy needs to perform the anonymization process and send the result to the participants [111]. HE-based PPDL does not require key distribution management because the computation can be performed on encrypted data. However, it has limitations in the computation format. The computation is limited to a polynomial of bounded degree; thus, it works in a linear nature. Another weakness of HE-based PPDL is the slow training process as it has huge complexity, and the computation process will lead to data swelling [112].

A bootstrapping idea [31], [34], [113] has been introduced to solve this problem by reducing the complexity and the computation time. The majority of the work focuses on polynomial approximation for non-linear operations. The main goal of DP-based PPDL is to perturb the sample data for the training process. It is often used for data such as histograms or tables. The main weakness of DP-based PPDL is its centralized nature. One main trusted coordinator that is responsible for data collection and giving the response to queries from participants. This trusted coordinator is vulnerable to a single point of failure. If this kind of failure occurs and each participant perturbs the training data, the model will yield poor accuracy [112]. Thus, a centralized coordinator is very susceptible to the single point of failure problem. In a nutshell, we can conclude that the data transmission

TABLE 8. Performance Comparison I.

References	Used PP Technique					Privacy Parameters			ML/DL Method	Dataset Type
	HE	Secure-MPC	Oblivious Transfer	Differential Privacy	Secure Enclaves	PoC	PoM	PoR		
CryptoNets [61]	O	X	X	X	X	O	X	O	CNN	Image
PATE [90]	X	X	X	O	X	O	O	X	GAN	Image
GAZELLE [88]	O	O	O	X	X	O	O	O	CNN	Image
TAPAS [72]	O	X	X	X	X	O	O	O	CNN	Image
FHE DiNN [75]	O	X	X	X	X	O	O	O	CNN	Image
E2DM [76]	O	X	X	X	X	O	O	O	CNN	Image
ABY3 [85]	X	O	O	X	X	X	O	O	CNN	Image
Xue18 [77]	O	X	X	X	X	O	X	O	CNN	Image
Aono17 [68]	O	X	X	X	X	O	X	O	NN	Image
Zhang19 [58]	O	X	X	X	X	O	X	O	C-Means algorithm	Image
ML Confidential [62]	O	X	X	X	X	O	X	O	Linear Means Classifier	CSV
Chabanne17 [70]	O	X	X	X	X	O	X	O	DNN	Image
CryptoDL [59]	O	X	X	X	X	O	X	O	CNN	Image
Liu18 [60]	O	X	X	X	X	O	X	O	CNN	Image
SecureML [83]	X	O	O	X	X	X	O	O	NN	Image
DeepSecure [71]	X	O	O	X	X	X	O	O	CNN	Image
MiniONN [84]	X	O	O	X	X	X	O	O	CNN	Image
SLALOM [92]	X	X	X	X	O	O	O	O	DNN	Image
SecureNN [87]	X	O	O	X	X	X	O	O	CNN	Image
Ryffel18 [99]	X	O	O	O	X	O	O	O	Federated Learning	CSV
Faster Crytonets [65]	O	X	X	X	X	O	X	O	CNN	Image
CodedPrivateML [89]	X	O	O	X	X	O	O	O	Logistic Regression	Image
Chiron [51]	X	X	X	X	O	O	O	O	DNN	Image
Bu19 [91]	X	X	X	O	X	O	O	O	DNN	Image
CrypTFlow [101]	X	O	O	X	O	O	O	O	DNN	Image
Chameleon [86]	X	O	O	X	X	X	O	O	CNN	Image
Chase17 [98]	X	O	O	O	X	X	O	O	CNN	Image
Ohrimenko16 [95]	X	O	O	X	O	O	X	O	DNN	CSV

approach reduces the computation overhead but increase the communication overhead.

C. ANALYSIS AND SUMMARY

After discussing the challenges and weaknesses in PPDL from the two categories above, we summarize the two main problems in PPDL: the computation overhead and communication overhead.

1) COMPUTATION OVERHEAD

One of the most important issues in MLaaS is the computation overhead. In MLaaS, the overhead issues occur during

TABLE 9. Performance Comparison II.

PPDL Type	Proposed Scheme	Accuracy (%)	Inference Time (s)
HE-based PPDL	Cryptonets [61]	Good (98.95)	Bad (297.5)
	Aono17 [68]	Good (97.00)	Good (120)
	Chabanne17 [70]	Good (99.30)	-
	CryptoDL [59]	Good (99.52)	Bad (320)
	TAPAS [72]	Good (98.60)	Good (147)
	FHE-DiNN [75]	Bad (96.35)	Good (1.64)
	E2DM [76]	Good (98.10)	Good (28.59)
	Xue18 [77]	Good (99.73)	-
	Liu18 [60]	Good (98.97)	Bad (477.6)
	Faster Cryptonets [65]	Bad (80.61)	-
	CryptoNN [78]	Bad (95.48)	-
	Zhang19 [58]	-	-
	ML Confidential [62]	Bad (95.00)	Bad (255.7)
Secure MPC-based PPDL	SecureML [83]	Bad (93.40)	Good (4.88)
	MiniONN [84]	Good (98.95)	Good (9.32)
	ABY3 [85]	Bad (94.00)	Good (0.01)
	DeepSecure [71]	Good (98.95)	Good (9.67)
	Chameleon [86]	Good (99.00)	Good (2.24)
	SecureNN [87]	Good (99.15)	Good (0.076)
	CodedPrivateML [89]	Bad (95.04)	Bad (110.9)
Differential Privacy-based PPDL	PATE [90]	Good (98.10)	-
	Bu19 [91]	Good (98.00)	-
Secure Enclaves-based PPDL	Chiron [51]	Bad (89.56)	-
	SLALOM [92]	Bad (92.4)	-
Hybrid-based PPDL	Ohrimenko16 [95]	Good (98.7)	Good (2.99)
	Chase17 [98]	Good (98.9)	-
	Gazelle [88]	-	Good (0.03)
	Ryffel18 [99]	Bad (70.3)	-
	CrypTFlow [101]	Bad (93.23)	Good (30)

the HE process, deep learning training (including inferencing), and data perturbation. Currently, utilizing deep learning for large-scale service is not feasible in real life because of this scalability problem.

2) COMMUNICATION OVERHEAD

In MLaaS, communication overhead occurs during the interaction among clients, model providers, and server providers. In particular, we can categorize communication overhead into the HE process, additive or multiplicative perturbation, and iterative communication. In the distributed machine learning scenario, including the federated learning, this factor is the main scalability problem that becomes the main issue. The iterative communications to exchange data and model parameters between each party will produce a significant overhead problem.

IX. ATTACKS ON DL MODEL AND PPDL AS THE POSSIBLE SOLUTIONS

A. SECURITY GOALS OF PPDL

PPDL solutions on DL-as-a-service frameworks have three major security goals. The first goal is to prevent the server from acquiring the training data in the training phase which would be sensitive data owned by the client. All PPDL schemes contain privacy measures to prevent the direct leak of the training data. HE- and MPC-based approaches solve this by encrypting and distributing the training data, respectively. Some methods perform lower-layer calculations in the client side while hardware-based approaches encapsulate lower-layer calculations inside some confidential environment.

The second security goal of PPDL aims to prevent the server from directly acquiring the input to the model in the prediction phase. In most cases, this goal is achieved together with the first goal. This goal is only applied when the client delegates prediction to the server.

The third goal is to prevent the server from taking advantage of white-box access of the model. With the white-box access on a model, a server (as an adversary) may deploy several known attacks which are known to be easy on the white-box assumption. As DNNs tend to have more parameters than other machine learning algorithms due to the hidden layers, black-box models could retain more information on training data.

Many cryptography-based approaches achieve the third goal by keeping the parameters encrypted. However, some PPDL models do not assume this third goal and allow the server to access the plaintext parameters of the model. For instance, DP-based models allow white-box access, but the schemes aim to make the information extractable from the parameters negligible.

Although there are many other types of attacks on DL models, in this section we only discuss the attacks that can be mitigated by some of the PPDL approaches. In other words, the following attacks are related to one of the goals of PPDL. Table 10 provides a brief summary on PPDL as a countermeasure against attacks.

B. MEMBERSHIP INFERENCE ATTACK

Generally, membership inference means deciding whether given data were used for generating some aggregation of the data (or not). In the context of deep learning, a model itself (including the model parameters) can be regarded as

TABLE 10. PPDL as Countermeasures Against Attacks on DL Models.

PPDL Types	Cryptography -based	DP -based	Hardware -based
Membership Inference Attack	O	Δ	X
Model Inversion Attack	O	Δ^*	X
Model Extraction Attack	O	N/A	N/A

O : An effective countermeasure for the given attack.
X : An ineffective countermeasure for the given attack.
Δ : The effectiveness of the defense against the given attack has a trade-off with the privacy-preserving parameters of DL models.
Δ^* : This trade-off has been confirmed for non-DL models, and it is expected to be the same for DL models.
N/A: The adversarial assumption of the given attack is not applicable for the PPDL method.

the 'aggregation' of the training data. Therefore, membership inference attacks on DL models indicate attacks to decide whether given data belong to the training dataset (or not). Shokri *et al.* [96] provided one of the first suggestions of membership inference attacks.

Membership inference attacks are the attacks for the models violating the first security goal of PPDL. Stronger versions of membership inference attacks include extraction of some properties of sensitive training data or even recovery of the training data, which can be reduced to normal membership inference attacks. Usually, membership inference attacks harness overfitting during training, producing a difference in accuracy between the training data and the other data. Some defensive mechanisms dedicated to membership inference have been proposed including dropout [114] and adversarial regularization [115].

In cryptography-based PPDL models, the security against the membership inference attack can be reduced to the security of the underlying cryptosystems. In such models, the adversarial server cannot obtain model parameters in plaintext. Only if the model is public can the adversary have black-box access of the model, just like any outsider attacker. For HW-based models, the adversarial server owns white-box models, allowing the use of white-box membership inference attacks.

For DP-based models, the trade-off between the model accuracy and the performance of membership inference attacks according to the selection of the privacy parameter has been studied [116]. Appropriate choices of the privacy parameter result in moderate utility with low membership inference accuracy. However, further experiments are required for the extensibility of their analysis toward other types of tasks outside image classification.

C. MODEL INVERSION ATTACK

As an attack toward the models does not satisfy the second security goal of PPDL, a model inversion attack is a prediction-phase attack introduced by Fredrikson *et al.* [66], [117]. Given the non-sensitive features of the original input data and their prediction results for a model, model inversion attacks aim to find the sensitive features of the input data.

In cryptography-based and HW-based PPDL models, we expect a similar advantage as that of membership inference attacks. For DP-based models, there has been limited research on the trade-off between the model accuracy and the attack performance, such as the analysis by Wang *et al.* [118] against regression models. Although a similar analysis for differentially private DL models remains for future work, we expect a similar trade-off.

D. MODEL EXTRACTION ATTACK

Model extraction attacks [67], also known as model-stealing attacks, are attacks toward the third security goal of PPDL. When a black-box (target) model is given, the objective of model extraction attacks is to construct a model equivalent to the target model. Once an adversary succeeds with a model extraction attack, the adversary then accesses a white-box model. The attacker can take direct advantage of the model if the model owner sells the model access. The obtained model also becomes a "stepping stone" [119] toward further attacks utilizing white-box models.

Again, adversarial servers against cryptography-based DL models have negligible advantages over those of outsiders, excepting that of the model structure. Without the help of the client, the server cannot obtain the decrypted parameter values of the model.

Most differentially private models and hardware-based PPDL models do not fulfill the third security goal, as they reveal model parameters to the server. The extraction of such PPDL models is meaningless for the adversarial servers, as the servers already have the white-box access on the models, which is the purpose of the attack. Although the servers possess some part of the model parameters, it is relatively easy to extract the remaining parameters by observing the intermediate activation levels.

X. CONCLUSION

In this paper, we have provided a complete review of state-of-the-art PPDL on MLaaS. Our discussion covers the classical PP method and the utilization of DL for PP. Our work also addresses the limitation of implementing novel DL techniques with PP, including the analysis of the original structure of NN and the modifications needed to use it in privacy-preserving environment. Furthermore, we have proposed a multi-scheme PPDL classification based on adversarial model, PP methods, and the challenges and weaknesses in state-of-the-art PPDL methods.

To summarize the main trend, an annual roadmap that highlights the development of PPDL complemented with detailed comparisons of each PPDL work has been presented. Security goals and attack models on PPDL also have been discussed, with the possible countermeasures for each scenario. In brief, the trade-off between accuracy and complexity during the

substitution process of the non-linear activation function is identified as the main challenge in PPDL.

An open problem for future research is reducing computational burden. How to divide the burden between a client and a server optimally to achieve the best performance is a big challenge that needs to be addressed in the future. Another challenge is to ensure the PoC, PoM, and PoR simultaneously with two extra computations from the client's and model's perspectives while maintaining the computational performance. Last but not least, implementing PPDL based on federated learning will be an interesting topic. We believe that the future direction of PPDL is going to focus on combining federated learning and state-of-the-art PPDL to overcome the current privacy issues during data collection phase in MLaaS.

ACKNOWLEDGMENT

This article was presented in part at the 2019 Machine Learning for Cyber Security: Second International Conference, ML4CS 2019, Xi'an, China, September 19-21, 2019.

REFERENCES

[1] H. C. Tanuwidjaja, R. Choi, and K. Kim, "A survey on deep learning techniques for privacy-preserving," in *Proc. Int. Conf. Mach. Learn. Cyber Secur.* Cham, Switzerland: Springer, 2019, pp. 29–46.

[2] V. K. Singh and A. K. Gupta, "From artificial to collective intelligence: Perspectives and implications," in *Proc. 5th Int. Symp. Appl. Comput. Intell. Informat.*, May 2009, pp. 545–550.

[3] E. Hesamifard, H. Takabi, M. Ghasemi, and C. Jones, "Privacy-preserving machine learning in cloud," in *Proc. Cloud Comput. Secur. Workshop (CCSW)*, 2017, pp. 39–43.

[4] E. Hesamifard, H. Takabi, M. Ghasemi, and N. W. Rebecca, "Privacy-preserving machine learning as a service," *Proc. Privacy Enhancing Technol.*, vol. 2018, no. 3, pp. 123–142, Jun. 2018.

[5] R. Mendes and J. P. Vilela, "Privacy-preserving data mining: Methods, metrics, and applications," *IEEE Access*, vol. 5, pp. 10562–10582, 2017.

[6] M. Siddula, L. Li, and Y. Li, "An empirical study on the privacy preservation of online social networks," *IEEE Access*, vol. 6, pp. 19912–19922, 2018.

[7] J. Zhang, B. Chen, Y. Zhao, X. Cheng, and F. Hu, "Data security and privacy-preserving in edge computing paradigm: Survey and open issues," *IEEE Access*, vol. 6, pp. 18209–18237, 2018.

[8] J. Domingo-Ferrer, O. Farràs, J. Ribes-González, and D. Sánchez, "Privacy-preserving cloud computing on sensitive data: A survey of methods, products and challenges," *Comput. Commun.*, vols. 140–141, pp. 38–60, May 2019.

[9] Z. Rui and Z. Yan, "A survey on biometric authentication: Toward secure and privacy-preserving identification," *IEEE Access*, vol. 7, pp. 5994–6009, 2019.

[10] A. Anand and A. Muthusamy, "Data security and privacy-preserving in cloud computing paradigm: Survey and open issues," in *Cloud Computing Applications and Techniques for E-Commerce*. Hershey, PA, USA: IGI Global, 2020, pp. 99–133.

[11] H.-Y. Tran and J. Hu, "Privacy-preserving big data analytics a comprehensive survey," *J. Parallel Distrib. Comput.*, vol. 134, pp. 207–218, Dec. 2019.

[12] M. Zheng, D. Xu, L. Jiang, C. Gu, R. Tan, and P. Cheng, "Challenges of privacy-preserving machine learning in IoT," in *Proc. 1st Int. Workshop Challenges Artif. Intell. Mach. Learn. Internet Things-AIChallengeIoT*, 2019, pp. 1–7.

[13] M. S. Riazi, B. Darvish Rouani, and F. Koushanfar, "Deep learning on private data," *IEEE Secur. Privacy*, vol. 17, no. 6, pp. 54–63, Nov. 2019.

[14] S. Sultan, "Privacy-preserving metering in smart grid for billing, operational metering, and incentive-based schemes: A survey," *Comput. Secur.*, vol. 84, pp. 148–165, Jul. 2019.

[15] R. Alvarez and M. Nojoumian, "Comprehensive survey on privacy-preserving protocols for sealed-bid auctions," *Comput. Secur.*, vol. 88, Jan. 2020, Art. no. 101502.

[16] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: K-anonymity and its enforcement through generalization and suppression," Sci. Res. Inst. (SRI) Int., Comput. Sci. Lab., Menlo Park, CA, USA, Tech. Rep. SRI-CSL-98-04, 1998.

[17] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "*L*-diversity: Privacy beyond *k*-anonymity," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, p. 3, 2007.

[18] N. Li, T. Li, and S. Venkatasubramanian, "T-closeness: Privacy beyond K-Anonymity and L-Diversity," in *Proc. IEEE 23rd Int. Conf. Data Eng.*, Apr. 2007, pp. 106–115.

[19] X. Xiao and Y. Tao, "M-invariance: Towards privacy preserving re-publication of dynamic datasets," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2007, pp. 689–700.

[20] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," in *Foundations of Secure Computation*, vol. 4, no. 11. Cambridge, MA, USA: Academia Press, pp. 169–180, 1978.

[21] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[22] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Symp. Symp. Theory Comput. (STOC)*, 2009, pp. 169–178.

[23] J. H. Cheon, J.-S. Coron, J. Kim, M. S. Lee, T. Lepoint, M. Tibouchi, and A. Yun, "Batch fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2013, pp. 315–335.

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

[25] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," *SIAM J. Comput.*, vol. 43, no. 2, pp. 831–871, 2014.

[26] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Advances in Cryptology (CRYPTO)*. Berlin, Germany: Springer, 2011, pp. 505–524.

[27] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Advances in Cryptology (CRYPTO)*. Berlin, Germany: Springer, 2013, pp. 75–92.

[28] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, p. 13, 2014.

[29] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," IACR Cryptol. ePrint Arch., Lyon, France, Tech. Rep. 2012/144, 2012, p. 144.

[30] M. Clear and C. M. Goldrick, "Attribute-based fully homomorphic encryption with a bounded number of inputs," *Int. J. Appl. Cryptogr.*, vol. 3, no. 4, pp. 363–376, 2017.

[31] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachene, "Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Springer, 2016, pp. 3–33.

[32] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "TFHE: Fast fully homomorphic encryption over the torus," *J. Cryptol.*, vol. 33, no. 1, pp. 34–91, Jan. 2020.

[33] S. Halevi and V. Shoup, "Algorithms in HElib," in *Proc. Annu. Cryptol. Conf.* Berlin, Germany: Springer, 2014, pp. 554–571.

[34] L. Ducas and D. Micciancio, "FHEW: Bootstrapping homomorphic encryption in less than a second," in *Proc. Annual Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2015, pp. 617–640.

[35] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Cham, Switzerland: Springer, 2017, pp. 409–437.

[36] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2005, pp. 457–473.

[37] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2011, pp. 253–273.

[38] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2007, pp. 535–554.

[39] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. Workshop Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1984, pp. 47–53.

[40] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2001, pp. 213–229.

[41] B. Waters, "Efficient identity-based encryption without random oracles," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2005, pp. 114–127.

[42] C. Gentry, "Practical identity-based encryption without random oracles," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2006, pp. 445–464.

[43] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. 14th Annu. ACM Symp. Theory Comput. (STOC)*, 2008, pp. 197–206.

[44] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption scheme," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2003, pp. 255–271.

[45] S. Agrawal, D. Boneh, and X. Boyen, "Efficient lattice (H)IBE in the standard model," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2010, pp. 553–572.

[46] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur. (CCS)*, 2006, pp. 89–98.

[47] A. C.-C. Yao, "How to generate and exchange secrets," in *Proc. 27th Annu. Symp. Found. Comput. Sci. (sfcs)*, Oct. 1986, pp. 162–167.

[48] O. Goldreich, S. Micali, and A. Wigderson, "How to play ANY mental game," in *Proc. 19th Annu. ACM Conf. Theory Comput. (STOC)*, 1987, pp. 218–229.

[49] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2006, pp. 265–284.

[50] L. Qi, X. Zhang, S. Li, S. Wan, Y. Wen, and W. Gong, "Spatial-temporal data-driven service recommendation with privacy-preservation," *Inf. Sci.*, vol. 515, pp. 91–102, Apr. 2020.

[51] T. Hunt, C. Song, R. Shokri, V. Shmatikov, and E. Witchel, "Chiron: Privacy-preserving machine learning as a service," 2018, *arXiv:1803.05961*. [Online]. Available: http://arxiv.org/abs/1803.05961

[52] R. Intel, *Software Guard Extensions Programming Reference*, Intel Corporation, Santa Clara, CA, USA, 2014.

[53] J. Doweck, W.-F. Kao, A. K.-Y. Lu, J. Mandelblat, A. Rahatekar, L. Rappoport, E. Rotem, A. Yasin, and A. Yoaz, "Inside 6th-generation intel core: New microarchitecture code-named skylake," *IEEE Micro*, vol. 37, no. 2, pp. 52–62, Mar. 2017.

[54] T. Hunt, Z. Zhu, Y. Xu, S. Peter, and E. Witchel, "Ryoan: A distributed sandbox for untrusted computation on secret data," *ACM Trans. Comput. Syst.*, vol. 35, no. 4, pp. 1–32, 2018.

[55] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, Contour and Grouping in Computer Vision*. Berlin, Germany: Springer, 1999, pp. 319–345.

[56] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[57] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167

[58] Q. Zhang, L. T. Yang, A. Castiglione, Z. Chen, and P. Li, "Secure weighted possibilistic c-means algorithm on cloud for clustering big data," *Inf. Sci.*, vol. 479, pp. 515–525, Apr. 2019.

[59] E. Hesamifard, H. Takabi, and M. Ghasemi, "CryptoDL: Deep neural networks over encrypted data," 2017, *arXiv:1711.05189*. [Online]. Available: http://arxiv.org/abs/1711.05189

[60] W. Liu, F. Pan, X. A. Wang, Y. Cao, and D. Tang, "Privacy-preserving all convolutional net based on homomorphic encryption," in *Proc. Int. Conf. Netw.-Based Inf. Syst.* Cham, Switzerland: Springer, 2018, pp. 752–762.

[61] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 201–210.

[62] T. Graepel, K. Lauter, and M. Naehrig, "ML confidential: Machine learning on encrypted data," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Berlin, Germany: Springer, 2012, pp. 1–21.

[63] M. Skurichina and R. P. W. Duin, "Bagging, boosting and the random subspace method for linear classifiers," *Pattern Anal. Appl.*, vol. 5, no. 2, pp. 121–135, Jun. 2002.

[64] S.-J. Kim, A. Magnani, and S. Boyd, "Robust Fisher discriminant analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 659–666.

[65] E. Chou, J. Beal, D. Levy, S. Yeung, A. Haque, and L. Fei-Fei, "Faster CryptoNets: Leveraging sparsity for real-world encrypted inference," 2018, *arXiv:1811.09953*. [Online]. Available: http://arxiv.org/abs/1811.09953

[66] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2015, pp. 1322–1333.

[67] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction apis," in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)*, 2016, pp. 601–618.

[68] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1333–1345, May 2018.

[69] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2015, pp. 1310–1321.

[70] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network," IACR Cryptol. ePrint Arch., Lyon, France, Tech. Rep. 2017/035, 2017, p. 35.

[71] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "DeepSecure: Scalable provably-secure deep learning," in *Proc. 55th ACM/ESDA/IEEE Design Autom. Conf. (DAC)*, Jun. 2018, pp. 1–6.

[72] A. Sanyal, M. J. Kusner, A. Gascón, and V. Kanade, "TAPAS: Tricks to accelerate (encrypted) prediction as a service," 2018, *arXiv:1806.03461*. [Online]. Available: http://arxiv.org/abs/1806.03461

[73] M. Kim and P. Smaragdis, "Bitwise neural networks," 2016, *arXiv:1601.06071*. [Online]. Available: http://arxiv.org/abs/1601.06071

[74] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6869–6898, 2017.

[75] F. Bourse, M. Minelli, M. Minihold, and P. Paillier, "Fast homomorphic evaluation of deep discretized neural networks," in *Proc. Annu. Int. Cryptol. Conf.* Cham, Switzerland: Springer, 2018, pp. 483–512.

[76] X. Jiang, M. Kim, K. Lauter, and Y. Song, "Secure outsourced matrix computation and application to neural networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Jan. 2018, pp. 1209–1222.

[77] H. Xue, Z. Huang, H. Lian, W. Qiu, J. Guo, S. Wang, and Z. Gong, "Distributed large scale privacy-preserving deep mining," in *Proc. IEEE 3rd Int. Conf. Data Sci. Cyberspace (DSC)*, Jun. 2018, pp. 418–422.

[78] R. Xu, J. B. D. Joshi, and C. Li, "CryptoNN: Training neural networks over encrypted data," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 1199–1209.

[79] R. Krishnapuram and J. M. Keller, "The possibilistic C-means algorithm: Insights and recommendations," *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 3, pp. 385–393, Aug. 1996.

[80] D. Gustafson and W. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," in *Proc. IEEE Conf. Decis. Control Including 17th Symp. Adapt. Processes*, Jan. 1978, pp. 761–766.

[81] P. Smyth, "Model selection for probabilistic clustering using cross-validated likelihood," *Statist. Comput.*, vol. 10, no. 1, pp. 63–72, Jan. 2000.

[82] A. Boulemtafes, A. Derhab, and Y. Challal, "A review of privacy-preserving techniques for deep learning," *Neurocomputing*, vol. 384, pp. 21–45, Apr. 2020.

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

[84] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via MiniONN transformations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 619–631.

[85] P. Mohassel and P. Rindal, "ABY^3: A mixed protocol framework for machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 35–52.

[86] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A hybrid secure computation framework for machine learning applications," in *Proc. Asia Conf. Comput. Commun. Secur.*, 2018, pp. 707–721.
[87] S. Wagh, D. Gupta, and N. Chandran, "Securenn: 3-Party secure computation for neural network training," *Proc. Privacy Enhancing Technol.*, vol. 2019, no. 3, pp. 26–49, 2019.
[88] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in *Proc. 27th USENIX Secur. Symp. (USENIX Secur.)*, 2018, pp. 1651–1669.
[89] J. So, B. Guler, A. Salman Avestimehr, and P. Mohassel, "CodedPrivateML: A fast and privacy-preserving framework for distributed machine learning," 2019, *arXiv:1902.00641*. [Online]. Available: http://arxiv.org/abs/1902.00641
[90] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," 2016, *arXiv:1610.05755*. [Online]. Available: http://arxiv.org/abs/1610.05755
[91] Z. Bu, J. Dong, Q. Long, and W. J. Su, "Deep learning with Gaussian differential privacy," 2019, *arXiv:1911.11607*. [Online]. Available: http://arxiv.org/abs/1911.11607
[92] F. Tramèr and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," 2018, *arXiv:1806.03287*. [Online]. Available: http://arxiv.org/abs/1806.03287
[93] J. Van Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx, "Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution," in *Proc. 27th USENIX Secur. Symp. (USENIX Secur.)*, 2018, pp. 991–1008.
[94] F. McKeen, I. Alexandrovich, I. Anati, D. Caspi, S. Johnson, R. Leslie-Hurd, and C. Rozas, "Intel software guard extensions (Intel SGX) support for dynamic memory management inside an enclave," in *Proc. Hardw. Architectural Support Secur. Privacy*, 2016, pp. 1–9.
[95] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, "Oblivious multi-party machine learning on trusted processors," in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)*, 2016, pp. 619–636.
[96] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 3–18.
[97] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 603–618.
[98] M. Chase, R. Gilad-Bachrach, K. Laine, K. E. Lauter, and P. Rindal, "Private collaborative neural network learning," IACR Cryptol. ePrint Arch., Lyon, France, Tech. Rep. 2017/762, 2017, p. 762.
[99] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach, "A generic framework for privacy preserving deep learning," 2018, *arXiv:1811.04017*. [Online]. Available: http://arxiv.org/abs/1811.04017
[100] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.
[101] N. Kumar, M. Rathee, N. Chandran, D. Gupta, A. Rastogi, and R. Sharma, "CrypTFlow: Secure TensorFlow inference," 2019, *arXiv:1909.07814*. [Online]. Available: http://arxiv.org/abs/1909.07814
[102] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, and K. Yang, "Large scale distributed deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1223–1231.
[103] J. Hamm, A. C. Champion, G. Chen, M. Belkin, and D. Xuan, "Crowd-ML: A privacy-preserving learning framework for a crowd of smart devices," in *Proc. IEEE 35th Int. Conf. Distrib. Comput. Syst.*, Jun. 2015, pp. 11–20.
[104] H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*. [Online]. Available: http://arxiv.org/abs/1602.05629
[105] G. Song and W. Chai, "Collaborative learning for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1832–1841.
[106] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur. (AISec)*, 2019, pp. 1–11.
[107] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," 2017, *arXiv:1711.10677*. [Online]. Available: http://arxiv.org/abs/1711.10677
[108] V. Mugunthan, A. Peraire-Bueno, and L. Kagal, "PrivacyFL: A simulator for privacy-preserving and secure federated learning," 2020, *arXiv:2002.08423*. [Online]. Available: http://arxiv.org/abs/2002.08423
[109] Z. Sun, P. Kairouz, A. Theertha Suresh, and H. Brendan McMahan, "Can you really backdoor federated learning?" 2019, *arXiv:1911.07963*. [Online]. Available: http://arxiv.org/abs/1911.07963
[110] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 2512–2520.
[111] I. J. Vergara-Laurens, L. G. Jaimes, and M. A. Labrador, "Privacy-preserving mechanisms for crowdsensing: Survey and research challenges," *IEEE Internet Things J.*, vol. 4, no. 4, pp. 855–869, Aug. 2017.
[112] L. Jiang, R. Tan, X. Lou, and G. Lin, "On lightweight privacy-preserving collaborative learning for Internet-of-Things objects," in *Proc. Int. Conf. Internet Things Design Implement.*, Apr. 2019, pp. 70–81.
[113] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "Bootstrapping for approximate homomorphic encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 2018, pp. 360–384.
[114] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "ML-leaks: Model and data independent membership inference attacks and defenses on machine learning models," 2018, *arXiv:1806.01246*. [Online]. Available: http://arxiv.org/abs/1806.01246
[115] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Jan. 2018, pp. 634–646.
[116] M. A. Rahman, T. Rahman, R. Laganière, N. Mohammed, and Y. Wang, "Membership inference attack against differentially private deep learning model," *Trans. Data Privacy*, vol. 11, no. 1, pp. 61–79, 2018.
[117] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, "Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing," in *Proc. 23rd USENIX Secur. Symp. (USENIX Secur.)*, 2014, pp. 17–32.
[118] Y. Wang, C. Si, and X. Wu, "Regression model fitting under differential privacy and model inversion attack," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015, pp. 1003–1009.
[119] 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.

HARRY CHANDRA TANUWIDJAJA received the B.S. and M.S. degrees in electrical engineering from the Bandung Institute of Technology (ITB), Indonesia, in 2013 and 2015, respectively. He is currently pursuing the Ph.D. degree with the School of Computing, Korea Advanced Institute of Science and Technology (KAIST), South Korea. His current research interests include malware detection, machine-learning, privacy-preserving, and intrusion detection systems.

RAKYONG CHOI received the B.S. and M.S. degrees from the Department of Mathematical Sciences, Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2011 and 2013, respectively, where he is currently pursuing the Ph.D. degree with the School of Computing. His current research interests include lattice-based cryptography, homomorphic cryptographic schemes, and their applications.

SEUNGGEUN BAEK received the B.S. degree from the School of Computing, Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2018, where he is currently pursuing the master's degree with the Graduate School of Information Security.

KWANGJO KIM (Member, IEEE) received the B.Sc. and M.Sc. degrees in electronic engineering from Yonsei University, Seoul, South Korea, in 1980 and 1983, respectively, and the Ph.D. degree from the Division of Electrical and Computer Engineering, Yokohama National University, Yokohama, Japan, in 1991. From 1979 to 1997, he worked at the Electronics and Telecommunications Research Institute (ETRI) as the Head in Coding Section #1. He was a Visiting Professor with the Massachusetts Institute of Technology, Cambridge, MA, USA; University of California at San Diego, La Jolla, CA, USA, in 2005; and the Khalifa University of Science, Technology and Research, Abu Dhabi, UAE, in 2012; and an Education Specialist with the Bandung Institute of Technology, Bandung, Indonesia, in 2013. He is currently a Full Professor with the School of Computing and Graduate School of Information Security, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, the Korean representative to IFIP TC-11 and the honorable President of the Korea Institute of Information Security and Cryptography (KIISC). His current research interests include the theory of cryptology, information security, and their applications. He served as a Board Member of the International Association for Cryptologic Research (IACR) from 2000 to 2004, the Chairperson of the Asiacrypt Steering Committee from 2005 to 2008, and the President of KIISC in 2009. He is a Fellow of the IACR, a member of IEICE and ACM, and a member of the IACR Fellow Selection Committee. Moreover, he serves as the General Chair of Asiacrypt2020 and PQCrypto2021 which will be held in Daejeon, South Korea, in 2020 and 2021, respectively. He serves as an Editor-in-Chief of the online journal *Cryptography* and an Editor of the *Journal of Mathematical Cryptology*.