Review of Privacy-Preserving Methods using Encrypted Data for Neural Network Training and Classification

by

Mario J Lorenzo

mario@mjlorenzo.com

April 2020

Table of Contents

Introduction

The use of real-world data to train machine learning models is complicated by the presence of sensitive data, such as personal identifiers, health, and financial information. This sensitive data is typically regulated, and unauthorized disclosure or leakage of this sensitive data may result in severe penalties or criminal prosecution (Liu et al, 2019). As such, data owners require that the sensitive data be fully de-identified (i.e. redacted) prior to the commencement of training classification activities by a 3rd party vendor. This de-identification process is of course very costly due to the amount of time required to properly redact all sensitive information. The process is also prone to human error where unintentional disclosure of sensitive information is leaked. This is the central issue addressed by the study of Privacy-Preserving methods.

The goal of this survey is to thoroughly review and assess the current research and state-of-the-art for a specific subfield of privacy-preserving methods that focus on the training and classification phases of machine learning algorithms. These methods provide confidentiality of sensitive data using encryption that prevents other parties from viewing the data except for the data owner who retains the encryption key. Another closely related field of study that is often addressed as part of the privacy-preservation research involves protecting (i.e. concealing) the details of the machine learning model and algorithms. The owners of these models are described here as "model owners". Like data owners, model owners may not wish to disclose the details of their intellectual property (i.e. their models) with other parties such as data owners.

Given the recent emergence of Neural Networks as the current mainstream of machine learning research, most of the recent research in this field is directed towards achieving high-performance variations of neural network models on well-known tasks and benchmarks. Therefore, considerable focus is spent in this survey thoroughly reviewing methods and performance of privacy-preserving neural network and underlying mathematical underpinnings.

This survey will also focus on the use of Homomorphic Cryptographic methods (Gentry, 2009), which enable the use of special secure arithmetic functions that can operate on the encrypted data and therefore serving as a theoretical framework and technical underpinning for privacy-preserving methods. Using these privacy-preserving methods, a data owner can confidently share sensitive information with a 3rd party vendor for training or using statistical or machine learning models.

The rest of the survey will be organized as follows: significance of the topic, background on Neural Networks, review of homomorphic encryption, a review of methods utilizing homomorphic encryption, an evaluation comparing the performance of best performing methods, a discussion on remaining challenges and barriers, and concludes with a perspective on future work and direction of study.

Significance

The study of privacy-preserving methods for training machine learning algorithm has become an active area of research largely driven by the revival of Artificial Intelligence during the last decade. The healthcare industry, among other industries, has been an area of interest for applying machine learning to solve a wide range of problems including clinical decision support system (Liu et al, 2019). One major challenge that inhibits the sharing of medical data for advancing the state-of-the-art in machine learning models involves the presence of sensitive

information. This information may include Personal Health Information (PHI), defined by HIPAA act, that imposes various restrictions and sanctions for misuse or unauthorized disclosures of that information. Fear of civil or criminal penalties have impeded the willingness for data owners and algorithm owners to collaborate (Wang et al, 2018).

These concerns about leakage of sensitive information has led to the emergence of privacy-preserving methods to mitigate these risks. Three well-known approaches exist including Differential Privacy, Secure Multiparty Computation (SMC), and cryptographic methods. The use of differential privacy methods rely on randomization and perturbation of the data. This approach has the disadvantage of negatively impacting the performance of the model because it alters the information in the training dataset (Shen et al, 2019). As such, differential-privacy methods fall outside the scope of this survey.

Another common approach requires the use of multi-party computation (or SMC). These methods involve one or more trusted parties that can be used to outsource specific computations by the algorithm owner. The disadvantage and weakness of such systems is based on their reliance on a trusted party. This survey will only include SMC methods when they are used within the context of a cryptographic approach to privacy-preservation.

The cryptographic method, and the focus of this study, involves the use of a cryptographic system used to ensure that the data is protected throughout the training or classification phases of a process. This has been an active direction of research for privacy-preservation worthy of further exploration.

Organization of Privacy-Preserving Methods

Privacy-preserving methods may vary by approach. The following are common patterns observed in the methods surveyed:

1) Privacy-Preserving methods utilizing Standalone Homomorphic Encryption schemes for classification with Neural Networks without multi-party computation
2) Hybrid Privacy-Preserving methods utilizing Homomorphic Encryption schemes for classification with Neural Networks and reliance on multi-party computation
3) Privacy-Preserving methods utilizing Homomorphic Encryption schemes for training Neural Networks with or without multi-party computation

This survey will focus on Standalone and Hybrid Homomorphic Encryption schemes for classification tasks with Neural Networks. Less focus is given to training Neural Networks given the few known implementations of training Neural Networks over encrypted data, but some important methods are nonetheless discussed.

Background

Neural Networks

Neural Networks are a branch of Machine Learning that utilize the concept of a neuron where each neuron has an incoming and outgoing connection forming a neural network. The first layer of neurons in a neural network is known as the input layer. The input to the first layer is in the form of a vector that represents the feature space. The network can have one or more layers, known as hidden layers, that can be connected using different network architectures. The most basic neural network is known as a Perceptron where a single layer is used in the network. These neural networks can grow with multiple layers known as Multi-Layer Perceptron (MLP) or Feed-forward networks. As the number of layers increase, the total number of neurons (i.e. parameters) of the model increases. This increases the depth of computation required to evaluate a given input vector. Recent advances in Neural Networks involve the use of Convolutional Neural Networks (CNN) that represent a reduction of dimensionality between on layer and the next. CNN are considered the state-of-the-art for tackling several types of classification tasks such as image classification among others. Each of these neurons in the network apply a non-linear activation function, such as Rectified Linear Unit (ReLU), Hyperbolic Tangent (tanh), or Sigmoid functions, to an incoming value and update it's weight and propagates the value forward to the next layer.

Using a method called Back-Propagation, a neural network can be trained by updating the weights of each neuron in accordance to a loss function, such as Stochastic Gradient Descent (SGD) (Yuan & Yu, 2012). This survey will not focus on Neural Network algorithms or architecture but rather the use of Homomorphic Encryption schemes with Neural Networks to achieve privacy-preserving Machine Learning.

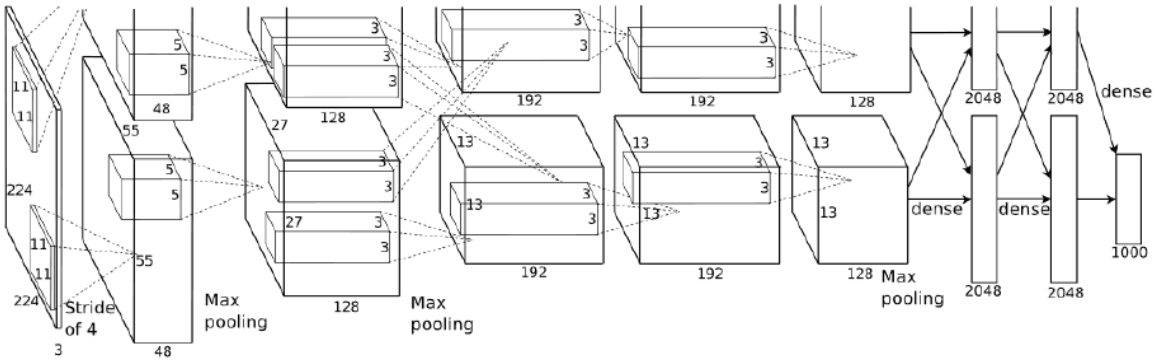

Figure 1 - Example of a CNN (Hesamifard et al, 2017)

Homomorphic Encryption

The integral underpinning behind all the cryptographic privacy-preserving methods for training or classification with Machine Learning rely on data that has been encrypted using homomorphic encryption scheme. Homomorphic encryption is a method of encryption that

preserves special homomorphic properties that allow for encrypted cipher data to be computed without requiring decryption of the data.

Homomorphic functions are defined within a numerical universe and a set of supported operators where the operation of any supported operator on operands (i.e. values) belonging to the defined universe are equivalent within the function and outside of the function (Birkhoff; 1940). The following example shows a homomorphic function that supports the multiplication (i.e. product) operator in the universe of positive integer values:

$$(\mathbb{Z},\cdot): f(x) = x^2$$

This implies that the above function preserves the following homomorphic property:

$$f(x \cdot y) = f(x) \cdot f(y)$$

This means that the multiplication can be done within the function or applied outside of the function and yield an equivalent result. The following is an example when x = 1 and y =2.

$$f(1 \cdot 2) = f(1) \cdot f(2)$$

$$(1 \cdot 2)^2 = 1^2 \cdot 2^2 = 4$$

By restricting the universe of the homomorphic function, it is possible to support a secure encryption method. An example of restricting the universe of a homomorphic function:

$$(\mathbb{Z}\ mod\ p, +): f(x) = x + p$$

$$(2)\ \%\ 5 = 2$$

$$(3)\ \%\ 5 = 3$$

$$(2+2)\ \%\ 5 = 4$$

$$(2+4)\ \%\ 5 = 1$$

This function exhibits a property where the set of values is bounded to a finite set no matter how large the value for *x*, the highest value of the set will be *p-1*. In this example it is the range of values {0, 1, 2, 3, 4}.

An example of applying this idea to encryption would be:

$$Enc(7) + Enc(21) = Enc(7 + 21)$$

This means that the addition of 7 and 21 produces an equivalent value when added inside or outside of the encryption function.

This property of homomorphic functions was described by Rivest, Adleman, & Dertouzos (1978) as theoretically allowing for a new type of cryptosystem they called “Homomorphic Encryption” that can allow for certain limited operations over cipher text without requiring the secret key.

A Homomorphic Encryption scheme supports 3 functions: Enc(key, p) to encrypt plaintext p with a secret key, Dec(key, c) to decrypt ciphertext c with secret key, and Eval(op, x, y) to evaluate the result of an operator on operand x and operand y.

Early homomorphic Encryption schemes suffered several drawbacks including limiting support to a single operator (such as addition or multiplication). This limited its practical use in most all machine learning algorithms given the need for multiple types of operators. These initial Homomorphic Encryption systems are referred to in the literature as "Partial Homomorphic Encryption". Boneh, Goh, & Nissim (2005) provided a theorical framework for Homomorphic Public-key cryptography system, known as BGN, that supported both multiplication and addition. In the BGN scheme, an unlimited number of addition operations can be conducted on the cipher text, but only one multiplication operation can be conducted. This limitation meant that once a multiplication operation was performed, the cipher text would need to be "refreshed" by decrypting and encrypting the data with the secret key. This process is described in the literature as "re-encrypt" or "refreshing" the cipher text (Chabanne, Wargny, & Milgram et al, 2017; Gentry, 2009; Boneh et al, 2005).

The reason for this limitation involves an inherent limitation in the homomorphic properties where each operation on the cipher text introduces some amount of noise. Operations such as multiplication introduce a lot of noise. When the noise reaches a specific threshold, the cipher text can no longer be decrypted with the secret key. The process of refreshing the cipher text, allows for the data to be decrypted prior to reaching the threshold and then encrypting to produce a fresh (or clean) cipher text (Hesamifred et al, 2018; Gentry, 2009). These Homomorphic Encryption schemes that limited the number of operations that could be conducted are known in the literature as "Somewhat Homomorphic Encryption" or SWHE.

A major advancement for Homomorphic Encryption occurred after the seminal work by Gentry (2009) at IBM Watson Research, where he proved the feasibility of homomorphic encryption for any mathematical operators without limiting the number of operations. This scheme is referred to as "Fully Homomorphic Encryption" (FHE) in the literature. In theory, this meant that there would be no need for refreshing the cipher text during deep computation. Gentry's (2009) work served as a catalyst for a new area of active research using homomorphic encryption in machine learning to provide privacy of sensitive data during the training or classification phases of a system.

One major drawback of the FHE scheme that was demonstrated to be impractical for real-world application centers around the technique used by (Gentry, 2009) known as "bootstrapping". Bootstrapping is a critical aspect that allows for taking a Somewhat Homomorphic Scheme (i.e. limited number of operations) and producing FHE scheme by removing the noise produced during an operation without the need for a secret key (Chabanne et al, 2017; Gentry, 2009). Unfortunately, this technique was computational expensive making the FHE scheme very inefficient and not practical for real-world use.

Gentry & Halevi (2011) provided an implementation of FHE which was improved by Halevi & Shoup (2014) and open sourced as "HELib". HELib has become a popular and widely used implementation of FHE by most privacy-preserving machine learning methods today. However, in order to workaround the inefficiency of the bootstrapping algorithm, most Homomorphic Encryption-based Machine Learning methods either limit the depth of computation (i.e. the number of consecutive operations on a ciphertext) or they perform a re-encryption to

refresh the ciphertext (Xie, Bilenko, Finley, et al, 2014; Zhang, Yang, & Chen, 2015; Shokri & Shamitkov, 2015; Gilad-Bachrach, Dowlin, Laine et al, 2016; Hesamifard et al, 2017).

Additional improvements to FHE were made by Gentry, Halevi, & Smart (2012) where they improved the bootstrapping technique and by Brakerski, Gentry, & Vaikuntanathan (2014) where they demonstrated an FHE scheme that did not require bootstrapping known as the BGV scheme. The BGV scheme allowed for unlimited addition and multiplication without bootstrapping but with a trade-off that results in a progressive deterioration of runtime efficiency as the depth of computation increased. This means that with each consecutive operation the computation time will increase. This survey did not identified any recent significant improvements of Homomorphic Encryption schemes since the BGV contribution by Gentry et al (2014).

Brief Review of Classical Machine Learning Methods using Homomorphic Encryption:

Initial work by Chaudhuri & Monteleoni (2009) demonstrated the feasibility of using Homomorphic encryption to train and classify with a logistic regression model and (Graepel, Lauter, & Naehrig; 2012) demonstrated feasibility training linear binary classifiers.

Aslett, Esperança, & Holmes (2015) further demonstrated the practical feasibility of using homomorphic encryption by implementing complex algorithms such as random forest and naïve Bayes classifiers and achieving competitive accuracy results when compared to training with an equivalent but unencrypted dataset. Their work claimed to be the first demonstration of homomorphic machine learning that kept the data encrypted throughout the training process and without requiring a multi-party computation, such as a trusted intermediary.

González-Serrano, Amor-Martín, & Casamayón-Antón (2018) proposed the use of a multi-party framework, that includes a data owner, algorithm owner, and a semi-trusted 3rd party cryptographic provider. In their system, the algorithm owner can invoke remote computation to the cryptographic provider which can perform secure operations over the data and return the results to the algorithm owner for algorithm training. This approach uses multiple private keys by the data owner to encrypt the data and assumes that the 3rd party cryptographic provider is a semi-trusted actor. The use of multi-party privacy-preserving method is considered suboptimal and introduces additional threat vectors that can compromise the sensitive data if there is collision between the algorithm owner and the cryptographic provider.

Kim, Song, Kim, Lee, & Cheon (2018) demonstrated a more computationally efficient use of homomorphic encryption for training logistic regression models by implementing a variation of Nesterov's accelerated gradient decent that reduces the number of iterations needed to converge on an optimal set of parameters. They demonstrated that their model can be trained in 6 minutes on a dataset of 1,579 samples with 18 features used to fit a logistic regression model that can predict a binary classification.

Shen, Tang, Zhu, Du, & Guizani (2019) demonstrated a solution that did not require a trusted 3rd party by employing a distributed ledger (blockchain) for data sharing and data integrity. Their work featured a modified version of the Support Vector Machine (SVM) algorithm that relies on homomorphic properties of Paillier cryptosystem in order to support the required mathematical operations by SVM. Their solution not only ensured that the sensitive data of the data owners was protected, but that the algorithm parameters of the algorithm owner is also protected.

Liu, Deng, Choo, & Yang (2019) present a different paradigm that involves multiple encrypted domains where multiple data owners each encrypt their data with a key and during training time, various mathematical computations required as part of the learning algorithm are actually performed within the domain of the data owner. They demonstrated their approach by implementing a Reinforcement Learning algorithm that can learn using patient data where each patient has their own encryption key for their data. This approach has the disadvantage of requiring each data owner to also provide an environment where computation can be performed and results returned to an external environment controlled by the algorithm owner.

Modern methods of machine learning involve the user of word embeddings that have been demonstrated to produce higher accuracy on natural language processing (NLP) tasks. Word embeddings (Mikolov et al, 2013) require that the words within a training set be encoded and looked-up within an embedding matrix.

Review of Neural Networks using Homomorphic Encryption

This section performs a comprehensive review of methods using Neural Networks to perform classification of Homomorphically Encrypted data using a pre-trained model. The pre-trained model was trained using plaintext. These methods include both standalone and multi-party computation schemes (SMC). The following section will discuss methods for training Neural Networks using Homomorphically Encrypted data.

The earliest known attempt to build a classifier using Neural Networks and Homomorphic Encryption begins with Barni, Orlandi, & Piva (2006) where they implement a small Neural Network that was previously trained on plaintext data and then adapted to an early Homomorphic Encryption scheme called Pailliers cryptosystem. Their approach required the use of a Secure Multi-Party Computation scheme that required the client that possesses the encryption key to be available during the execution of the classification algorithm. Their method performed all the feed-forward propagation of the neural network weights stopping short of applying the non-linear activation function. Because Homomorphic Encryption does not support complex functions such as Sigmoid, tanh, or ReLU, all considered popular choices today, Barni et al (2006) delegated the activation function operation to the client by sending the intermediate weights over the network so that the client could decrypt, apply the activation function, and return the results to the server to continue the forward propagation. This presented many drawbacks that made this approach impractical for real-world use. The amount of communication between client and server resulted in an inefficient system. Another drawback involved the potential for revealing critical information about the design and parameters of the model when the intermediate data is returned to the client.

Orlandi, Piva, & Barni (2007) made some minor improvements on the work by Barni et al (2006) by addressing the leakage of model information by masking the data sent back to the client, but still relied on the same general approach which deemed it inefficient and not for practical use.

After a brief period of no advancement, Gentry (2009) revitalizes the interest in Homomorphic Encryption with his seminal work, discussed earlier, that proved the feasibility of Fully Homomorphic Encryption (FHE) and later led to an implementation known as HELib (Halevi & Shoup, 2014; Gentry & Halevi, 2011). This served as a turning point for several key achievements in privacy-preserving methods using FHE for Neural Networks.

A collaborative solution, known as CryptoNets, by Microsoft Research and Carnegie Mellon University, presented a theoretical framework for a standalone Neural Network that uses FHE encrypted data and did not require multi-party computation (Xie, Bilenko, Finley, et al, 2014). The implementation of CryptoNets by Gilad-Bachrach, Dowlin, Laine et al (2016) was the first to demonstrate a standalone Neural Network classifier that did not use multi-party computation and leverage FHE encrypted data. They designed a 2-layer Convolutional Neural Network (CNN) that could classify images of numbers. This classification task is a well-known benchmark using the MNIST dataset. They demonstrated an accuracy of 98.95% using the secured version of the model, falling short of the 99.77% state-of-the-art performance achieved using conventional (unsecure) Neural Network.

In their approach, Gilad-Bachrach et al (2016) had to workaround several previously discussed limitations of using FHE for computation. Because FHE does not support complex activation functions, delegating this computation to a client was the previously method of choice with Orlandi et al (2007) and Barni et al (2006). Instead, Gilad-Bachrach et al (2016) experimented with several alternative polynomial functions and eventually chose to substitute the activation function with the square function: x^2. After experimenting with high-degree polynomials, they found that limits on computational depth imposed by FHE operations would prevent a better approximation and traded off accuracy for efficiency in this experiment.

Due to the selection of the square function as the activation function and the computational depth limits of FHE, their model was not viable beyond 2 hidden layers. However, their impressive achievement served as another catalyst for additional research interest in this space.

Chabanne, Wargny, Milgram, et al (2017) improved the state-of-the-art beyond CryptoNets accuracy and efficiency and were the first to produce a privacy-preserving Neural Network with more than 2 hidden layers. Their approach, like CryptoNets, was compatible with FHE but differed in two ways. First they identified better polynomial approximation functions for the traditional non-linear activation functions, in this case ReLU. Second, they leveraged batch normalization process (Loffe & Szegedy, 2015) to accelerate the propagation through the Neural Network. They attempted, but failed, to produce a trainable Neural Network using FHE, but achieved a new state-of-the-art in accuracy of 99.3% when measured with the MNIST benchmark. An improvement over the previous state-of-the-art in accuracy of 98.95% by CryptoNets.

Chabanne et al (2017) were also the first to leverage the BGV scheme of FHE by Brakerski et al, (2014), discussed in an earlier section, that removed the limits on computational depth for multiplication and addition and without incurring the heavy computation of the original FHE bootstrapping technique by (Gentry, 2009). These additional benefits provided by BGV allowed for the additional layers in their CNN and higher degree polynomial approximation.

Chabanne et al (2017) used a Taylor series polynomial approximation of the ReLU which resulted in a 6 degree polynomial that increased the computational depth required to evaluate the function and therefore resulted in poor runtime efficiency due to BGV's reduced performance as consecutive FHE operations are performed.

Hesamifard, Takabi, & Ghasemi (2017) continues the focus on improving the polynomial approximation of the common activation functions by performing an exhaustive evaluation of approximation functions that can both yield the optimal accuracy and efficiency of the model. Their system, called CryptoDL, achieved a new state-of-the-art accuracy on the MNIST

benchmark, achieving an accuracy of 99.52% that is within .04% of the best known state-of-the-art for the conventional (unsecure) accuracy of 99.56%. Their model architecture featured a CNN architecture with 3-hidden layers that was previously trained on plaintext and adapted to evaluate FHE encrypted data.

CryptoDL (Hesamifard et al, 2017) demonstrated that the use of low-degree polynomials could be used to achieve high accuracy, an open research question at the time of their work. They evaluated several approximation methods including using conventional Numerical methods, Taylor series, Standard Chebyshev, Modified Chebyshev, and ultimately their own approach which proved to yield the best model performance.

They reported the following model accuracy for each respective approximation method for ReLU: Taylor series – 40.28%, Standard Chebyshev – 68.98%, Modified Chebyshev – 88.53. They concluded that Taylor series produced high-degree polynomials and performed poorly for points that fell outside the approximation interval. Standard and Modified Chebyshev had better accuracy than Taylor but exhibited a characteristic where values below a certain interval would drop and then increase. This behavior was shown to confuse the Neural Network since it introduced ambiguity where two distant points could produce the same value (see Figure 2 below). This led to a new approach.

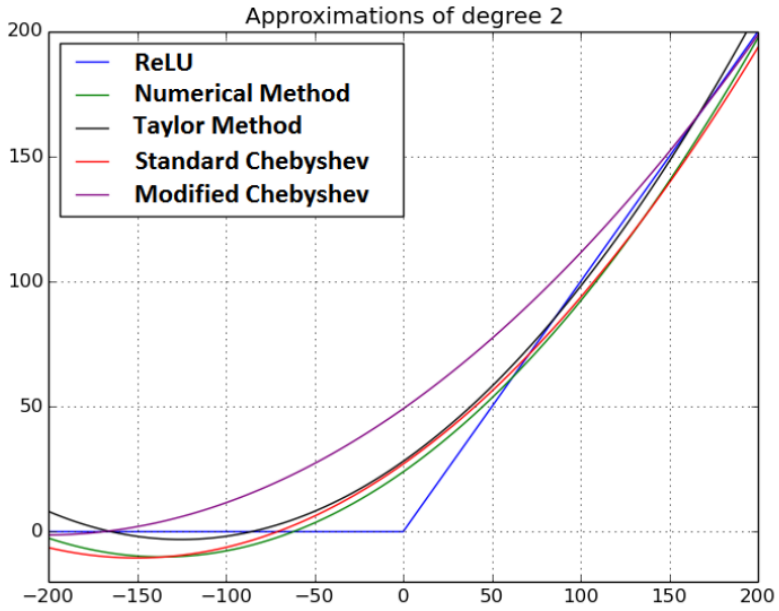

Figure 2 – polynomial approximation of ReLU (Hesamifard et al, 2017)

Their polynomial approximation method was based on an insight that the approximation of the activation function was not as important as the approximation of the derivative of the activation function. This meant that the previous evaluation framework for evaluating polynomial approximation of activation function was not directly representing the best performance within the model. Hesamifard et al (2017) observed that the polynomial approximation of the Sigmoid function, was very close to the derivative of the ReLU function. This was a significant finding that directly contributed to the higher performance of their model. Figure 3 shows the proposed polynomial approximation for ReLU compared to the original ReLU function. Note that the

proposed method does not closely follow the original ReLU but does not exhibit the ambiguity produced by the other approximation methods leading to a better model.

This approximation was a low-degree (2 degree) polynomial which also made it efficient for computing within the depth limits of FHE. As a result, their CryptoDL system also outperformed other previously reported prediction rates achieving 164,000 predictions per hour.

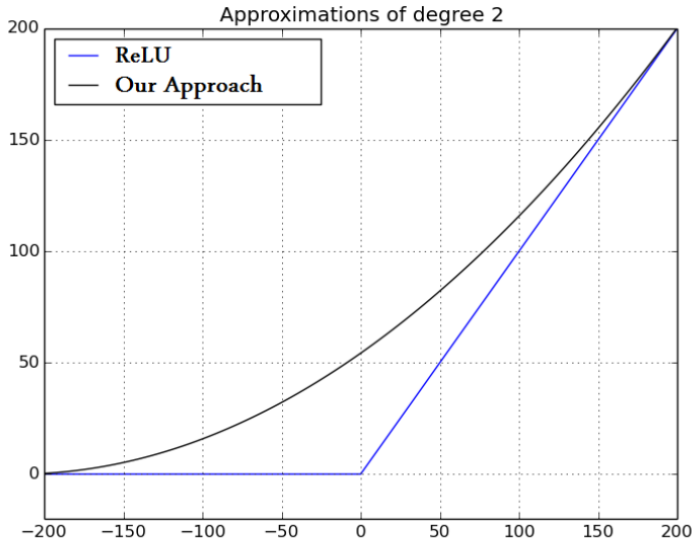

Figure 3 - proposed polynomial approximation by (Hesamifard et al) 2017

More impressive yet, was their demonstration of being able to train their CryptoDL CNN model using FHE encrypted data. This was a significant achievement. However, their model did not perform as well when they trained over ciphertext data achieving an accuracy of 91.5% when evaluated using the MNIST benchmark. Nevertheless, their work demonstrated key contributions to polynomial approximation of activation functions that would be leveraged future models.

In (Hesamifred, Takabi, & Ghasemi et al, 2018) and (Hesamifred, Takabi, & Ghasemi, 2019) they make an incremental improvement to CryptoDL by using bifurcated system of functions that optimizes the approximation function based on an interval. This allows for more than one polynomial function for a given activation function leading to a more optimal approximation. In their work they also provide a general algorithm (Figure 4) for generating polynomial approximation using this bifurcated system of functions approach. Their method is based on identifying various inflection points of the target function by identifying the min, max, and mean and constructing multiple polynomials that optimize around the intervals for [-min, min], [-max, max], and [-mean, mean].

Input: Dataset, MaxDegree, ActivationFunction
Output: A set of Polynomials

$MeanArray \leftarrow \{\}$;
$Mean = 0$;
$PolynomialSet \leftarrow \{\}$;
for $i \leftarrow 1$ **to** $\#Features$ **do**
  $mean \leftarrow$ mean of $Feature[i]$;
  $MeanArray \leftarrow MeanArray \cup \{mean\}$;
end
$Mean \leftarrow$ mean of $MeanArray$;
for $i \leftarrow 1$ **to** $MaxDegree$ **do**
  $poly \leftarrow$ Approximate the ActivationFunction in the interval $[-Mean, Mean]$ with precision 10 and degree i;
  $PolynomialSet \leftarrow PolynomialSet \cup \{poly\}$;
end
return $PolynomialSet$;

Figure 4 - General algorithm for polynomial approximation (Hesamifard et al, 2018)

This method of fails to surpass the accuracy previously achieved by (Hesamifard et al, 2017) resulting in an accuracy of 99.25% when evaluated with the MNIST benchmark but still surpasses all other methods. The contribution of this work is seen as providing a general method for providing approximation to any activation function within an FHE Neural Network scheme.

Brief Review of Training Neural Networks using Homomorphic Methods

A vital part of training a Neural Network involves the ability to perform back-propagation where the weights of the neurons are adjusted to optimize a loss-function. This process requires the ability to perform computation of a loss function using only the supported homomorphic encryption operators multiplication and addition.

The earliest known work to attempt a privacy-preserving approach to training a Neural Network model was by Chen & Zhong (2009) where they leveraged a multi-party computation scheme that used a special Back-propagation algorithm that sent ciphertext to the client where the data was updated and sent back to the server. This approach had the drawback of exposing the details of the model algorithm and was very inefficient due to the amount of communication between the client and server in order to train a reasonably sized network.

Yuan & Yu (2012) were the first to leverage BGN homomorphic encryption method (discussed in an earlier section) to perform back-propagation training of a neural network. Unfortunately, due to the limitations of BGN, namely the restriction of performing one multiplication operation before requiring a refresh, their approach continue to rely on a multi-party computational scheme. Although they were able to reduce the number of interactions between the

client and the server by using the BGN scheme, the method was not practical for reasonably sized networks.

Samet & Miri (2012) demonstrated an alternative but similar approach that proposed a more efficient back-propagation alternative to further reduce the amount of interactions, but this approach only worked for networks with a single hidden layer and therefore not practical to achieve current state-of-the-art performance on most tasks.

Takabi & Hesamifard (2016) demonstrated that using a polynomial approximation for the activation and loss functions, more of the computation could be performed securely on the server. Their approach was able to successfully train a small neural network that used a basic distance-squared function as a loss function to perform the updates. Unfortunately, due to the limits on computational depth imposed by FHE, the ciphertext required periodic refresh that incurred a client-server round trip and therefore affected the efficiency of the overall process.

The most recent attempt to train a Neural Network with FHE was by Hesamifard et al (2017) where they achieved an accuracy of 91.5% using the FHE trained method. This model featured a CNN with 3 layers and made use of their polynomial approximation of activation functions discussed at length in the previous section. In the Future Work section, a discussion is presented on the barriers and opportunities remaining for training Neural Networks using FHE schemes.

Comparative Evaluation of Methods

The evaluation of privacy-preserving Neural Network using Homomorphic encryption is evaluated here using 3 primary metrics: accuracy, efficiency, and confidentiality (Chabanne et al, 2017). At the time of this review, the state-of-the-art performance across all 3 metrics was achieved by (Hesamifard et al, 2019) which incrementally improved the work in (Hesamifard et al, 2018) and (Hesamifard et al, 2017). This work demonstrated the highest accuracy when evaluated using the MNIST benchmark. This benchmark measure the overall accuracy of a model when classifying images of numbers.

The following table summarizes the best performing models surveyed:

Method Name	**Authors**	**Standalone or SMC**	**Classify or Both (Classify and Train)**	**MNIST Benchmark Accuracy**	**MNIST Benchmark Runtime**

CryptoDL	**Hesamifard et al, 2017**	**Standalone**	**Both**	**99.52%**	**320s**
-	**Chabanne et al, 2017**	**Standalone**	**Classify**	**99.3%**	**-**
CryptoNets	**Gilad-Bachrach et al, 2016**	**Standalone**	**Classify**	**98.95%**	**697s**
CryptoDL	**Hesamifard et al, 2019**	**Standalone**	**Both**	**99.2%**	**458s**

The CryptoDL work by Hesamifard et al (2017) outperformed the other methods in both accuracy and runtime efficiency when measured against the MNIST benchmark. Note that Hesamifard et al's (2017) work outperformed their own successors of CryptoDL. This was due to a change in focus from achieving the highest accuracy to focusing on better generalization of polynomial approximation for any activation using a proposed algorithm.

Future Work

The study of privacy-preserving methods is an active and robust field of study that will undoubtedly continue as we head into an era of unprecedented connectivity and collection of data. The need to leverage this data legally and ethically will continue to drive a proliferation of Artificial Intelligence methods including the expanded use of Neural Networks trained to provide more precise and accurate results to individuals across industry sectors including most notably: healthcare. As a result, having methods that allow technology providers to build models that leverage sensitive data in a secure way without revealing the data is paramount to the privacy of individuals around world.

In this section we discuss several key underlying problem areas that are worthy of additional investigation and can potentially address current barriers and challenges encountered by the current state-of-the-art methods reviewed earlier. First set of barriers involve existing limitations of Homomorphic Encryption schemes which directly contribute to suboptimal accuracy and computational efficiency of models during training or classification phases. The impact of model accuracy can be attributed to the lack of support for more complex functions, such as well-known activation functions like tanh, sigmoid, and ReLU. Currently all Homomorphic Encryption schemes are limited to just multiplication and addition. This limitation requires the use of polynomial approximation which results in expensive and inefficient computation in order to match the performance of complex non-linear activation functions.

Another barrier results from the boundary on computational depth when performing Homomorphic operations over ciphertext. This results in either limiting activation functions to low-degree polynomials which reduces the accuracy of the model or the need for multi-party computing as a workaround to re-encrypt (refresh) the ciphertext requiring communication with an external system and introducing additional latency into the algorithm's computation. These limitations have also restricted the number of layers within the model and therefore limiting the

ability for the model to learn complex patterns hidden within the feature space and resulting in lower accuracy.

Another potential for model accuracy improvement involves the approximation of activation functions. Many of the contribution by the reviewed methods in this survey involve incremental improvements to the performance of approximation techniques. Most of these techniques involve the use of polynomial approximation. Due to the inherent limitations on computational depth, these methods have required low-order polynomials that must trade-off between accuracy and efficiency of the model.

Lastly, a relatively small amount of focus has been given to training neural networks using homomorphic encryption. Several methods reviewed demonstrated the feasibility for simple neural networks such as Multi-Layer Perceptron (Feed-forward) layers. In practice, obtaining enough plaintext data without sensitive information in order to pre-train a model is very challenging. This results in a major impediment to the applicability of these methods if there is no data available to begin with. As such, focus should be given to improving the state-of-the-art of training Deep Neural Networks using encrypted data to obtain data that would otherwise be unavailable due to its sensitive nature. This method of training (or learning a model) introduces several restrictions on the ability for the model developer to observe the accuracy of the model and tune or relabel data. A possible hybrid approach worth exploring involves pre-training a model using available data that can be inspected and performing incremental learning (sometimes known as transfer learning) with blind data (i.e. encrypted data) that cannot be inspected.

Any improvements to the problem areas outlined above will result in meaningful progress in privacy-preservation methods and Machine Learning and will most certainly play a vital role in shaping the future of how and where Artificial Intelligence is used.

References

Hesamifard, E., Takabi, H., & Ghasemi, M. (2019, March). Deep neural networks classification over encrypted data. In Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (pp. 97-108).

Liu, X., Deng, R., Choo, K. K. R., & Yang, Y. (2019). Privacy-preserving reinforcement learning design for patient-centric dynamic treatment regimes. IEEE Transactions on Emerging Topics in Computing.

Shen, M., Tang, X., Zhu, L., Du, X., & Guizani, M. (2019). Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities. IEEE Internet of Things Journal, 6(5), 7702-7712.

González-Serrano, F. J., Amor-Martín, A., & Casamayón-Antón, J. (2018). Supervised machine learning using encrypted training data. International Journal of Information Security, 17(4), 365-377.

Hesamifard, E., Takabi, H., Ghasemi, M., & Wright, R. N. (2018). Privacy-preserving machine learning as a service. Proceedings on Privacy Enhancing Technologies, 2018(3), 123-142.

Kim, A., Song, Y., Kim, M., Lee, K., & Cheon, J. H. (2018). Logistic regression model training based on the approximate homomorphic encryption. BMC medical genomics, 11(4), 83.

Al, M., Chanyaswad, T., & Kung, S. Y. (2018, April). Multi-Kernel, Deep Neural Network and Hybrid Models for Privacy Preserving Machine Learning. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 2891-2895). IEEE.

Wang, Q., Du, M., Chen, X., Chen, Y., Zhou, P., Chen, X., & Huang, X. (2018). Privacy-preserving collaborative model learning: The case of word vector training. IEEE Transactions on Knowledge and Data Engineering, 30(12), 2381-2393.

Chabanne, H., de Wargny, A., Milgram, J., Morel, C., & Prouff, E. (2017). Privacy-preserving classification on deep neural network. IACR Cryptology ePrint Archive, 2017, 35.

Chanyaswad, T., Chang, J. M., & Kung, S. Y. (2017, May). A compressive multi-kernel method for privacy-preserving machine learning. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 4079-4086). IEEE.

Hesamifard, E., Takabi, H., & Ghasemi, M. (2017). Cryptodl: Deep neural networks over encrypted data. arXiv preprint arXiv:1711.05189.

Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., & Wernsing, J. (2016, June). Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In International Conference on Machine Learning (pp. 201-210).

Takabi, H., Hesamifard, E., & Ghasemi, M. (2016, December). Privacy preserving multi-party machine learning with homomorphic encryption. In 29th Annual Conference on Neural Information Processing Systems (NIPS).

Aslett, L. J., Esperança, P. M., & Holmes, C. C. (2015). Encrypted statistical machine learning: new privacy preserving methods. arXiv preprint arXiv:1508.06845.

Shokri, R., & Shmatikov, V. (2015, October). Privacy-preserving deep learning. In Proceedings of the 22nd ACM SIGSAC conference on computer and communications security (pp. 1310-1321).

Zhang, Q., Yang, L. T., & Chen, Z. (2015). Privacy preserving deep computation model on cloud for big data feature learning. IEEE Transactions on Computers, 65(5), 1351-1362.

Brakerski, Z., Gentry, C., & Vaikuntanathan, V. (2014). (Leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory (TOCT), 6(3), 1-36.

Halevi, S., & Shoup, V. (2014, August). Algorithms in helib. In Annual Cryptology Conference (pp. 554-571). Springer, Berlin, Heidelberg.

Xie, P., Bilenko, M., Finley, T., Gilad-Bachrach, R., Lauter, K., & Naehrig, M. (2014). Crypto-nets: Neural networks over encrypted data. arXiv preprint arXiv:1412.6181.

Graepel, T., Lauter, K., & Naehrig, M. (2012, November). ML confidential: Machine learning on encrypted data. In International Conference on Information Security and Cryptology (pp. 1-21). Springer, Berlin, Heidelberg.

Gentry, C., Halevi, S., & Smart, N. P. (2012, May). Better bootstrapping in fully homomorphic encryption. In International Workshop on Public Key Cryptography (pp. 1-16). Springer, Berlin, Heidelberg.

Samet, S., & Miri, A. (2012). Privacy-preserving back-propagation and extreme learning machine algorithms. Data & Knowledge Engineering, 79, 40-61.

Yuan, J., & Yu, S. (2012, September). Privacy preserving back-propagation learning made practical with cloud computing. In International Conference on Security and Privacy in Communication Systems (pp. 292-309). Springer, Berlin, Heidelberg.

Gentry, C., & Halevi, S. (2011, May). Implementing gentry's fully-homomorphic encryption scheme. In Annual international conference on the theory and applications of cryptographic techniques (pp. 129-148). Springer, Berlin, Heidelberg.

Chaudhuri, K., & Monteleoni, C. (2009). Privacy-preserving logistic regression. In Advances in neural information processing systems (pp. 289-296).

Chen, T., & Zhong, S. (2009). Privacy-preserving backpropagation neural network learning. IEEE Transactions on Neural Networks, 20(10), 1554-1564.

Gentry, C. (2009, May). Fully homomorphic encryption using ideal lattices. In Proceedings of the forty-first annual ACM symposium on Theory of computing (pp. 169-178).

Aggarwal, C. C., & Philip, S. Y. (2008). A general survey of privacy-preserving data mining models and algorithms. In Privacy-preserving data mining (pp. 11-52). Springer, Boston, MA.

Schlitter, N. (2008). A protocol for privacy preserving neural network learning on horizontal partitioned data. PSD.

Orlandi, C., Piva, A., & Barni, M. (2007). Oblivious neural network computing via homomorphic encryption. EURASIP Journal on Information Security, 2007, 1-11.

Barni, M., Orlandi, C., & Piva, A. (2006, September). A privacy-preserving protocol for neural-network-based computation. In Proceedings of the 8th workshop on Multimedia and security (pp. 146-151).

Boneh, D., Goh, E. J., & Nissim, K. (2005, February). Evaluating 2-DNF formulas on ciphertexts. In Theory of Cryptography Conference (pp. 325-341). Springer, Berlin, Heidelberg.

Agrawal, R., & Srikant, R. (2000, May). Privacy-preserving data mining. In Proceedings of the 2000 ACM SIGMOD international conference on Management of data (pp. 439-450).

Rivest, R. L., Adleman, L., & Dertouzos, M. L. (1978). On data banks and privacy homomorphisms. Foundations of secure computation, 4(11), 169-180.

Birkhoff, G. (1940). Lattice theory (Vol. 25). American Mathematical Soc..