Enabling Secure Intelligent Network with Cloud-Assisted Privacy-Preserving Machine Learning

Yong Yu, Huilin Li, Ruonan Chen, Yanqi Zhao, Haomiao Yang, and Xiaojiang Du

Abstract

Intelligent networks are regarded as existing networks incorporating some intelligent mechanisms such as cognitive and cooperative approaches to improve network performance. Security is highly essential in intelligent networks but has received less attention so far. In this article, we propose a framework that enables a secure intelligent network with the assistance of cloud-assisted privacy-preserving machine learning. In the framework, the cloud server can first generate a model using outsourced machine learning algorithms and then process testing data from the network with the generated model in real time, which reflects to the network and makes it more intelligent. At the same time, the proposal guarantees the security and privacy of both the training data and the testing data in the sense that the proposed framework takes advantage of differential privacy to perform privacy-preserving data analysis and homomorphic encryption to conduct valid operations over encrypted data. The performance evaluations of the core primitives employed in the framework including differential privacy and homomorphic encryption algorithms demonstrate the practicability of our proposal.

Introduction

Having the services loaded into switches in the traditional plain old telephone system complicates the introduction and management of sophisticated services. The growing demand for advanced user-oriented services and the desire to manage the network more cost effectively drive the evolution of a new networking architecture, known as intelligent networking (IN) [1].

IN is essentially an architectural concept for the provision, creation, and management of services that separates the service logic from the underlying physical switching system. The origin of IN can be traced back to 1986 when the basic concept was introduced in the IN/1 definition proposed by Regional Bell Operating Companies. In 1989, the European Telecommunications Standards Institute (ETSI) and the International Telecommunication Union (ITU) began to define the target IN architecture in accordance with the structured development process, aiming to promote the standardization of an international IN. They defined a particular capability set in each phase of evolution. A capability set mainly focuses on two aspects, namely service requirement and network requirement, including service creation, management, interaction, processing, network management, and interworking.

The Intelligent Network Conceptual Model (INCM) acts a pivotal part in the process of the target IN architecture, which serves as a complete framework for the design of capability sets. INCM is structured into four layers, and the close interrelation with each other depicting the engineering process of IN is portrayed in Fig. 1.

The top layer is the service plane (SP), where users and service providers can describe services without considering their implementation, which is a service-oriented view. The second layer is the global functional plane (GFP), consisting of basic call processing, two interaction points known as point of initiation (POI) and point of return (POR), and a set of service-independent building blocks (SIBs). Each SIB is a unit of functionality, and a chain of SIBs constitutes the service logic described in the SP. The distributed functional plane (DFP) is the third layer, enabling network designers to describe the functional architecture in a distributed view with a range of functional entities (FEs). Any given FE is composed of various functional entity actions (FEAs), and each FEA is performed by a series of elementary functions (EFs). Moreover, a sequence of FEAs and the information flows through them realize the SIB in the second layer. The bottom layer is the physical plane (PP), where multi-equipment vendors model the physical architecture with physical entities (PEs). Each FE from DFP is mapped to one or more PEs, driven by the upper-level service logic.

Traditional networking approaches associated with manual, reactive, and centrally administered operations are usually time-consuming and error-prone. However, the next generation network will be large-scale, complex, and heterogeneous. Thus, the traditional networking approaches are unsuitable for the next generation network. Facing the dilemma of data explosion but knowledge shortage, network operators try to optimize the network with the assistance of advanced data analytics such as machine learning (ML) and artificial intelligence (AI), which has attracted much attention from academia and industry so far. For example, ETSI is defining a cognitive network management architecture by using AI techniques and context-aware

Digital Object Identifier:
10.1109/MNET.2019.1800362

Yong Yu is with Shaanxi Normal University and State Key Laboratory of Cryptology; Huilin Li, Ruonan Chen, and Yanqi Zhao are with Shaanxi Normal University; Haomiao Yang is with the University of Electronic Science and Technology of China; Xiaojiang Du is with Temple University.

0890-8044/19/$25.00 © 2019 IEEE

policies to help operators automate network configurations. In order to build an automatic, proactive, self-aware, and predictive network, different levels of analytics such as system analytics, user and service analytics, and radio analytics have been adopted in particular scenarios. For instance, optimized subscriber-centric wireless offload improves the throughput and fairness of the network.

Our Contributions. The contributions of this article are summarized as follows:

1. We review the conceptual model and the working principle of the IN, in which ML technology is recommended to be used.
2. We investigate the security and privacy issues when using ML to optimize the network, and summarize the solutions to these issues.
3. We propose a secure cloud-IN framework in which the techniques of privacy-preserving ML are employed. The proposed framework takes advantage of differential privacy and homomorphic encryption to guarantee the privacy of both the training data and the testing data.

Organization. The rest of this article is organized as follows. We introduce ML and its privacy issues. We show some existing solutions to these issues. We describe our proposed cloud-IN framework. We implement some related core primitives in our framework. We then conclude this article.

Machine Learning and Its Privacy Issues

The next generation networks are not only service-driven but also data-driven. The increasing user demands and the emergence of IoT devices bring about data explosion, which drives the adoption of advanced data analytics to extract useful information. Machine learning techniques help network operators who can access large amounts of data to make the network more intelligent, especially in predictive analytics. For example, network operators can predict changes and adjust allocation strategies in real time in a variety of scenarios including traffic congestion prediction, balancing the load distribution, and so on.

Machine Learning

If dark clouds are gathering in the sky when you get ready to leave your home, you would carry your umbrella, which comes from your life experience. Computers could help humans to make judgments or estimates using "experience." From the perspective of computers, "experience" usually appears in the form of data. The purpose of ML is to learn how to perform a task by generating a model from the data. Let $D = \{x_1, x_2, \cdots, x_m\}$ represent a dataset containing m samples. Each sample x_1 is described by d attributes. $x_i = (x_{i1}; x_{i2}; \cdots; x_{id})$ denotes a vector of the d-dimension sample space $\mathcal{X}$, namely $x_i \in \mathcal{X}$, where x_{ij} denotes the jth attribute value of x_i.

Sample data is insufficient if you desire to generate a model to predict relevant results. The information result of the training sample is required, such as ((humidity; dark clouds; thunder), it will rain). The information result of the sample is called a "label," and the sample with a label is called "example." Generally, (x_i, y_i) indicates the ith sample, where $y_i \in \mathcal{Y}$ is the label of sample x_1, and $\mathcal{Y}$ is the set of all labels, which is called "output space" or "label space."

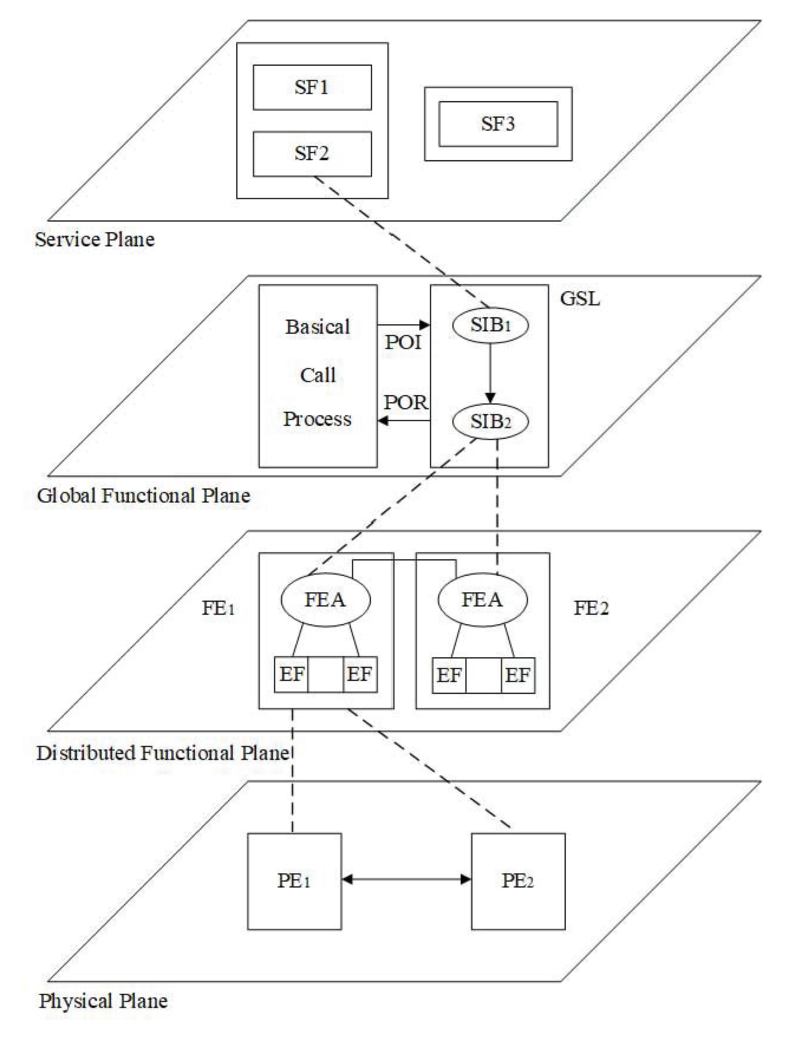


FIGURE 1. Intelligent network conceptual model.

Learning tasks can be divided into two categories according to whether the training data contains labels or not, namely, supervised learning and unsupervised learning. The former is represented by classification and regression; the latter is represented by clustering. If the result we desire to predict is some discrete values, such a learning task is called classification. If the result is continuous values, the learning task is called regression. Clustering is such a learning task that divides the training data into several groups, and the training samples usually do not contain labels.

Privacy Issues

In spite of being extensively used in diverse domains, ML brings about many security and privacy issues that are receiving increasing attention.

The common security threats in ML include poisoning attacks, adversarial attacks, and oracle attacks. Moreover, the collected data of users sometimes involve personal private information, which results in potential privacy leaking issues. In the training phase, attackers are able to access sensitive information by stealing training data. In the prediction phase, attackers are able to extract training data or object model information utilizing reverse attacks or membership inference attacks. The malicious access to the model and the expo-

The European Union issued the General Data Protection Regulation (GDPR) to punish the illegal use of data in 2018, which is claimed as the strictest data protection regulation in history. GDPR is used to punish dishonest behaviors after data leakage, but how to prevent data privacy from being leaked is an urgent but challenging issue.

sure of model parameters also raise privacy concerns. In 2015, Fredrikson *et al.* recovered a face image used in the training phase successfully by exploiting confidence information revealed along with predictions [2].

The European Union issued the General Data Protection Regulation (GDPR) to punish the illegal use of data in 2018, which is claimed as the strictest data protection regulation in history.[1] GDPR is used to punish dishonest behaviors after data leakage, but how to prevent data privacy from being leaked is an urgent but challenging issue.

Tools for Solving the Issues

Privacy-preserving data analysis techniques can be adopted to address these issues. In this section, we review some primitives including differential privacy, homomorphic encryption, deterministic encryption, and frequency-smoothing encryption that can be used to guarantee the data privacy in ML.

Differential Privacy

Differential privacy [3] is a powerful tool to perform privacy-preserving data analysis, which has already been applied to Apple's iOS operating system and Google's Chrome browser. It permits curators to carry out benign aggregate analysis while providing meaningful protection of the privacy of each individual. In a statistical database, differential privacy guarantees that the removal or addition of a single item does not essentially affect the outcome of any analysis. The formal definition of differential privacy [3] is as follows.

A randomized function K gives ε-differential privacy if for all data sets D and D' differing on at most one element, and all $S \subseteq Range(K)$,

$$\Pr[K(D) \in S] \leq \exp(\varepsilon)\Pr[K(D') \in S],$$

where ε is the privacy risk factor.

Differential privacy has the following properties.

Composability. If two queries are answered with ε_1 and ε_2 for differential privacy level, then the pair of queries is $(\varepsilon_1 + \varepsilon_2)$ differential privacy level.

Post-processing. Whatever processes are performed on the results of a differential privacy algorithm, the processed results are still differentially private.

Differential privacy techniques were suggested to be applied to ML due to its nice properties. However, the introduction of noise reduces the availability of the model; thus, how to balance the data privacy and model accuracy has become a hot research topic. In 2017, Ligett *et al.* [4] proposed a general "noise reduction" framework to maximize the privacy level under the fixed accuracy requirement.

Homomorphic Encryption

Homomorphic encryption allows anyone (not just the key holder) to conduct certain algebraic operations on the ciphertext c, which is equivalent to performing corresponding operations on plaintext m. Mathematically, given a homomorphic encryption function E, and two plaintexts m_1, m_2, one can publicly compute $E(m_1 * m_2) = E(m_1) \circ E(m_2)$. There are some well-known homomorphic encryption schemes such as ElGamal encryption and Paillier encryption. Ehsan *et al.* [5] utilized homomorphic encryption to protect the dataset privacy, where the activation functions in the neural network were replaced with polynomial approximation.

Other Techniques

In 2007, Bellare *et al.* introduced deterministic encryption [6], which allows to search encrypted data. Deterministic encryption enables semantic security and at the same time provides as-strong-as-possible privacy, where the encryption algorithm is deterministic.

However, classical frequency analysis attack on deterministic encryption is possible. If the distribution of plaintext is not uniform, an adversary who has a reference dataset can calculate the expected plaintext frequencies. When an adversary can access a snapshot of encrypted data, they can match the frequency in the encrypted domain to the plaintext domain. In order to resist inference attacks caused by frequency analysis while protecting data privacy, Lacharit *et al.* introduced frequency-smoothing encryption [7], which extends deterministic encryption to somewhat randomized encryption.

Secure Cloud-Intelligent Network Framework

In this section, we propose a secure cloud-IN architecture assisted by privacy-preserving ML.

With the advances of cloud computing, the techniques of integrating on-site applications with cloud services have been extensively investigated. The cloud-based IN optimizes the network performance by taking advantage of cloud computing capacity, which could dynamically adjust resource allocations as needed. Driven by the demand to reduce network overhead and improve scalability, predictability, as well as adaptability of the network, ML as a service is offered to perform ML algorithms on the cloud. However, the cloud is not fully trusted [8, 9], which leads to privacy issues in ML since the raw data accessed from the network are usually privacy-sensitive.

As shown in Fig. 2, the proposed cloud-IN framework consists of two parts, namely the network platform and the cloud:

- Network platform: The basic network platform of our proposed framework is composed of network-connected devices such as network cards, switches, routers, and transmission media in physics. To adapt the expansion of network scale and the diversity of network devices, we create a new virtualized networking layer which decouples the function of network layers from hardware and is programmatic provisioning.
- Cloud: The cloud servers in the proposed cloud-intelligent network framework have huge storage space and strong computing

[1] http://scikit-learn.org/stable/

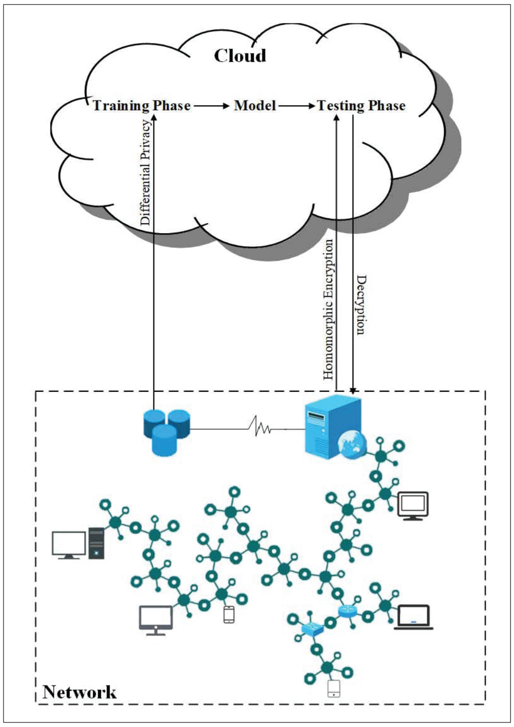


FIGURE 2. Secure cloud-intelligent network framework.

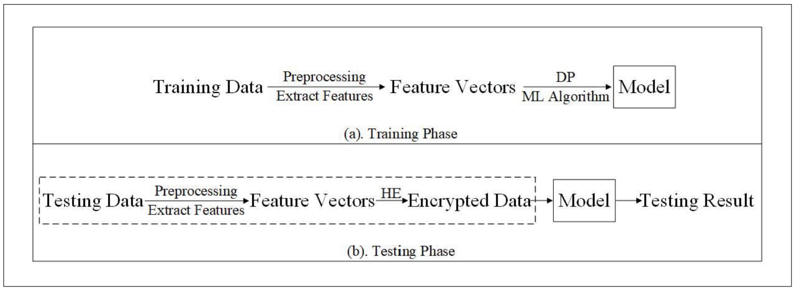


FIGURE 3. Outsourced machine learning tasks overview.

power, where the received data can be processed in real time and turned into quantifiable information. Privacy-preserving ML algorithms are performed on the cloud to improve both the privacy and value of data, thus making the network more intelligent.

Privacy-preserving ML is the core process of the framework. Considering the expensive computation cost of homomorphic encryption, we enforce differential privacy in the training phase to protect the training model and directly protect real-time data in the testing phase by homomorphic encryption, and in the training phase after data preprocessing including data cleaning, conversion, and clustering. The cloud servers first extract features of the uploaded data and assume a fixed privacy requirement ε according to the specific objectives. They then attempt to maximize the accuracy of the trained model. We perturb the objective function by adding noise and generate a noisy model, which prevents reverse attacks and membership inference attacks so as to protect the training data. The training phase is shown in Fig. 3a. In the testing phase, local servers or clients first preprocess the real-time data and then encrypt the data with homomorphic encryption. Then the cloud performs classification or prediction tasks over the encrypted data using the generated model, which is noisy but sufficiently accurate, and finally returns the encrypted results back to the underlying network. The testing phase is shown in Fig. 3b.

Implementation

In this section, we report the implementations with a living example of the proposed secure cloud-IN framework. We first show some implementation results of homomorphic encryptions. Then we implement a fundamental but important ML algorithm called a linear regression algorithm, including linear regression under differential privacy and linear regression over homomorphic encryption ciphertexts.

Implementation of Homomorphic Encryption

We give two typical implementations of homomorphic encryption (HE), including a full HE (FHE) named HELIB, and a somewhat vector HE named VHE. We conduct all simulations for HE algorithms on a laptop with i3-4130 CPU @ 1.40 GHz and 8 GB RAM running Ubuntu 14.04. HELIB and VHE are built over software packages in [10, 11], respectively, which both invoke the well-known high-performance number theory C++ Library (NTL) for arbitrary length integers.[2] We use gcc/g++ 4.8.4 to compile all used programs in our experiments.

In our experiments, the security parameters [12, 13] are set based on the following consideration. First of all, the security level is taken as $\lambda = 128$ to guarantee practical security. In order to compare the performance effectively, we need to set parameters for HELIB and VHE, respectively, as follows. For HELIB, the native plaintext space is $\mathbb{Z}[X]/(\Phi_m(X), p^r)$, where $m = 1025$, $p = 2$, $r = 10$. Then we set $L = 10$, $c = 2$, and $w = 64$, denoting the number of levels, the number of columns in key switching matrix, and the Hamming weight of a secret key, respectively. Finally, for the monic irreducible polynomial G over $\mathbb{Z}_p$, we only require that the degree of G divides the degree of the irreducible factors of $\Phi_m(X)$ modulo p. On the other hand, considering the associated VHE parameters, we assign $p = 10$ for the same plaintext space. Further, we specify that $\omega < 2^{30}$, $l = 50$, $q = 2^{128}$, and *eBound* = 200 to ensure the correctness of ciphertext operations [11].

Note that most scenarios using HE, such as privacy-preserving image processing and ML [14], are required to encrypt data items (or records) containing multiple attributes (or features). Consequently, a data record can be treated as a vector with m dimension. To improve efficiency, it is expected to encipher the vector in a batch manner, that is, record-wise, not attribute by attribute. Fortunately, both HELIB and VHE support the integer vector batch encryption. In addition, calculating inner product in ciphertext domains is an important operation for many security applications (e.g., the privacy-preserving similarity measure in top-k retrieval). Therefore, in the following, we give a comparison between HELIB and VHE in terms of time costs of the key generating, vector encryption ($m = 40$) and inner production calculation, as illustrated in Fig. 4.

To test the *KeyGen*, *VectorEnc*, and *Encrypted InnerProduct* algorithms of each scheme, we repeat each algorithm 100 times and get the

[2] http://shoup.net/ntl/

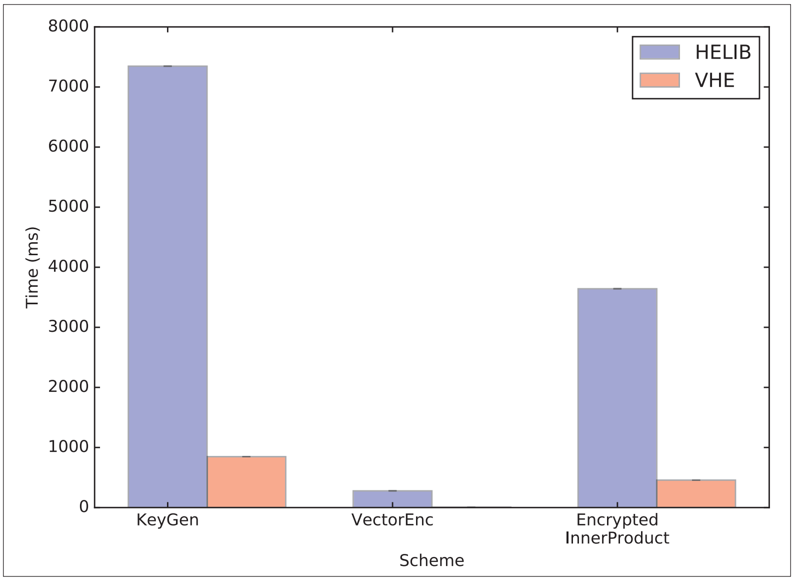


FIGURE 4. Time comparison.

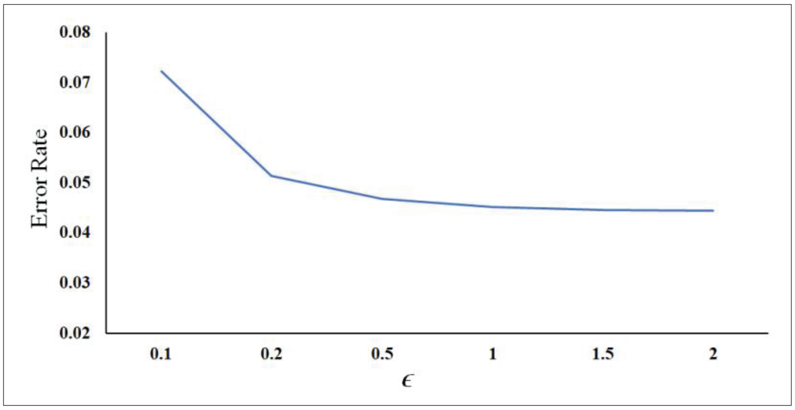


FIGURE 5. Implementation of linear regression under differential privacy.

average running time. As shown in Fig. 4, VHE is much more efficient than HELIB. For example, for security level $\lambda = 128$ and a vector with the dimension of $m = 40$, the time of *KeyGen*, *VectorEnc*, and *Encrypted InnerProduct* is about 7347 ms, 278 ms, and 3642 ms, respectively, in HELIB. Meanwhile, in VHE, the time is 847 ms, 5 ms, and 457 ms, respectively. It shows that VHE is orders of magnitude faster than HELIB. The reason is also obvious: VHE is specially designed for integer vector homomorphism, only requiring support for limited homomorphic operations, such as the addition, key switching, and inner product, while HELIB is developed for fully homomorphic encryption, which can evaluate arbitrary polynomial-depth circuits. To this end, HELIB has to pay extra cost to implement some expensive operations such as KeySwitch, ModulusSwitch, Re-linearity, and Bootstrapping.

Implementation of Linear Regression under Differential Privacy

To demonstrate the feasibility of enforcing differential privacy in our framework, we implement linear regression to predict the value of annual income in Brazil. In order to protect the training model, we apply the function mechanism [15] to perturb the objective function of the optimization problem rather than its results, which is a general framework for regression analysis under ε-differential privacy. The dataset loaded from Integrated Public Use Microdata Series[3] contains 190,000 records, and each record has 13 attributes. All the computations are conducted on the Matlab (version 2013b) on a desktop with 3 GHz Intel Core i5-7400 CPU and 8 GB RAM.

We set the index for dividing the dataset into training set and testing set to 0.8, the training epoch to 800, then change the value of privacy budget ε to 0.1, 0.2, 0.5, 1, 1.5, 2, respectively, and measure the error rate of the results, as shown in Fig. 5. Experiment results show that the value of error rate decreased with the increasing of privacy budget ε. Thus, in the training phase of the proposed privacy-preserving ML process, it is practical to dynamically adjust the value of ε and select the optimum one according to the specific dataset and objectives, and thus to reach high accuracy of results with less noise injected.

Implementation of Linear Regression over Homomorphic Encryption

We implement linear regression to predict house prices in Boston over homomorphic encrypted data, which is based on the TensorFlow,[4] scikit-learn,[5] and python-paillier[6] libraries. TensorFlow is an open source software library developed by Google using data flow graphs for high-performance numerical computation. Scikit-learn is a set of simple and efficient tools with various algorithms of ML for data mining and data analysis. Python-paillier is a partially homomorphic encryption library in python. All the computations are conducted on the Jupyter Notebook web application on a MacBook Pro with 2 GHz Intel Core i5 and 8 GB RAM.

We set the index for dividing the dataset into training set and testing set to 0.8, the value of learning rate to 0.01, and the training epoch to 800. The dataset we used has 506 samples and is loaded from sklearn. First, we train a model based on the training data and measure the root mean square error of the training process, as shown in Fig. 6a. After generating a linear regression model, we test the model over testing data, as shown in Fig. 6b. Then we use the generated model to perform prediction over encrypted data, pick 100 rows randomly, and encrypt them with the paillier encryption scheme from the python-paillier library as a new input. After that we decrypt the cipher predictions, as shown in Fig. 6c. Comparing the Fig. 6c with Fig. 6b, the predictions over encrypted data reach the expected demand. The simple implementation comes to the conclusion that HE is a practical method for privacy-preserving ML.

Conclusion

Machine learning can be used to promote network performance, and thus to make the network more intelligent. However, there are numerous security and privacy issues when machine learning is applied. In this article, we propose a secure cloud-intelligent network framework assisted by privacy-preserving machine learning. The proposal guarantees the privacy of both the training data and the testing data in the sense that it takes advantage of differential privacy to perform privacy-preserving data analysis and homomor-

[3] https://international.ipums.org/international/

[4] https://www.tensorflow.org

[5] http://scikit-learn.org/stable/

[6] https://python-paillier.readthedocs.io/en/develop/

phic encryption to conduct valid operations over encrypted data. The experimental evaluations of the core tools of the framework show its practicability. Our future work is to deploy the framework in real-world intelligent networks.

Acknowledgment

This work was supported by National Key R&D Program of China (2017YFB0802000), National Natural Science Foundation of China (61872229), National Cryptography Development Fund during the 13th Five-year Plan Period (MMJJ20170216) and Fundamental Research Funds for the Central Universities (GK201702004).

References

[1] O. Martikainen, J. Lipiäinen, and K. Molin, "Tutorial on Intelligent Networks," Lappeenranta Univ. Technology, 1994.

[2] M. Fredrikso, S. Jha, and T. Ristenpart, "Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures," *Proc. ACM CCS*, CO, 2015, pp. 1322–33.

[3] C. Dwork, "Differential Privacy: A Survey of Results," *Int'l. Conf. Theory and Applications of Models of Computation*, Xi'an, China, 2008, pp. 1–19.

[4] K. Ligett, S. Neel, and A. Roth, "Accuracy First: Selecting a Differential Privacy Level for Accuracy Constrained ERM," *Advances in Neural Info. Processing Systems*, Long Beach, CA, Dec. 2017, pp. 2566–76.

[5] E. Hesamifard *et al.*, "Privacy-Preserving Machine Learning as a Service," *Proc. Privacy Enhancing Technologies*, 2018, vol. 3, pp. 123–42.

[6] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and Efficiently Searchable Encryption," *Annual Int'l. Cryptology Conf.*, CA, 2007, pp. 535–52.

[7] M. S. Lacharit and K. G. Paterson, "Frequency-Smoothing Encryption: Preventing Snapshot Attacks on Deterministically Encrypted Data," *IACR Trans. Symmetric Cryptology*, vol. 1, 2018, pp. 277–313.

[8] Y. Li *et al.*, "Fuzzy Identity-Based Data Integrity Auditing for Reliable Cloud Storage Systems," *IEEE Trans. Dependable and Secure Computing*, vol. 16, no. 1, 2019, pp. 72–83.

[9] Y. Yu *et al.*, "Assured Data Deletion with Fine-Grained Access Control for Fog-Based Industrial Applications," *IEEE Trans. Industrial Informatics*, vol. 14, no. 10, 2018, pp. 4538–47.

[10] S. Halevi and V. Shoup, "Helib," retrieved from HELib; https://github.com.shaih/HElib, 2014.

[11] A. Yu, W. L. Lai, and J. Payor, "Efficient Integer Vector Homomorphic Encryption"; https://courses.csail.mit.edu/6.857/2015/files/yu-lai-payor.pdf, 2015.

[12] X. Du *et al.*, "A Routing-Driven Elliptic Curve Cryptography Based Key Management Scheme for Heterogeneous Sensor Networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, 2009, pp. 1223–29.

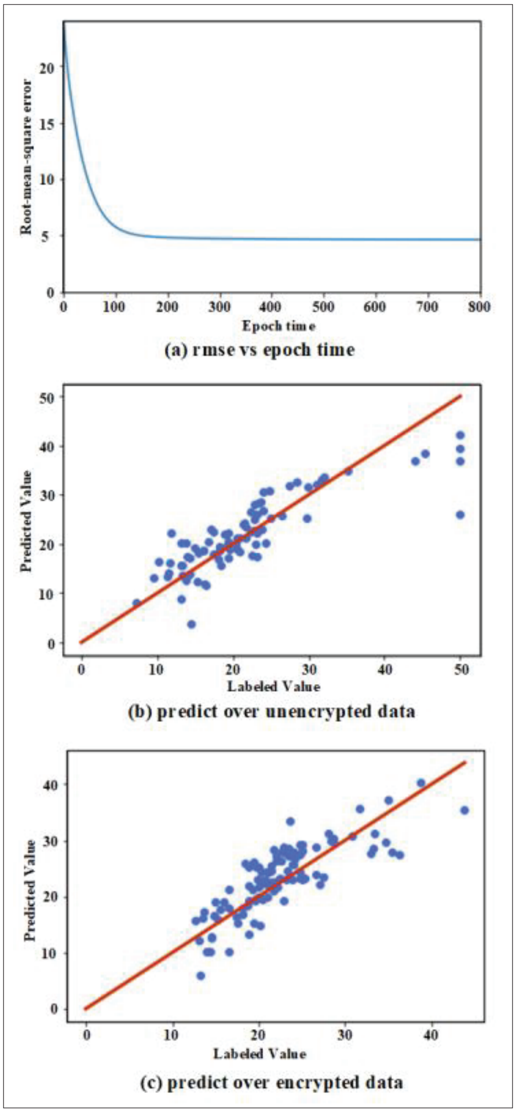
[13] X. Du *et al.*, "An Effective Key Management Scheme for Heterogeneous Sensor Networks," *Ad Hoc Networks*, vol. 5, no. 1, 2007, pp. 24–34.

[14] H. Yang *et al.*, "Efficient and Secure kNN Classification Over Encrypted Data Using Vector Homomorphic Encryption," *IEEE ICC*, MO, May. 2018, pp. 1–7.

[15] J. Zhang *et al.*, "Functional Mechanism: Regression Analysis Under Differential Privacy," *Proc. VLDB Endowment*, vol. 5, 2012, pp. 1364–75.

Biographies

Yong Yu (yuyong@snnu.edu.cn) is currently a professor at Shaanxi Normal University, Xi'an, China. He holds the prestigious one hundred talent Professorship of Shaanxi Province as well. He received his Ph.D. degree in cryptography from Xidian University in 2008. He has authored over 50 refereed journal and conference papers. His research interests are cryptography and its applications, especially public encryption, digital signature, and secure cloud computing. He is an Associate Editor of *Soft Computing*.



(a) rmse vs epoch time

(b) predict over unencrypted data

(c) predict over encrypted data

FIGURE 6. Implementation of linear regression over homomorphic encryption.

Huilin Li is currently a Master's student in the School of Computer Science, Shaanxi Normal University. Her research interest is smart contract security.

Ruonan Chen is currently a Master's student in the School of Computer Science, Shaanxi Normal University. Her research interest is cloud security.

Yanqi Zhao is currently a Master's candidate in the School of Computer Science, Shaanxi Normal University. His research interests are digital signatures and blockchain.

Haomiao Yang is currently an associate professor at the University of Electronic Science and Technology of China, Chengdu. His research interest is public key cryptography.

Xiaojiang Du [SM] (dxj@ieee.org) is currently a professor at Temple University, Philadelphia, Pennsylvania. He has published over 200 papers and has been awarded more than $5 million in research grants. His research interests are security, systems, and computer networks.