International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 12 | December - 2023 SJIF Rating: 8.176 ISSN: 2582-3930

Homomorphic Encryption Methods Review

Shreya verma[1]
Mtech Scholar[1]
Dept of Computer Science, RSR Rungta College of Engineering and Technology kohka -Kurud road bhilai

Mohd. Sajid Ansari[2]
Assistant professor [2]
Dept of Computer Science, RSR Rungta College of Engineering and Technology kohka -Kurud road bhilai

***Abstract*— Today, cloud technology continues to evolve. Using cloud services allows you to get financial benefit so Still, many companies and users are in no hurry to transfer their infrastructure to the cloud due to incompletely resolved problems related to the security of data storage and processing. In the case of storing and processing open data, a cloud provider gets access to the data, and an attacker can also gain by hacking an account. In the case of encrypted data transmission to the cloud, confidentiality will be preserved only in data storage tasks, since data processing will require a decryption task. The use of homomorphic ciphers allows the processing of encrypted data without violating their privacy. Homomorphic encryption is actively beginning to be used in machine learning tasks to transfer and ensure the confidentiality of resource-intensive operations for training a neural network in the cloud. The article offers a review and comparison of existing methods of homomorphic encryption for machine learning tasks.**

Keywords—homomorphic encryption; neural networks; modular arithmetic; distributed data storage; machine learning

I. INTRODUCTION

Cloud storage and data processing systems are widely used in the design of the IT infrastructure of enterprises and are becoming more widespread in society. Cloud technology is convenient for organizing the interaction of users and information service providers, as well as automatic data collection, processing, storage, and dissemination systems. However, an important task at the development stage of any cloud data processing system is to ensure confidentiality. The main problem, in this case, is the inability to guarantee customer confidentiality of data, since any cloud service provider can be compromised as a result of hacking or technical failure.

The purpose of encryption is to ensure the confidentiality of data processes. As a result, new functions were obtained that allow delegating computations for an unreliable computer. For this goal, we want to provide an untrusted cloud provider with only an encrypted version of the data processing. The cloud provider will perform calculations on this encrypted data, therefore, not knowing anything about its real value. It will send back the result of data processing the encrypted, and we will decrypt it.

For this reason, the encryption scheme should represent a specific structure. Consider the main approaches to constructing encrypted data processing schemes based on homomorphic encryption, see Section II. Section III presents the main unresolved issues and future work.

II. METHODS REVIEW

The cloud is an excellent platform for creating or placing pre-trained models because it offers cheap data storage, almost zero deployment cost, and high computing services [1]. However, it has some disadvantages that entail problems that need to be addressed. One of the main issues of cloud technology is the issue of data privacy. When using DLaaS, the user uploads his data to the cloud, which, in turn, evaluates the model according to the input data and sends the results back to the user. At each stage of this process, attackers have many opportunities for data theft.

As an example, suppose there is a financial service provider who claims to own a model that can be used to predict the market value of a particular class of companies with a high degree of accuracy. A service provider, ideally, would like to monetize its service by placing it in the cloud and providing paid forecasts to its customers. But a situation may occur in which a potential client may not want to use the service and share their confidential data with the cloud, despite the potential profit. An ideal solution to this problem is to protect both the model and personal data. This solution should require minimal participation of the client in the calculations, except for entering data and receiving the result of work. Also, the solution should be effective and have high performance.

There are at least three possible methods to tackle the problem above

1. Trusted Computing Base (TCB) [2], the use of hardware primitives for performing calculations in an isolated environment, for example, INTEL Software Guard Extensions (SGX) or AMD Secure Encrypted Virtualization (SEV);

2. Multi-Party Computation (MPC), which are algorithmic solutions that use cryptography to jointly evaluate a particular function between several parties without revealing the personal input of any party;

3. Fully homomorphic encryption (FHE), allows you to process data without decrypting it.

TCB solutions are highly efficient and suitable for the cloud computing paradigm. However, today, there are attacks on, TCB which some programs allow for obtaining information [3, 4, 5]. MPC solutions require a significant amount of communication between the parties. On the other hand, FHE- based solutions are similar to MPC solutions except that they are not interactive., i.e., do not require constant communication between the parties. However, they are computationally expensive. Providing computer resources is a more straightforward task than providing a secure connection, so FHE is better suited to solving the problem than MPC.

Machine learning based on FHE while maintaining

confidentiality was previously reviewed by Graepel et al. [6], and Aslett et al. [7]. Following them, Dowlin et al. [8] proposed CryptoNets, the first neural network based on encrypted data, providing a deep learning method in which the output phase allows for maintaining confidentiality. After the publication of thiswork, works appeared [9–13] in which many cryptographic methods were used to achieve similar goals. The main disadvantage of these FHE-based solutions is the high computational cost. For example, CryptoNets took 570 seconds to evaluate an FHE-friendly model on encrypted samples from the MNIST dataset at a security level (80 bits). Also, this scheme requires large open text ($t \approx 2^{80}$), which should have been decomposed using the Chinese remainder theorem (CRT) into two smaller (2^{40}) modules. Besides, the scheme (YASHE) that they used is not recommended because of the attack on it described in [14].

Krizhevsky et al. [15] proposed the AlexNet scheme and showed the advantages of convolutional neural networks (CNN) implemented on GPU in image classification problems. For the practical implementation of homomorphic convolutional neural networks (HCNN), many problems still have to be solved. This works as follows. Encryption masks the input, called plaintext, a random error taken from some distribution, which leads to encrypted text that does not reveal anything about what it encrypts. For decryption, a secretkey is used to filter noise and extract plain text. During the calculations, the noise in the ciphertexts grows in a controlled manner. Still, at some point, it increases to such an extent that no further calculations can be performed without errors during decryption.

Research in the field of deep learning, which preserves confidentiality, can be divided into two parts: the former use homomorphic encryption, and the latter combines it with secure multi-party computing (MPC) methods. The systems CryptoNets, Dowlin, et al. [8], FHE-Dinn, Bourse, et al. [16], and E2DM, Jiang et al. [11] use only fully homomorphic encryption to solve this problem. Dowlin et al. [8] were the first to propose using FHE as the basis for designing neural networks for deep learning while maintaining confidentiality that can work on encrypted data. They proposed using polynomial approximations of the most common ReLU activation function and using union layers only during the training phase to reduce the depth of their neural network chain. However, they used the YASHE scheme from Bos et al. [17], which is no longer safe due to the attack proposed by Albrecht et al. [14]. Also, they need a sizeable clear text of more significant than 80 bits to accommodate the result of their neural network. This makes it very difficult to scale to deeper networks, as the intermediate layers in these networks will quickly reach several hundred bits.

Bourse et al. [16] proposed a new type of neural network called Discretized Neural Networks (Dinn) to output encrypted data. The weights and input data of traditional CNNs are divided into elements lying at −1,1, and the fast boot TFHE scheme proposed by Chilotti et al. [14] was used to double the function of neuron activation. Each neuron calculates the weighted sum of its inputs, and the activation function is a sign function of sign (z), which displays the sign of the input z. Although this method can be applied to arbitrary deep networks, it does not have a sufficient accuracy of 96.35% on a lower-performance MNIST dataset.

Jiang et al. [11] proposed a new matrix multiplication method with FHE and evaluated a neural network on a dataset. They also considered the possibility of packing the entire image into one ciphertext compared to the approach of Dowlin et al. [8], which put only one pixel in the ciphertext but evaluate large sets of images at a time. They achieved good performance, rating 64 images in just under 29 seconds, but with worse performance.

Some of the main limitations of simple FHE-based solutions are the need to approximate non-polynomial activation functions and long computation times. To solve theseproblems, Liu et al. [13] proposed the MiniONN method. Theytake commonly used protocols in deep learning and turn them into forgotten protocols. Using MPC, they can evaluate neural networks without changing the learning phase, while maintaining accuracy, since approximation is not required for activation functions. However, MPC has its drawbacks. In this parameter, for each calculation, a connection is required between the data owner and the model owner, which leads to the use of networks with high bandwidth.

Juvekar et al. [12] developed GAZELLE, in which, instead of using FHE, they use alternating approaches of an additively homomorphic encryption scheme for levels of convolution type and distorted chains for levels of activation and association. Thus, communication complexity is reduced compared to MiniONN but is still significant.

One of the promising approaches for ensuring the confidentiality of data processed in the cloud is homomorphic encryption. Homomorphic encryption is meant a method of encrypting data that allows it to be processed in an encrypted form without decryption, which ensures confidentiality. For example, machine learning methods for processing encrypted confidential data: personal, medical, and commercial data have a high potential for practical application. At the same time, FHE schemes that support the addition and multiplication of encrypted numbers and allow to implementation of a wide range of algorithms for processing sensitive data deserve special attention.

Significant results in the development of the theory of completely homomorphic encryption schemes were achieved by Craig Gentry in 2009 [18]. Gentry proposed a method for constructing an FHE based on an arbitrary homomorphic cipher. However, the methods proposed by Gentry have two significant drawbacks that do not allow their use in practice: large redundancy and high computational complexity.

To eliminate these shortcomings, leveled homomorphic encryption schemes have been proposed that allow you to operate for adding encrypted numbers as many times as you like, and the number of operations to multiply encrypted numbers is a limited number of times. This approach allowed us to reduce the redundancy of encrypted data of homomorphic encryption schemes. However, the problem of effective implementation of the schemes is open.

The basis of modern fully homomorphic ciphers is the NP-difficult task of Learning With Errors (LWE) and its variant, which has important practical applications - learning with errors in the polynomial ring (RLWE). In LWE-based encryption schemes, a small error is added to the encrypted message, which is eliminated during the decryption process. Depending on the method of adding errors to the ciphertext, two important types of modern homomorphic encryption schemes based on RLWE can be distinguished: BGV [19] and

BFV [20]. These schemes are included in the developed standard of homomorphic encryption [21] and are the basis of many FHE schemes with various properties. For example, a fully homomorphic encryption scheme for approximate numbers by Cheon et al. [22] is based on the BGV approach, which is more efficient than peers and has a controlled increase in encryption errors.

However, the computational complexity and redundancy of encryption schemes make them inapplicable in practice at the moment. This is because, to achieve the required level of cryptographic scheme strength (more than 128 bits of semantic power), RLWE-based ciphers, according to the introduced standards of homomorphic encryption [21], require computations with large degrees polynomials (more than 4096) with significant coefficients (several hundreds of bits). General-purpose processors are ineffective for such tasks since the maximum level of parallelism is required. The creation of specialized accelerators (mathematical coprocessors) will solve the issue of the effectiveness of homomorphic encryption schemes and will expand the range of their applicability. One of the main objectives of the proposed project is to develop a computing strategy that applies to a data center equipped with similar hardware accelerators.

III. FURTHER WORK

The main task of such accelerators is to implement operations in a ring of large polynomials with significant coefficients. The key approaches for this are the use of the Number-Theoretic Algorithms (NTA) as a variant of the discrete Fourier transform in finite fields to accelerate the multiplication of polynomials and a system of residual classes to accelerate the work by performing arithmetic operations in parallel. The Residue Number System (RNS) is a non-positional number system in which numbers are represented as residues by dividing them into pre-selected, mutually simple modules. This representation allows you to simultaneously perform the operations of multiplication and addition of numbers to the residue ring without transferring data between bits (residues), speeding up calculations with multi-bit numbers. An important feature of RNS is the possibility of homomorphic data processing. Adaptation of homomorphic encryption algorithms for acceleration using RNS (BEHZ 2016 and HPS 2018 for BFV, HEAX for CKKS 2017) allowed us to create several fairly effective practical implementations, including both Microsoft SEAL software and hardware HEAX solutions. RNW is actively used in a variety of cryptographic algorithms, including homomorphic ciphers based on problems other than RLWE [23], and allows significant acceleration to be achieved.

However, many operations in the RNS cannot be implemented effectively. These include non-modular operations – operations that, in one form or another, require anestimate of the positional value of a number, which leads to inter-module interactions and cannot be implemented in parallel. Such operations, such as scaling, dividing, and comparing numbers in magnitude, are considered complex in RNS and require a separate study. In various cases, when used for homomorphic encryption, such operations can, on the one hand, reduce the performance of operations and, on the other hand, lead to the accumulation of additional errors in the ciphertext. At the moment, effective methods for calculating positional characteristics have been developed that apply to a wide class of algorithms and allow the development of universal devices based on RNS.

Further optimization of homomorphic encryption algorithms using RNS requires taking into account the features of the implementation of algorithms of this type on specific platforms (FPGA, ASIC). Currently, RNS is applied to already developed schemes that do not take into account the features of this number system. The combination of parameters of encryption schemes and parameters of RNS to achieve maximum performance is currently a poorly studied problem that affects the choice of specific approaches to the implementation of algorithms in RNS and the organization of homomorphic calculations. Within the framework of the project, it is planned to develop new methods for homomorphic encryption using RNS, combining, on the one hand, the possibility of efficient hardware implementation, and on the other, efficient algorithms for performing non-modular operations in RNS, as well as direct and inverse conversion to RNS.

To achieve this result, careful consideration of the compatibility parameters of homomorphic encryption schemes and algorithms in RNS is required. It is necessary to develop generalized models of homomorphic ciphers in RNWE-based RNWEs, which allow for a detailed analysis of the developed algorithms and take into account all the features of devices operating in RNWEs. The maximum effect can be achieved when developing a specialized homomorphic encryption scheme adapted for the use of RNS. It is necessary to take into account the trends in the development of homomorphic ciphers for approximate calculations that have great potential for practical application.

REFERENCES

[1]. J. Yashpalsinh, M. Kirit, Cloud computing-concepts, architecture, and challenges. In Computing, Electronics and Electrical Technologies (ICCEET), International Conference on. IEEE, 2012, pp. 877–880

[2]. John M Rushby, Design, and verification of secure systems, ACM, 1981,vol. 15, No. 5, pp. 12-21

[3]. G. Chen, S. Chen, Y. Xiao, Y. Zhang, Z. Lin, T.H. Lai, SgxPectre Attacks: Leaking Enclave Secrets via Speculative Execution, 2018

[4]. P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, Y. Yarom, Spectre attacks: Exploiting speculative execution, 2018

[5]. M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn,S. Mangard, P. Kocher, D. Genkin, Y. Yarom, M. Hamburg, Meltdown: Reading Kernel Memory from User Space. In 27th USENIX Security Symposium (USENIX Security 18), 2018

[6]. T. Graepel, K. Lauter, M. Naehrig, ML Confidential: Machine Learning on Encrypted Data, 2013, pp. 1–21

[7]. Louis J. M. Aslett, Pedro M. Esperanca, C. C. Holmes, Encrypted statistical machine learning: new privacy preserving methods. ArXiv e- prints (2015). arXiv:1508.06845, 2015

[8]. N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig,,J. Wernsing, CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy, Technical Report, 2016

[9]. F. Bourse, M. Minelli, M. Minihold, P. Paillier, Fast Homomorphic Evaluation of Deep Discretized Neural Networks, 2018, pp. 483–512

[10]. E. Hesamifard, H. Takabi, M. Ghasemi, R. N. Wright, Privacy-

preserving Machine Learning as a Service. PoPETs 2018, 2018, pp. 123–142.

[11]. X. Jiang, M. Kim, K. Lauter, Y. Song, Secure Outsourced Matrix Computation and Application to Neural Networks, In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18)

[12]. C. Juvekar, V. Vaikuntanathan, A Chandrakasan, GAZELLE: A Low Latency Framework for Secure Neural Network Inference, In 27th USENIX Security Symposium (USENIX Security 18). USENIX Association, . 2018, pp. 1651–1669.

[13]. J. Liu, M. Juuti, Y. Lu, N. Asokan, Oblivious Neural Network Predictions via MiniONN Transformations, 2017, pp. 619–631.

[14]. M. Albrecht, S. Bai, L. Ducas, A Subfield Lattice Attack on Overstretched NTRU Assumptions - Cryptanalysis of Some FHE and Graded Encoding Schemes, 2016, pp. 153–178.

[15]. Krizhevsky, I. Sutskever, G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems – vol. 1, 2012, pp. 1097–1105.

[16]. F. Bourse, M. Minelli, M. Minihold, P. Paillier, Fast Homomorphic Evaluation of Deep Discretized Neural Networks, 2018, pp. 483–512.

[17]. J. Bos, K Lauter, J. Loftus, M. Naehrig, Improved Security for a Ring-Based Fully Homomorphic Encryption Scheme, 2013, 45–64.

[18]. C. Gentry, Fully homomorphic encryption using ideal lattices, in Proc. 41st Annu. ACM Symp. Theory Comput, 2009, 169–178.

[19]. Z. Brakerski, C. Gentry, V. Vaikuntanathan, Fully homomorphic encryption without bootstrapping, In ITCS'12 Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, 2012, pp. 309-325.

[20]. J. Fan, F. Vercauteren, Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. http://eprint.iacr.org/2012/144.pdf

[21]. M. Albrecht, M. Chase, H. C., J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, S. Lokam, D. Micciancio, D. Moody, T. Morrison, A. Sahai, V. Vaikuntanathan, Security of homomorphic encryption, HomomorphicEncryption. org, Redmond WA, Technical Report, 2017.

[22]. J.H. Cheon, A. Kim, M. Kim, Y. Song, Homomorphic encryption for arithmetic of approximate numbers. In International Conference on the Theory and Application of Cryptology and Information Security, 2017, pp. 409-437.

[23]. P. Alves, D. Aranha, Efficient GPGPU implementation of the leveled fully homomorphic encryption scheme YASHE.