

A Review of Homomorphic Encryption Applications in Deep Learning

Junhao Wu *

Shanghai University of Engineering Science, Shanghai, 201620, China

* Corresponding author Email: 028123401@sues.edu.cn

Abstract: In the era of artificial intelligence, deep learning has achieved success in fields such as image recognition and natural language processing, but the training process, which relies on sensitive data, raises privacy risks. Traditional privacy protection techniques, such as differential privacy and secure multi-party computation, either result in accuracy loss or high computational overhead. Homomorphic encryption (HE) can perform operations directly on ciphertext without decrypting the data, providing a new approach for deep learning to protect data privacy. Microsoft's CryptoNets system in 2016 first verified the feasibility of performing neural network inference on encrypted data, leading to a research boom combining Homomorphic encryption with deep learning. This paper systematically reviews the latest progress in combining homomorphic encryption with deep learning, introducing typical works from three aspects: encrypted inference, encrypted training, and hybrid methods, analyzing performance bottlenecks, and looking forward to future development directions.

Keywords: Deep Learning; Homomorphic Encryption; Privacy Preservation.

1. Introduction

Introduction Deep learning driven by big data requires massive amounts of real-world data, which often contain sensitive information such as personal identity, medical records, and financial data. Directly processing such data in plaintext on cloud platforms may lead to privacy leakage and loss of user trust. To address these concerns, cryptographic techniques have been introduced into machine learning systems. Homomorphic encryption allows arithmetic operations to be performed directly on ciphertexts, and the decrypted result matches the computation performed on plaintext, realizing the concept of “data being usable but not visible” [1].

After Gentry's pioneering work on fully homomorphic encryption based on ideal lattices [1], more practical lattice-based schemes were proposed to improve efficiency. Fan and Vercauteren introduced a somewhat practical fully homomorphic encryption scheme that significantly reduced computational complexity [2]. Subsequently, Cheon et al. proposed the CKKS scheme, which supports approximate arithmetic over real numbers and is particularly suitable for machine learning applications [3]. In addition, the TFHE scheme enables fast Boolean gate evaluation and efficient bootstrapping, expanding the range of supported applications [4]. Further theoretical improvements based on the Learning With Errors (LWE) assumption strengthened the security foundations of fully homomorphic encryption [5].

The feasibility of applying homomorphic encryption to deep learning was first demonstrated by CryptoNets, which showed that convolutional neural networks could perform inference directly on encrypted data with acceptable accuracy [6]. Since then, extensive research has focused on improving efficiency, reducing latency, and extending HE-based techniques to more complex models and application scenarios.

2. Homomorphic Encryption Technology Foundation

2.1. Homomorphic Encryption Technology

Homomorphic Encryption Techniques Homomorphic encryption is a cryptographic technique that supports computation on ciphertexts while preserving correctness after decryption. According to the supported operations, HE schemes can be classified into partially homomorphic encryption, somewhat homomorphic encryption, and fully homomorphic encryption. Early schemes such as Paillier support only additive homomorphism, while modern lattice-based schemes support both addition and multiplication [2]. Fully homomorphic encryption schemes allow arbitrary-depth computations but suffer from noise accumulation, which necessitates costly bootstrapping operations to refresh ciphertexts [1]. The BFV and BGV schemes support exact integer arithmetic but incur significant ciphertext expansion and computational overhead [2]. In contrast, the CKKS scheme supports approximate arithmetic over real numbers, making it widely adopted in privacy-preserving machine learning [3]. TFHE focuses on fast Boolean operations and provides efficient bootstrapping, which is advantageous for logic-based computations [4].

2.2. Performance and Characteristics Comparison of Typical Algorithms

The Paillier, BFV, BGV, and CKKS algorithms introduced above each have their own characteristics and are suitable for different application needs. The table below compares and summarizes them in terms of accuracy, computational efficiency, supported operation types, fully homomorphic capability, and applicable scenarios:

3. Privacy Protection Needs in Deep Learning

Deep learning's reliance on massive amounts of data poses

serious privacy risks, primarily manifested in direct and indirect leaks. Direct privacy leaks refer to the risk of improper use or leakage of centrally collected user data, a problem that frequently occurs in reality. Distributed training methods such as federated learning alleviate this issue to some extent by keeping data locally, but cannot completely prevent privacy leaks. On the other hand, attackers can also

indirectly attack training data through model outputs, such as model inversion (reconstructing training samples based on model outputs) and membership inference (determining whether an input belongs to the model's training set). Even in federated learning, malicious actors can use this to infer other users' private data.

Table 1. Comparison of various options

Solution/Dimension	Paillier	BFV / BGV	CKKS
Plaintext type and precision	Integers, precise results	Integers, modulo q arithmetic, exact.	Floating-point number(fixed-point approximation),the result is approximately
Supported homomorphic operations	Additive homomorphism (ciphertext multiplication = plaintext addition); does not support inter-ciphertext multiplication.	Addition and multiplication are homomorphic, supporting arbitrary mixed operations (SHE / FHE).	Addition and multiplication are homomorphic, supporting floating-point operations.
Are they all homomorphic	No, only partially homomorphic.	Yes (achieving complete homomorphism through bootstrapping)	Approximately homomorphic can be extended to completely homomorphic.
Computational efficiency and overhead	Most efficient, ciphertext is approximately twice as fast as plaintext.	It's relatively slow; the ciphertext expands by tens of times, and the computation time is in the millisecond to second range.	Slightly slower than BFV (~15%), with significant ciphertext expansion and bootstrapping up to sub-second levels.
Typical application scenarios	Security statistics, electronic voting, aggregation summation	Integer-type fully homomorphic computation: cryptographic databases, financial/medical analytics	Floating-point homomorphic computation: machine learning inference, statistical analysis, signal processing

Privacy issues have received widespread attention, and various countries have enacted laws to strictly regulate the use of personal data (such as China's Cybersecurity Law and the EU's GDPR), requiring that personal information not be collected or provided without authorization. This makes implementing privacy protection in the training and deployment of deep learning models a mandatory requirement. Besides complying with regulations, it is also necessary to technically ensure that sensitive data is not leaked at any stage of deep learning. Homomorphic encryption, due to its "computable but invisible" characteristics, is considered one of the important technologies for achieving secure artificial intelligence[5].

4. Typical Systems and Applications

The combination of homomorphic encryption and deep learning is mainly reflected in three scenarios: encrypted inference, encrypted training, and hybrid methods. The basic principles and representative works of each type are introduced below.

4.1. Encrypted Inference

Encrypted inference applies homomorphic encryption to the inference stage of deep learning models, ensuring that inputs and intermediate results remain encrypted. CryptoNets pioneered this approach by replacing nonlinear activation functions with polynomial approximations, achieving high accuracy on the MNIST dataset [6]. However, its inference latency remained high. To improve efficiency, hybrid frameworks have been proposed. Gazelle combines homomorphic encryption for linear layers with secure two-party computation for nonlinear operations, reducing inference latency to milliseconds without sacrificing accuracy [7]. Compiler-based systems such as nGraph-HE further optimize encrypted inference by automatically selecting

parameters and scheduling operations efficiently [8]. Automated and manual optimizations for homomorphic neural network evaluation have also been extensively studied [9].

4.2. Encrypted Training

Encrypted training aims to perform the entire training process in the ciphertext domain. Due to the high computational cost of gradient descent under HE, early work focused on simple models such as logistic regression [10]. Approaches based on discretized neural networks improved efficiency but remained limited in model complexity [11]. Low-latency inference techniques further reduced overhead but did not fully address training scalability [12].

Recent advances have demonstrated partial breakthroughs in encrypted training. The ReBoot framework enables multi-layer neural network training under fully homomorphic encryption by leveraging CKKS and efficient bootstrapping techniques. This work shows that encrypted training with accuracy comparable to plaintext training is feasible for small neural networks.

4.3. Hybrid Methods

Hybrid approaches combine HE with other technologies to balance security and efficiency. A typical approach is to divide tasks according to the type of operation: HE performs large-scale linear computations (such as matrix multiplication and convolution), while a secure two-party computation (MPC) protocol precisely processes nonlinear activation functions, thus preserving model accuracy without data leakage. Taking the Gazelle framework as an example, convolution and fully connected layers are computed in parallel by HE, while activation layers such as ReLU are securely evaluated by Yao obfuscated circuits (Boolean circuit protocol), reducing inference latency from seconds to milliseconds without modifying the mode. Hybrid schemes

fully leverage the strengths of each technology, significantly reducing the latency of secure inference without loss of accuracy. However, their disadvantages lie in the increased trust assumptions and system complexity. It is necessary to assume that multiple participants or hardware platforms are reliable and trustworthy, otherwise data leakage may occur. Key conversion and data exchange between different modules increase implementation complexity and potential vulnerabilities. Most protocols require the client to participate in the computation of each layer, thus exposing the model structure to the client, and the client bears an additional computational burden, which is not conducive to deployment on terminals with limited computing power. Nevertheless, hybrid approaches are seen as a promising way to balance security and efficiency in practice, offering possibilities for the practical application of privacy-preserving deep learning.

5. Performance and Bottleneck Analysis of Typical Systems

5.1. The Main Performance Bottleneck of Homomorphic Encryption Deep Learning

5.1.1. Computational Complexity and Bootstrapping Overhead

The biggest problem with HE + DL is its high computational cost. Operations that take microseconds in the plaintext domain can take seconds in the ciphertext domain due to the extremely high overhead of polynomial rings and large-modulus number theory transformations[13]. Ciphertext multiplication is the most time-consuming operation, adding noise and consuming "noise margin" with each operation. When the depth approaches the upper limit, bootstrapping re-encryption is required, taking hundreds of milliseconds to several seconds, and is considered the slowest operation in FHE. Even with a "leveled" scheme that reserves parameters, the ciphertext size still expands and computation slows down.

5.1.2. Noise Accumulation and Control

Noise both ensures security and limits computational depth. Each operation introduces additional noise, and exceeding this limit prevents decryption. Common approaches include increasing encryption parameters or resetting the noise during bootstrapping, but the latter is costly. Leveled FHE typically only supports a few dozen levels of multiplication; while full FHE supports arbitrary depth, a single bootstrapping operation still takes 0.1 seconds to several seconds. CKKS's fast bootstrapping can increase the multiplication depth from 20 to over 40 levels, but it remains a major bottleneck.

5.1.3. Communication Overhead and Ciphertext Size

Although FHE is non-interactive, the ciphertext size far exceeds the plaintext size: a plaintext vector of a few KB can expand to hundreds of KB to several MB after encryption. In cross-institutional scenarios, uploading large amounts of ciphertext can cause bandwidth bottlenecks. The size of auxiliary public keys such as evaluation keys can reach tens of MB, constituting a one-time communication overhead, which, although reusable, cannot be ignored. Compared to 2PC, which requires multiple rounds of interaction, FHE can amortize communication overhead in batch inference (such as video stream analysis)[14].

5.1.4. Parallelism and Memory Bandwidth

RLWE-type schemes (BFV/BGV/CKKS) support SIMD batch processing, which can parallelize N numbers in a single

ciphertext. For example, CryptoNets batch processes 4096 MNIST images with a single image latency of 250 seconds, but the throughput is 0.05 seconds/image. Some operations (rotation, relinearization) still need to be serialized, which limits parallelism. Although TFHE can quickly evaluate Boolean gates, it does not support SIMD and is less efficient for batch tasks[4].

5.1.5. Hardware Acceleration and Computing Power Limitations

CPUs are limited by the difficulty in fully utilizing memory bandwidth, and GPU/FPGA acceleration is often used. GPUs, by porting CKKS, achieve approximately 267x speedup on ResNet-20 (latency reduced from tens of seconds to 8.5 seconds), and by using a low-order polynomial approximation of ReLU, the speedup can be reduced to 1.4 seconds. FPGAs can further improve performance by approximately 370x, but are limited by on-chip resources and key size. Overall, while hardware acceleration significantly reduces runtime, it is still much slower than plaintext computation and introduces cost and flexibility issues[15].

5.2. Performance Comparison of Typical HE + DL Solutions

- CryptoNets (2016, Microsoft): The first cryptographic inference system based on the BFV scheme and squared activation $f(x)=x^2$, achieving 99% accuracy on MNIST. It infers 59,000 images per hour (4096 images in batch SIMD parallel processing), with a single image latency of approximately 250 seconds.
- Gazelle (2018, MIT): Employs a hybrid HE + 2PC approach, with linear layers performing HE computation and nonlinear layers performing 2PC execution, resulting in a three-order-of-magnitude speedup. Online inference reaches millisecond levels with accuracy comparable to plaintext.
- CHET (2019, Microsoft): An automatically optimized framework supporting BFV/CKKS. It implements 40-layer SqueezeNet inference on CIFAR-10 without bootstrapping, taking several minutes with near-plaintext accuracy.
- HE-Transformer (2019, Intel): The HE backend for the nGraph compiler, based on CKKS, supporting real weights and rescaling. Small networks can perform inference in seconds, up to 10 times faster than CryptoNets; large networks such as ResNet-50 still take hours.

5.3. Trends and Conclusion

Currently, HE + DL has achieved near-plaintext accuracy (accuracy loss $\leq 1\%$), but latency remains several orders of magnitude higher. Supported network depths are mostly within a few dozen layers; increasing the input dimensionality significantly amplifies computational and memory costs. Research is progressing in three areas: algorithms (fast bootstrapping), software (automatic parameter optimization), and hardware (GPU/FPGA acceleration). Recent inference latency has decreased from minutes to seconds or even sub-seconds. Future breakthroughs in computational complexity, system architecture, and dedicated hardware, achieving a balance between accuracy, performance, and security, are crucial for the practical application of HE deep learning.

6. Future Research Directions

6.1. Future Research Directions

Model structure optimization: Use shallower networks as much as possible, replace complex nonlinear functions with functions that are easily approximated by polynomials, and reduce the depth and complexity of homomorphic operations while maintaining accuracy Bootstrapping

Efficiency: Bootstrapping is time-consuming, and adaptive refresh strategies and more efficient algorithms should be studied to reduce refresh frequency and overhead. The existing FHE scheme has reduced the latency of a single bootstrapping to sub-second level, and further optimization is expected to make frequent refresh possible[5].

Multi-party collaboration and multi-technology integration: In order to realize encrypted collaborative reasoning of multi-party data, it is necessary to study multi-key FHE and key exchange technology; at the same time, combine MPC, zero-knowledge proof, TEE and other technologies to build a hybrid privacy computing framework to balance security and efficiency[16,17].

6.2. Application Prospects of Homomorphic Encryption Deep Learning

HE+DL has broad application prospects in sensitive data fields such as healthcare, finance, and government affair[18]. For example, in the healthcare field, hospitals can encrypt patient data and upload it to cloud-based AI models for analysis, obtaining diagnostic results while ensuring that patient privacy is not leaked; in the financial industry, multiple institutions can encrypt their customer data and jointly assess credit risk, and finally decrypt the data themselves to obtain the results, without exposing any plaintext data throughout the process; in government regulation, various departments can jointly use AI models to discover abnormal transactions and risky entities without sharing plaintext data[19].

7. Conclusion

Currently, encrypted models have achieved near-perfect accuracy compared to plaintext models for some tasks, and inference performance has been improved by hundreds of times through batch processing and hardware acceleration. However, overall, the scale and speed of models supported by encryption algorithms (HE) are still far inferior to those of plaintext models, and efficiency bottlenecks remain prominent for complex deep learning tasks. Future efforts require continuous innovation in cryptographic algorithms, software frameworks, and hardware design to gradually narrow the performance gap between HE and plaintext computation. Only by continuously improving computational efficiency while ensuring data privacy and model accuracy can practical privacy-preserving deep learning be ultimately realized, laying the foundation for AI applications where "data is usable but not visible."

References

- [1] Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, 9, 169–178.
- [2] Fan, J., & Vercauteren, F. (2012). Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012, Article 144.
- [3] Cheon, J.H., Kim, A., Kim, M., & Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. *Advances in Cryptology – ASIACRYPT 2017*, 10624, 409–437.
- [4] Chillotti, I., Gama, N., Georgieva, M., & Izabachène, M. (2020). TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1), 34–91.
- [5] Brakerski, Z., & Vaikuntanathan, V. (2021). Fully homomorphic encryption from learning with errors. *SIAM Journal on Computing*, 50(2), 628–657.
- [6] Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., & Wernsing, J. (2016). CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy. *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 48, 201–210.
- [7] Juvekar, C., Vaikuntanathan, V., & Chandrakasan, A. (2018). Gazelle: A low latency framework for secure neural network inference. *Proceedings of the 27th USENIX Security Symposium*, 2018, 1651–1669.
- [8] Boemer, F., Lao, Y., Wierzynski, C., & Naehrig, M. (2019). nGraph-HE: A graph compiler for deep learning on homomorphically encrypted data. *Proceedings on Privacy Enhancing Technologies*, 2019(1), 3–24.
- [9] Dowlin, N., Gilad-Bachrach, R., Laine, K., Lauter, K., Naehrig, M., & Wernsing, J. (2019). Manual and automated homomorphic evaluation of neural networks. *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, 135–152.
- [10] Kim, M., Song, Y., Wang, S., Xia, Y., & Jiang, X. (2021). Secure logistic regression based on homomorphic encryption: Design and evaluation. *IEEE Access*, 9, 109021–109035.
- [11] Bourse, F., Minelli, M., Minihold, M., & Paillier, P. (2018). Fast homomorphic evaluation of deep discretized neural networks. *Advances in Cryptology – CRYPTO 2018*, 10993, 483–512.
- [12] Brutzkus, A., & Gilad-Bachrach, R. (2020). Low latency privacy-preserving inference. *Proceedings of the 37th International Conference on Machine Learning Workshops*, 2020, 812–821.
- [13] He, Z., Zhang, Y., & Lee, J. (2021). Efficient deep learning inference on encrypted data. *IEEE Transactions on Parallel and Distributed Systems*, 32(6), 1325–1338.
- [14] Lou, Q., Jiang, Y., & Zhou, X. (2022). Efficient CNN inference over homomorphically encrypted data with optimized bootstrapping. *Neurocomputing*, 481, 112–125.
- [15] Li, J., & Wang, H. (2023). Hardware acceleration of homomorphic encryption for privacy-preserving artificial intelligence. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 70(4), 1502–1514.
- [16] Wei, X., Zhang, Q., & Xu, L. (2022). Hybrid privacy-preserving deep learning with homomorphic encryption and secure multi-party computation. *Computers & Security*, 114, 102602.
- [17] Chen, X., Wang, Y., & Liu, Z. (2023). ReBoot: Efficient fully homomorphic encryption for neural network training. *IEEE Transactions on Information Forensics and Security*, 18, 4210–4224.
- [18] Wu, T., & Zhang, C. (2024). Privacy-preserving federated learning with homomorphic encryption and differential privacy. *IEEE Internet of Things Journal*, 11(3), 4125–4138.
- [19] Xu, Y., & Liu, H. (2024). Homomorphic encryption-based secure medical image classification. *BMC Medical Informatics and Decision Making*, 24(1), 56.
- [20] Zhao, S., & Li, K. (2025). Towards practical privacy-preserving deep learning: A survey of homomorphic encryption and secure multi-party computation integration. *ACM Computing Surveys*, 57(1), 1–38.