

Hardware-Accelerated Privacy-Preserving Neural Network Inference via Efficient Activation Design

Willem Sitters

Department of Computer Science, University of Houston, Houston, TX, USA.
contactwillem@uh.edu

Hiral Mukhopadhyay

Department of Computer Science, Colorado State University, Fort Collins, CO, USA.
hiral1983@colostate.edu

Ingetraud Wesack

Department of Electrical Engineering and Computer Science, University of Missouri,
Columbia, MO, USA.
ingetraud495@missouri.edu

Abstract

Privacy-preserving neural network inference has shifted from a narrow cryptographic curiosity to a systems problem whose viability depends on hardware, compilers, network conditions, governance, and model architecture. Secure multiparty computation, homomorphic encryption, and trusted execution environments each impose distinct computational and communication bottlenecks, but activation functions consistently emerge as a central point of tension. Nonlinear activations such as the rectified linear unit are essential to modern predictive accuracy, yet they are expensive under encryption and secret sharing because they require comparisons, branching, or high-degree approximation. This paper examines hardware-accelerated privacy-preserving inference through the lens of efficient activation design. It argues that activation choices are not merely model-level hyperparameters but cross-layer infrastructure decisions that shape accelerator utilization, memory movement, communication rounds, energy consumption, auditability, and fairness. The analysis considers graphics processing units, field-programmable gate arrays, application-specific integrated circuits, and heterogeneous cloud-edge deployments. It then explores how ReLU-reduced architectures, quantized activations, lookup tables, polynomial approximations, and mixed-protocol execution alter the trade-offs among latency, throughput, accuracy, privacy guarantees, and sustainability. The paper also discusses governance implications, including certification, liability, export controls, and equitable access. The central claim is that efficient activation design can unlock practical private inference only when it is co-designed with hardware acceleration, cryptographic protocols, and policy frameworks rather than treated as an isolated optimization.

Keywords

privacy-preserving inference; secure multiparty computation; homomorphic encryption; hardware acceleration; activation functions; neural network architecture; trustworthy systems; energy efficiency.

1. Introduction

Machine learning inference is increasingly embedded in domains where inputs are sensitive, including healthcare diagnostics, financial risk assessment, identity verification, public safety, and workplace monitoring. In these settings, sending raw data to a remote model provider creates legal, ethical, and operational risks. Privacy-preserving neural network inference seeks to compute predictions without revealing the input to the model owner or the model to the data owner. Early work on secure computation established the theoretical feasibility of such protocols through garbled circuits and general secure multiparty computation [1], [2]. Later systems demonstrated that neural networks could be evaluated under encryption or secret sharing with acceptable accuracy [3], [4]. Yet the transition from feasibility to deployable infrastructure remains incomplete. The dominant obstacles are not only cryptographic in the narrow sense but also architectural, economic, and institutional.

A recurring bottleneck is the activation function. Modern neural networks rely on nonlinearities to learn complex representations. The rectified linear unit and its variants are favored because they are simple in plaintext computation and support sparse gradients. Under privacy-preserving protocols, however, the same operation becomes costly. In secure multiparty computation, computing a rectified linear unit often requires a private comparison, which translates into communication rounds, oblivious transfer, or garbled circuit evaluation. In homomorphic encryption, nonlinear functions must be approximated by polynomials or evaluated through lookup tables, both of which increase multiplicative depth and noise growth. Consequently, activation design sits at the intersection of model quality, cryptographic cost, and hardware efficiency. This paper argues that efficient activation design should be treated as a first-class systems concern for hardware-accelerated private inference.

The hardware landscape further complicates the picture. Graphics processing units offer massive parallelism but can be underutilized when secure computation is dominated by communication or cryptographic preprocessing. Field-programmable gate arrays provide reconfigurable data paths and low-latency custom operators, yet they require careful placement and routing for large models. Application-specific integrated circuits can achieve exceptional energy efficiency but lack flexibility as models and cryptographic standards evolve. Trusted execution environments offer a simpler programming model but introduce side-channel and attestation concerns. Across these platforms, activation functions determine whether an accelerator is compute-bound, memory-bound, or communication-bound. A private inference stack that ignores activation design risks optimizing the wrong layer of the system.

The paper also situates private inference within broader socio-technical infrastructures. Privacy is not a binary property; it is a bundle of guarantees that must be governed, audited, and communicated. Hardware acceleration can improve accessibility by lowering cost, but it can also concentrate power among providers capable of deploying specialized silicon. Efficient activation design may reduce energy consumption, yet cryptographic overhead can still produce substantial carbon footprints. Fairness concerns arise when approximations, quantization, or reduced-precision arithmetic degrade accuracy unevenly across subpopulations. Policy questions concern certification, liability, export controls, and the extent to which private inference should be treated as critical infrastructure. The analysis therefore proceeds from cryptographic foundations to hardware architectures, then to deployment, governance, robustness, fairness, and sustainability.

2. Foundations of Privacy-Preserving Neural Network Inference

Privacy-preserving inference protocols generally fall into three families. The first is secure multiparty computation, in which multiple parties jointly evaluate a function while revealing only the output. Garbled circuits provide one foundation, allowing one party to encrypt a circuit and another to evaluate it without learning intermediate values [1]. Secret-sharing approaches distribute inputs across parties and perform linear and nonlinear operations through interactive protocols [2]. SecureML demonstrated that two-party computation could support scalable machine learning tasks, including inference and training, by combining secret sharing with garbled circuits [4]. MiniONN transformed neural networks into oblivious primitives, reducing the cost of nonlinear operations through custom protocols [5]. Gazelle introduced a low-latency framework that separated linear layers, which are cheap under homomorphic operations, from nonlinear layers, which require communication [6]. Chameleon explored hybrid secure computation for machine learning applications, balancing arithmetic and Boolean sharing according to operation type [7]. These systems collectively show that private inference is feasible, but they also show that performance is dominated by nonlinear layers.

The second family is homomorphic encryption, which allows computation directly on ciphertexts. CryptoNets demonstrated that neural networks could be evaluated over encrypted data with high throughput, albeit with latency and limited depth [3]. Homomorphic encryption is attractive because it can support non-interactive or low-interaction inference, but activation functions are problematic. The rectified linear unit is not naturally compatible with efficient homomorphic evaluation because it involves a comparison against zero. Practitioners therefore approximate it with polynomials, rational functions, or piecewise constructions. Each approximation consumes multiplicative depth, increases ciphertext size, and introduces numerical error. The resulting system must choose between privacy parameters, model accuracy, and latency. This trade-off is not merely mathematical; it determines whether inference can run at the edge, in a regional cloud, or only in a centralized data center with abundant memory and bandwidth.

The third family is trusted execution environments, which rely on hardware-enforced isolation rather than cryptographic computation. These environments can execute standard neural network operators with relatively low overhead, including rectified linear units, because the hardware protects data in use. However, they require trust in the processor vendor and the attestation infrastructure. They are also vulnerable to microarchitectural side channels, fault injection, and supply-chain compromise. Hybrid designs combine trusted execution with cryptographic protection, using hardware to accelerate preprocessing or to protect keys. In all three families, activation design influences the allocation of work among central processing units, accelerators, and network links. A private inference system is therefore not simply a neural network wrapped in encryption; it is a distributed system whose bottlenecks move as protocols and hardware change.

Recent work has begun to address activation cost directly. One line of research proposes ReLU-reduced neural networks for two-party computation based private inference [8]. Rather than treating the rectified linear unit as immutable, such work asks whether alternative activation schedules, pruning strategies, or architectural substitutions can reduce the number of costly nonlinear operations while preserving accuracy. This insight is important because it reframes activation design as a protocol-aware architectural choice. If a model can achieve comparable accuracy with fewer secure comparisons or lower-degree approximations, the entire system benefits. The same logic applies to quantized activations, which can be

evaluated with lookup tables or Boolean circuits, and to smooth activations, which may be more amenable to polynomial approximation. The remainder of this paper examines how these choices interact with hardware acceleration and deployment.

3. The Activation Bottleneck in Secure Computation

Activation functions are expensive in private inference because they break the linear structure that cryptographic protocols handle efficiently. Matrix multiplication and convolution can often be expressed as additions and multiplications, which map well to homomorphic encryption, secret sharing, or hardware accelerators. Nonlinear activations, by contrast, require comparisons, branching, or high-degree approximation. In secure two-party computation, a rectified linear unit can be computed by comparing a secret-shared value with zero and selecting either zero or the value. That comparison typically requires communication between parties, and the number of comparisons scales with the number of activation neurons and the number of layers. Communication latency therefore grows with network depth, even if the arithmetic itself is fast. This is why low-latency frameworks invest heavily in reducing rounds for nonlinear operations [6], [10].

Homomorphic encryption presents a different version of the same problem. Because encryption schemes support addition and multiplication, a rectified linear unit must be approximated by a polynomial or a piecewise function. Higher-degree approximations improve accuracy but increase multiplicative depth and noise. Bootstrapping can refresh ciphertexts, but it is computationally expensive [15]. Lookup tables can represent arbitrary functions, yet they require large ciphertexts or clever encoding, and they may introduce access patterns that leak information if not carefully designed. As a result, homomorphic inference systems often favor shallow networks, low-degree activations, or hybrid designs that outsource nonlinear layers to a trusted environment. The choice of activation thus determines whether the system remains within a practical noise budget or requires costly refreshing.

Secret-sharing protocols add another dimension. In three-party or multi-party settings, activations can be evaluated using precomputed correlated randomness, oblivious transfer, or garbled circuits. SecureNN showed that three-party computation can support neural network training with efficient nonlinear protocols [13]. ABY3 introduced a mixed-protocol framework that switches between arithmetic, Boolean, and binary secret sharing depending on the operation [14]. CrypTFlow2 focused on practical two-party secure inference, optimizing both linear and nonlinear layers for realistic network architectures [10]. CrypTen provided a machine-learning-friendly framework for secure computation, making it easier to experiment with activation approximations and protocol choices [11]. These systems reveal a general pattern: linear layers benefit from arithmetic sharing, while nonlinear layers benefit from Boolean sharing or garbled circuits. The cost of switching between representations can dominate, so activation design must consider not only the operation itself but also the conversion overhead.

Hardware acceleration can mitigate some costs but not all. Piranha demonstrated that graphics processing units can accelerate secure computation, improving throughput for privacy-preserving protocols [12]. Yet accelerators cannot eliminate communication rounds in multi-party settings. They can speed up local cryptographic operations, but if the protocol requires many round trips, latency remains bounded by network distance. This is a central structural trade-off. A hardware-accelerated private inference system may achieve high throughput in a data center where parties are co-located, but it may perform poorly across geographic regions.

Efficient activation design reduces the number of rounds and the volume of communication, making hardware acceleration more effective. It also reduces the amount of preprocessing material required, which can be the dominant cost in protocols based on oblivious transfer.

Quantization and pruning interact strongly with activation design. Quantized activations can be computed with integer arithmetic or lookup tables, which are often more accelerator-friendly than floating-point polynomials. Pruning can remove entire neurons or channels, reducing the number of activations that must be evaluated securely. Knowledge distillation can train a smaller student network with fewer nonlinearities. However, these techniques may reduce accuracy, and the accuracy loss may be uneven across classes or demographic groups. A private inference system that adopts aggressive activation reduction without fairness auditing risks producing predictions that are both less accurate and less equitable. The activation bottleneck is therefore not only a performance problem but also a model governance problem.

4. Hardware Acceleration Architectures for Private Inference

Hardware acceleration for private inference spans a spectrum from general-purpose processors to specialized cryptographic engines. Graphics processing units are attractive because they offer high parallelism and mature programming ecosystems. They can accelerate matrix multiplication, number theoretic transforms, and garbled circuit generation. Piranha showed that GPU acceleration can substantially improve secure computation performance [12]. However, GPUs are optimized for arithmetic intensity, and many private inference protocols are communication-bound. The accelerator may sit idle while parties exchange shares or wait for network synchronization. Efficient activation design can increase arithmetic intensity by batching activations, fusing operations, or replacing comparisons with table lookups that map well to GPU memory hierarchies. It can also reduce the number of synchronization points, allowing the GPU to remain utilized.

Field-programmable gate arrays offer a different set of trade-offs. They can implement custom data paths for modular arithmetic, polynomial evaluation, and secret-sharing operations. They can achieve low latency and predictable throughput, which is valuable for edge deployment. However, they require significant engineering effort and may struggle with large models that exceed on-chip memory. Activation functions are a natural target for FPGA acceleration because they are repeated across layers and can be implemented as lookup tables, piecewise linear units, or polynomial evaluators. A reconfigurable fabric can also adapt to different cryptographic parameters, such as ring dimensions or security levels. Yet reconfiguration overhead and limited toolchain maturity can impede deployment. The governance implication is that FPGA-based private inference may be more suitable for regulated environments where auditability and deterministic latency matter more than peak throughput.

Application-specific integrated circuits can deliver the best energy efficiency for fixed workloads. A private inference accelerator might dedicate silicon to number theoretic transforms, modular multipliers, oblivious transfer generation, or secure comparison. Efficient activation design influences the silicon area devoted to nonlinear units. If a model uses fewer rectified linear units or replaces them with polynomial approximations, the accelerator can allocate more area to linear layers or memory. If activations are quantized, the accelerator can use smaller arithmetic units and lower-power memory. However, ASICs are inflexible. A change in cryptographic standards, model architecture, or privacy regulation could render the chip obsolete. This creates a tension between efficiency and adaptability. Heterogeneous

systems that combine CPUs, GPUs, FPGAs, and ASICs can mitigate this tension, but they require compilers and runtimes that can partition work across devices while preserving privacy guarantees.

Homomorphic encryption accelerators are a particularly active area. Number theoretic transforms, modular reduction, and bootstrapping are computationally intensive and can benefit from custom hardware. EVA explored compiler techniques for efficient homomorphic computation, including scheduling and parameter selection [16]. CryptFlow provided an end-to-end system for secure TensorFlow inference, bridging high-level models and cryptographic protocols [17]. These systems show that hardware acceleration must be co-designed with compilers and libraries. Activation design enters the picture because polynomial approximations determine the depth of the computation graph. A lower-degree approximation reduces the number of modular multiplications and the frequency of bootstrapping. It also reduces memory traffic, which is often the limiting factor in homomorphic accelerators. A survey of homomorphic encryption schemes highlights that practical deployment depends on balancing security, functionality, and performance [18].

Trusted execution environments occupy a distinct architectural position. They can run standard neural network frameworks with minimal modification, making them attractive for organizations that lack cryptographic expertise. They can accelerate activation functions using ordinary vector instructions or tensor cores. However, they shift the trust boundary from mathematics to hardware. Side-channel attacks, speculative execution flaws, and firmware vulnerabilities can undermine confidentiality. Hybrid approaches use trusted execution for activation-heavy layers and cryptographic protocols for linear layers. This partitioning can reduce cryptographic overhead while maintaining stronger guarantees for sensitive operations. The choice of activation function affects how much work is assigned to the trusted environment. If activations are cheap in plaintext but expensive in cryptography, a hybrid design may push them into the trusted environment. If activations are approximated cheaply in cryptography, the trusted environment may be avoided altogether. Hardware acceleration therefore cannot be evaluated independently of activation design.

5. Efficient Activation Design as a Cross-Layer Systems Problem

Efficient activation design is often framed as a model compression problem. In reality, it is a cross-layer systems problem that touches training, compilation, cryptography, hardware, and governance. At the training layer, researchers can replace rectified linear units with smooth activations, piecewise linear functions, or learned approximations that are more compatible with secure computation. They can also reduce the number of activations through pruning, low-rank factorization, or architectural search. At the compilation layer, activation approximations can be fused with adjacent operations to reduce memory movement and communication rounds. At the cryptographic layer, the choice of activation determines whether arithmetic sharing, Boolean sharing, homomorphic evaluation, or garbled circuits are most efficient. At the hardware layer, activation design affects utilization of tensor cores, lookup tables, and memory bandwidth. At the governance layer, it affects auditability, reproducibility, and fairness.

One promising direction is ReLU-reduced neural network design. The required reference in this paper examines ReLU-reduced neural networks for two-party computation based private inference [8]. The broader lesson is that reducing the number of rectified linear units can lower communication and computation without necessarily sacrificing accuracy, provided that training accounts for the change. This is not simply a matter of deleting activations. It

involves retraining, regularization, and architecture search to preserve representational capacity. The systems benefit is substantial: fewer secure comparisons, fewer garbled circuits, and fewer communication rounds. The hardware benefit is also significant: fewer nonlinear operations mean more time can be spent on dense linear algebra, which accelerators handle efficiently. The privacy benefit is indirect but important: lower cost makes stronger security parameters more feasible within a given latency budget.

Quantized activations offer another path. By restricting activations to a small set of discrete values, implementations can use lookup tables or Boolean circuits instead of floating-point polynomials. This can reduce memory footprint and improve cache behavior. However, quantization introduces approximation error that may interact with cryptographic approximation error. If both the activation and the encryption scheme introduce error, the combined effect can degrade accuracy in ways that are difficult to predict. Careful error analysis and calibration are required. Quantized activations also raise fairness concerns because quantization thresholds may affect different regions of the input space unevenly. A model that performs well on average may perform poorly for underrepresented groups. Efficient activation design must therefore include evaluation beyond aggregate accuracy.

Polynomial approximations are central to homomorphic inference. A low-degree polynomial can approximate a rectified linear unit over a bounded interval, but the approximation error grows outside that interval. Piecewise polynomials can improve accuracy but increase branching and depth. Rational approximations can be more accurate but require division, which is expensive under encryption. Lookup tables can represent arbitrary functions but may leak access patterns. Each approach changes the balance between multiplicative depth, noise growth, and memory usage. Hardware acceleration can evaluate polynomials efficiently, but only if the degree and coefficient precision are chosen with the accelerator in mind. A GPU or FPGA can pipeline polynomial evaluation, while an ASIC can dedicate multipliers to it. The optimal activation design is therefore hardware-dependent and protocol-dependent.

Mixed-protocol execution adds another layer of complexity. A private inference system may evaluate linear layers under homomorphic encryption, nonlinear layers under garbled circuits, and preprocessing under secret sharing. Each transition incurs conversion overhead. Efficient activation design can reduce the number of transitions by grouping nonlinear operations or replacing them with arithmetic-friendly approximations. It can also enable better scheduling. For example, if activations are approximated as low-degree polynomials, they can remain in the arithmetic domain and avoid conversion to Boolean sharing. If they are computed as comparisons, they may require a switch. Compilers must reason about these trade-offs automatically. This is a significant research challenge because the optimal schedule depends on network topology, cryptographic parameters, hardware capabilities, and network latency.

6. Deployment, Governance, and Sustainability Implications

Deployment of hardware-accelerated private inference will likely be heterogeneous. Some workloads will run in centralized clouds with abundant GPUs and specialized cryptographic accelerators. Others will run at the edge, where latency, bandwidth, and energy are constrained. Federated learning architectures may combine local computation with secure aggregation, raising questions about how activations are evaluated on client devices [20]. A hospital may use private inference to query a remote diagnostic model without revealing patient data. A bank may use it to detect fraud without sharing transaction records. A mobile application may use it to personalize recommendations without exposing user behavior. Each

scenario has different trust assumptions, regulatory requirements, and performance constraints. Activation design must be adapted to each context rather than treated as a universal recipe.

Governance is critical because private inference systems make claims about confidentiality that are difficult for non-experts to verify. Certification regimes may require documentation of cryptographic parameters, hardware attestation, side-channel mitigations, and accuracy tolerances. Efficient activation design affects all of these. If a model uses polynomial approximations, the certification must specify the approximation error and its impact on decisions. If it uses trusted execution, the certification must address firmware updates and vulnerability disclosure. If it uses hardware acceleration, the certification must cover supply-chain integrity and export controls. Without such governance, private inference could become a form of privacy theater, offering technical guarantees that are undermined by operational practices.

Sustainability is an often-overlooked dimension. Cryptographic inference is more expensive than plaintext inference, sometimes by orders of magnitude. Hardware acceleration can reduce energy per operation, but the total energy depends on utilization, cooling, and embodied carbon. Efficient activation design reduces the number of expensive nonlinear operations, which can lower energy consumption directly. It can also reduce communication, which lowers network energy. However, aggressive approximation may require retraining, additional validation, and more complex compilers, all of which have costs. A life-cycle assessment should consider data center efficiency, accelerator manufacturing, and device replacement cycles. The goal is not merely fast private inference but sustainable private inference that can be operated at scale without disproportionate environmental impact.

Interoperability and standards are also important. If every vendor implements a proprietary activation approximation or hardware interface, portability suffers. Open standards for cryptographic parameters, model representations, and accelerator interfaces can reduce fragmentation. They can also enable independent auditing. Efficient activation design should be expressible in model formats and compiler intermediate representations so that it can be verified and reproduced. This is a governance as well as an engineering challenge. Standards bodies, regulators, and industry consortia must collaborate to define what constitutes acceptable privacy, accuracy, and efficiency. Otherwise, private inference may remain confined to bespoke deployments rather than becoming a trusted infrastructure.

7. Robustness, Fairness, and Policy Considerations

Robustness is a first-order concern for private inference. Cryptographic protocols can protect confidentiality, but they do not automatically protect integrity or availability. Adversarial inputs can cause misclassification even when privacy is preserved [19]. Fault injection can corrupt secret shares or cause incorrect outputs. Side-channel leakage can reveal information about inputs or keys, especially in trusted execution environments. Efficient activation design interacts with robustness in subtle ways. Polynomial approximations may be more sensitive to input perturbations than exact rectified linear units. Quantized activations may introduce discontinuities that adversaries can exploit. Lookup tables may create access patterns that leak information. A robust private inference system must therefore evaluate not only accuracy under benign conditions but also adversarial robustness and side-channel resistance.

Fairness is another critical dimension. Privacy-preserving inference is often justified as a way to protect vulnerable populations, but it can also obscure bias. If a model is trained on biased data, encryption does not remove that bias. If activation approximations degrade accuracy

unevenly, they may exacerbate existing disparities. If private inference is expensive, it may be available only to well-resourced institutions, leaving smaller clinics, community organizations, and individual users without access. Hardware acceleration can lower costs, but the benefits may accrue to cloud providers and large enterprises. Policy interventions may be needed to ensure equitable access, including public computing infrastructure, open standards, and subsidies for under-resourced sectors. Efficient activation design can help by reducing the computational barrier, but it is not a substitute for deliberate fairness governance.

Policy implications extend to liability, transparency, and accountability. If a private inference service makes a consequential decision, who is responsible when it fails? The model owner, the data owner, the hardware vendor, and the protocol implementer may all share responsibility. Certification and audit trails can clarify roles. Export controls may restrict the deployment of advanced cryptographic hardware, which could limit access in some regions. Data protection regulations may require privacy by design and data minimization, which align with private inference but also impose documentation burdens. Antitrust and competition policy may need to address concentration in the market for private inference accelerators. The technical choices around activation design are therefore embedded in a broader policy ecosystem.

Finally, the research agenda should include longitudinal studies of deployed private inference systems. Most published results report latency, throughput, and accuracy under controlled conditions. There is less evidence about operational reliability, maintenance costs, user trust, and real-world privacy outcomes. Efficient activation design should be evaluated not only in benchmarks but also in clinical, financial, and civic settings. Interdisciplinary collaboration among cryptographers, hardware architects, machine learning researchers, social scientists, and policymakers is essential. The goal is not to optimize a single layer but to build an infrastructure that is private, accurate, efficient, robust, fair, and sustainable.

8. Conclusion

Hardware-accelerated privacy-preserving neural network inference is a systems challenge that cannot be solved by cryptography or machine learning alone. Activation functions are a central bottleneck because they are essential for model quality yet expensive under secure computation. Efficient activation design offers a way to reduce communication, computation, memory, and energy costs while preserving accuracy. This paper has argued that activation choices must be co-designed with hardware accelerators, cryptographic protocols, compilers, and governance frameworks. ReLU-reduced architectures, quantized activations, polynomial approximations, lookup tables, and mixed-protocol execution each offer advantages and impose trade-offs. The optimal approach depends on the threat model, deployment context, hardware platform, and regulatory environment.

The broader implication is that private inference should be treated as critical socio-technical infrastructure. Hardware acceleration can make it practical, but it can also concentrate power and create new dependencies. Efficient activation design can improve sustainability and accessibility, but it can also introduce fairness and robustness risks if deployed without oversight. Future work should focus on automated cross-layer optimization, standardized benchmarking, independent auditing, and equitable access. By aligning model architecture, cryptographic protocol, hardware acceleration, and policy, the research community can move private inference from laboratory demonstrations to trustworthy public infrastructure.

References

1. Yao, A. C. (1986). How to generate and exchange secrets. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science* (pp. 162–167). IEEE.
2. Goldreich, O., Micali, S., & Wigderson, A. (1987). How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing* (pp. 218–229). ACM.
3. 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. In *Proceedings of the 33rd International Conference on Machine Learning* (pp. 201–210). PMLR.
4. Mohassel, P., & Zhang, Y. (2017). SecureML: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy* (pp. 19–38). IEEE.
5. Liu, J., Juuti, M., Lu, Y., & Asokan, N. (2017). Oblivious neural network predictions via MiniONN transformations. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 619–631). ACM.
6. Juvekar, C., Vaikuntanathan, V., & Chandrakasan, A. (2018). GAZELLE: A low latency framework for secure neural network inference. In *27th USENIX Security Symposium* (pp. 1651–1669). USENIX Association.
7. Riazi, M. S., Weinert, C., Tkachenko, O., Songhori, E. M., Schneider, T., & Koushanfar, F. (2018). Chameleon: A hybrid secure computation framework for machine learning applications. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security* (pp. 707–721). ACM.
8. Peng, H., Zhou, S., Luo, Y., Xu, N., Duan, S., Ran, R., ... & Ding, C. (2023). Rrnet: Towards relu-reduced neural network for two-party computation based private inference. *arXiv preprint arXiv:2302.02292*.
9. Mishra, P., Lehmkuhl, R., Srinivasan, A., Zheng, W., & Popa, R. A. (2020). Delphi: A cryptographic inference service for neural networks. In *29th USENIX Security Symposium* (pp. 2505–2522). USENIX Association.
10. Rathee, D., Rathee, M., Kumar, N., Chandran, N., Gupta, D., Rastogi, A., & Sharma, R. (2020). CrypTFlow2: Practical 2-party secure inference. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (pp. 325–342). ACM.
11. Knott, B., Venkataraman, S., Hannun, A., Sengupta, S., Ibrahim, M., & van der Maaten, L. (2021). CrypTen: Secure multi-party computation meets machine learning. In *Advances in Neural Information Processing Systems 34* (pp. 4961–4973). Curran Associates.
12. Watson, J.-L., Wagh, S., Gagliardoni, T., & Gupta, D. (2022). Piranha: A GPU platform for secure computation. In *31st USENIX Security Symposium* (pp. 827–844). USENIX Association.
13. Wagh, S., Gupta, D., & Chandran, N. (2019). SecureNN: 3-party secure computation for neural network training. *Proceedings on Privacy Enhancing Technologies*, 2019(3), 26–49.

14. Mohassel, P., & Rindal, P. (2018). ABY3: A mixed protocol framework for machine learning. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (pp. 35–52). ACM.
15. Chillotti, I., Gama, N., Georgieva, M., & Izabachène, M. (2020). TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1), 34–91.
16. Dathathri, R., Kostova, B., Saarikivi, O., Dai, W., Laine, K., & Lauter, K. (2020). EVA: An encrypted vector arithmetic language and compiler for efficient homomorphic computation. In Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation (pp. 546–561). ACM.
17. Kumar, N., Rathee, M., Chandran, N., Gupta, D., Rastogi, A., & Sharma, R. (2020). CryptTFflow: Secure TensorFlow inference. In 2020 IEEE Symposium on Security and Privacy (pp. 336–353). IEEE.
18. Acar, A., Aksu, H., Uluagac, A. S., & Conti, M. (2018). A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys*, 51(4), Article 79.
19. Papernot, N., McDaniel, P., Sinha, A., & Wellman, M. P. (2018). SoK: Security and privacy in machine learning. In 2018 IEEE European Symposium on Security and Privacy (pp. 399–414). IEEE.
20. Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., ... & Zhao, S. (2021). Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2), 1–210.