IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 27 May 2025, accepted 10 June 2025, date of publication 13 June 2025, date of current version 20 June 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3579667

RESEARCH ARTICLE

ANN Activation Function Estimators for Homomorphic Encrypted Inference

MHD RAJA ABOU HARB AND BARIS CELIKTAS
Computer Engineering Department, Isik University, 34398 Istanbul, Türkiye
Corresponding author: Mhd Raja Abou Harb (21comp9001@isik.edu.tr)

ABSTRACT Homomorphic Encryption (HE) enables secure computations on encrypted data, facilitating machine learning inference in sensitive environments such as healthcare and finance. However, efficiently handling non-linear activation functions, specifically Sigmoid and Tanh, remains a significant computational challenge for encrypted inference using Artificial Neural Networks (ANNs). This study introduces a lightweight, ANN-based estimator designed to accurately approximate activation functions under homomorphic encryption. Unlike traditional polynomial and piecewise linear approximations, the proposed ANN estimators achieve superior accuracy with lower computational overhead associated with bootstrapping or high-degree polynomial techniques. These estimators are trained on plaintext data and seamlessly integrated into encrypted inference pipelines, significantly outperforming conventional methods. Experimental evaluations demonstrate notable improvements, with ANN estimators enhancing accuracy by approximately 2% for Sigmoid and up to 73% for Tanh functions, improving F1-scores by approximately 2% for Sigmoid and up to 88% for Tanh, and markedly reducing Mean Square Error (MSE) by up to 96% compared to polynomial approximations. The ANN estimator achieves an accuracy of 97.70% and an AUC of 0.9997 when integrated into a CNN architecture on the MNIST dataset, and an accuracy of 85.25% with an AUC of 0.9459 on the UCI Heart Disease dataset during ciphertext inference. These results underscore the estimator's practical effectiveness and computational feasibility, making it suitable for secure and efficient ANN inference in encrypted environments.

INDEX TERMS Activation function estimator, artificial neural network, encrypted inference, homomorphic encryption.

I. INTRODUCTION

Over the past decade, cloud computing adoption has surged, driven largely by its cost-effectiveness and the reduction of operational and regulatory risks. A notable outcome of this trend is Machine Learning as a Service (MLaaS), which allows users to access advanced machine learning capabilities without deep technical expertise or significant infrastructure investment. However, the widespread adoption of MLaaS has intensified concerns over data privacy. As a result, considerable research has focused on Privacy-Preserving Machine Learning (PPML) techniques to address these challenges.

The associate editor coordinating the review of this manuscript and approving it for publication was Pinjia Zhang.

Key approaches include differential privacy, which guarantees that adding or removing a single data point has minimal effect on the model's results [1]; federated learning, which enables model training on distributed data without sharing raw data [2]; and homomorphic encryption (HE), which allows computations to be performed on encrypted data without decryption [3]. These methods play a vital role in protecting data privacy while facilitating advanced machine learning functionalities in cloud-based environments.

The integration of HE with ANNs has presented significant challenges, primarily due to the difficulty in introducing non-linearities, which are essential for model performance. These non-linear activation functions (e.g., Sigmoid and Tanh) require complex computations that extend beyond the simple arithmetic operations typically supported by HE schemes.

Developing effective approaches to approximate non-linear activation functions in encrypted environments is a critical area of research within privacy-preserving machine learning. Early works relied on polynomial or piecewise linear approximations—simplifying arithmetic under HE but often failing to capture the full complexity of functions like Sigmoid or Tanh. Despite these limitations, they demonstrated the feasibility of secure non-linear approximations, fueling ongoing innovation in the field.

A major motivation for our study is the growing need to protect sensitive data—such as financial transactions and medical records—within cloud-based machine learning. While HE enables operations on encrypted data, it primarily supports basic arithmetic, leaving non-linear activations as a persistent bottleneck. Existing approximation methods frequently lose precision in rapidly changing regions or require separate solutions for each activation variant, which compromises accuracy and computational feasibility. Hence, we propose a single-layer ANN-based estimator that models these non-linearities under HE more flexibly, aiming to enhance accuracy without incurring the substantial overhead commonly found in higher-degree polynomial or piecewise linear schemes.

Building upon our previous work in [4], which introduced the ANN-based estimator for approximating non-linear activation functions, this study extends the methodology to encrypted inference scenarios. Although our previous work demonstrated the estimator's effectiveness in HE environments, this study further integrates it directly into the encrypted machine learning inference pipeline, enhancing its applicability and performance in secure inference settings. Our approach focuses on enhancing the accuracy and efficiency of encrypted machine learning inference. Unlike traditional polynomial and piecewise linear approximations, ANN-based estimators provide greater flexibility in capturing complex non-linearities while remaining compatible with the constraints of HE. These estimators are trained on plaintext data and seamlessly applied during encrypted inference, ensuring minimal loss in model performance. Extensive experiments reveal that ANN-based estimators surpass traditional methods in both Mean Square Error (MSE) and accuracy, establishing them as a reliable solution for secure machine learning applications. This contribution holds particular importance in cloud-based environments, where safeguarding data privacy and optimizing computational efficiency is crucial.

Additionally, to assess the generalizability and practical applicability of the proposed approach, two supplementary experiments were conducted. In the first, the ANN estimator was integrated into a CNN architecture, demonstrating strong performance under encrypted inference conditions compared to similar previous work. In the second, and to simulate a real-world scenario, the estimator was applied to homomorphically encrypted inference for ANN classifiers trained on the UCI Heart Disease dataset, yielding effective results in a medical prediction context. These experiments further validate the robustness and adaptability of our method in diverse datasets and application domains.

The main contributions of this work are summarized as follows:

- We propose a novel, lightweight ANN-based estimator for efficient and accurate approximation of non-linear activation functions (Sigmoid and Tanh) under HE.
- Unlike existing approaches relying on low-degree polynomial or piecewise linear approximations, our estimator offers flexibility and generalizability, adapting to various activation functions without requiring architectural or encryption parameter changes.
- We provide comprehensive empirical evaluations on the MNIST and UCI Heart Disease datasets, demonstrating substantial improvements in accuracy, F1-score, and mean squared error compared to state-of-the-art polynomial and piecewise estimators, while maintaining practical computational efficiency.
- We validate the effectiveness of our estimator within both ANN and CNN architectures under ciphertext inference, demonstrating its suitability and practical applicability for secure machine learning inference in cloud environments.

The structure of the paper is organized as follows: Section II provides a comprehensive review of related work, discussing prior research efforts on activation function approximation in homomorphic encrypted settings. Section III introduces the necessary preliminaries, including foundational concepts of HE and the rationale for using ANN-based estimators. Section IV presents the proposed solution, describing the overall architecture, the design of the ANN estimator, and the encrypted inference pipeline. Section V details the experimental setup, covering datasets, model configurations, and evaluation metrics. SectionVI provides a thorough analysis of the results, including comparative benchmarks, case studies on real healthcare data, and performance evaluation under encryption. Section VII offers a critical discussion of the findings, limitations, and a security analysis illustrated by the proposed threat model. Finally, Section VIII concludes the paper and outlines directions for future research, particularly on optimizing ANN estimators and empirically evaluating robustness against privacy attacks.

II. PREVIOUS WORKS

To overcome the challenges of computing non-linear activation functions within HE, some earlier studies proposed designing ANNs without activation functions entirely [5]. Another approach involved constructing linear ANNs [6], [7], or using linear approximations of activation functions [8]. While these methods simplify computation under HE, they compromise the core advantages of ANNs by eliminating the non-linear features that are essential for capturing complex data patterns.

To introduce non-linearity, some studies applied the activation function at the final layer after decryption [9]. However, this approach is suboptimal because it restricts the

network's ability to learn complex relationships throughout the intermediate layers. By keeping the hidden layers linear, the model loses its capacity to model intricate decision boundaries, which is a fundamental strength of non-linear activation functions in ANNs.

Other research has explored using Trusted Execution Environments (TEEs) as an alternative to HE for securely performing non-linear activation function computations. TEEs offer a secure, isolated processing environment, protecting sensitive computations and data even within potentially compromised operating systems. However, TEEs introduce hardware dependencies and additional trust assumptions, highlighting the advantages of exploring purely cryptographic solutions such as HE-based ANN estimators. TEEs safeguard data confidentiality by using secure enclaves and remote attestation to verify computation integrity, ensuring protection even if the surrounding environment is compromised [10].

Another approach found in previous work involves using dual-cloud setups without collusion [11], [12], [13]. In this method, one Cloud Service Provider (CSP) performs the classification process, while a second CSP generates key pairs for each client and processes the activation functions on the encrypted weight sums computed by the first cloud. The second cloud, holding the decryption keys, decrypts the sums, processes them with the activation functions, and re-encrypts the results before returning them to the first cloud. While this approach addresses the challenge of non-linear activation in ANNs on homomorphically encrypted data, it relies on trust in the CSP managing the key pairs and increases the vulnerability to potential attacks.

Table 1 summarizes previous works aimed at estimating activation functions for use with HE data.

Among the advanced polynomial-based methods highlighted in Table 1 is the approach of Lee et al. [23], who demonstrated an Residue Number System-Cheon-Kim-Kim-Song (RNS-CKKS)-based implementation of ResNet-20 with bootstrapping to handle deeper networks under HE. Their work focused on minimax polynomial approximations for ReLU and Softmax, which allowed the model to maintain near-native accuracy (92.43% $\pm$ 2.65% on Canadian Institute For Advanced Research (CIFAR)-10) without retraining from scratch on encrypted data. Crucially, they used repeated bootstrapping to manage ciphertext noise across many layers, a key requirement for complex networks such as ResNet-20. Although their study centers on ReLU and Softmax, the fundamental technique—training polynomials to approximate non-linear functions—readily extends to other activations, including Sigmoid and Tanh, by adjusting the approximation intervals and polynomial degrees.

Table 1 indicates that time complexity is a common challenge. To address this, many solutions rely on low-degree polynomial approximations for activation functions, as these are more compatible with HE schemes such as Cheon-Kim-Kim-Song (CKKS) and Brakerski/Fan-Vercauteren (BFV). These polynomial approximations simplify computations on encrypted data and help mitigate computational burden. However, the reliance on low-degree polynomials comes with a trade-off: they struggle to accurately capture the intricate behavior of non-linear activation functions (e.g., Sigmoid and Tanh), leading to reduced model expressiveness and accuracy. This limitation highlights the ongoing challenge of balancing computational efficiency with the effectiveness of neural networks in secure computations on encrypted data.

Although some studies report good performance in the classification problems they solve, many of them employ advanced noise-management strategies—such as frequent bootstrapping—that significantly inflate computational overhead and system complexity. In contrast, our approach forgoes bootstrapping in favor of a simpler architecture, showing that secure inference can be made more practical without incurring the heavy resource demands associated with noise-resetting procedures. This design choice naturally places an emphasis on the feasibility and ease of integration in PPML environments, rather than solely on maximizing performance metrics.

In this study, we propose a new method for approximating non-linear activation functions through a streamlined ANN design. Our strategy draws on the Universal Approximation Theorem, which states that a sufficiently large neural network with suitable parameters can approximate any continuous function to a chosen level of precision [29]. By leveraging this principle, we employ a lightweight ANN to model and approximate complex non-linear activation functions—such as Sigmoid and Tanh—without relying on more specialized polynomial expansions or piecewise linear constructs.

Because our approach can be adapted to various activation functions, it yields accurate approximations for a range of non-linearities. This adaptability makes it notably valuable in scenarios requiring flexibility and generalization since the same ANN-based framework can handle multiple activation functions. Moreover, the simple architecture of ANN keeps computational costs manageable, which is advantageous for PPML environments where both efficiency and accuracy are paramount, all while reducing implementation complexity through the avoidance of bootstrapping.

III. PRELIMINARIES

A. BASIC CONCEPTS OF HE

HE is a cryptographic technique that enables computations to be performed directly on encrypted data without the need for decryption. This capability ensures that sensitive information remains secure throughout the processing, making HE an ideal solution for preserving privacy in untrusted environments. Due to this feature, HE is specifically well-suited for MLaaS platforms, as it allows the use of advanced machine learning techniques while keeping users' data confidential.

A typical HE system includes several essential components: representations of data in both plaintext and

TABLE 1. Summary of previous works on activation function estimation for HE-based ANNs.

Ref.	Model or framework	HE scheme	Activation function handling	Datasets	Performance	Limitations
Usman et al. (2025) [14]	HoRNS-CNN: privacy-preserving deep CNN with RNS-FHE for neuro-biomarker classification	RNS-FHE; energy-efficient, FPGA-friendly, 3-moduli set	ReLU replaced by degree-3 Taylor polynomial; BN for stability & OpenNeuro	C-BIRD MRI (patch-based, 148 subjects, dyslexia vs. control)	91.4% accuracy (encrypted), 400k features/hr, 42.4% energy saving	Decryption slower than encryption; tested only on binary task; cipher expansion; hardware eval only
Allavarpu et al. (2025) [15]	Neural network for credit risk analysis integrating TenSEAL and Torch	CKKS (via TenSEAL library)	Uses approximate arithmetic for NN layers; does not explicitly detail specialized polynomial approximations for activation functions	Real-world financial datasets from multiple countries	Comparable accuracy with/without HE (e.g., minor F1 drop); validated resilience in credit-risk classification	Overhead associated with encrypted data operations; mainly tested on tabular financial data rather than large-scale image or deep networks
Lee Junghyun (2024) [16]	Polynomial-approximation-based inference on large-scale CNNs (ResNet, VGG)	RNS-CKKS	Proposes composite minimax and weighted least-squares polynomials for ReLU and max-pooling; allows evaluation of deep nets on encrypted data	ImageNet, CIFAR	Can reach 77% top-1 on ImageNet (ResNet-152) via polynomial-based ReLU under encryption	Expensive bootstrapping for deeper networks; polynomial management is complex; still large run-time overhead relative to plaintext
Xiong et al. (2024) [17]	CNN with "Self-Learnable Activation Function" (SLAF) for HE	CKKS-based fully homomorphic approach	Learns polynomial-like activation parameters to reduce approximation error under HE constraints	UTKFace (face recognition)	Accuracy gains of 0.88–3.15% over standard polynomial methods, 4.87–9.67% over CryptoNets for face-verification	Tested on a single face dataset; learning custom activations adds overhead; large-scale or more complex tasks not extensively shown
Song and Shi (2024) [18]	ReActHE	CKKS	Scaled power function on residues	Heart Failure, Stroke, Hepatitis C, TCGA, SARS-CoV-2, Yeast Genomics	COVID dataset: 98.87%, Heart Failure: 91.14%, Stroke: 89.27%	High computational cost, challenges in optimizing deep networks
Zhang et al. (2024) [19]	CrossNet	CKKS	2nd-degree polynomials	MNIST, CIFAR-10	MNIST: 98.70%, CIFAR-10: Competitive with state-of-the-art	Computational and communication overhead, client dependency
Shi and Zhao (2023) [20]	Binary convolutional neural network (BCNN) over a vector-based homomorphic encryption (VHE)	Proposed "efficient integer vector" VHE scheme	Binarizes activation to ± 1 (sign) for simpler polynomial overhead	MNIST	93.75% train accuracy, 86% test accuracy with reduced overhead	Binarization lowers representational capacity; mostly tested on simpler MNIST tasks
Khan and Michalas (2023) [21]	CNN with Chebyshev Polynomials	HE with Chebyshev Polynomials	Low-degree Chebyshev polynomials	MNIST	MNIST: 98.5%	Limited generalizability due to CNN dependency
Nguyen et al. (2023) [22]	HeFUN	CKKS	Polynomial approximation for activation functions like ReLU	MNIST, CIFAR-10	MNIST: 98.9%, CIFAR-10: 85.7%	Computational complexity due to polynomial approximations, increased latency for larger models
Lee et al. (2022) [23]	ResNet-20 with RNS-CKKS	RNS-CKKS	Minimax polynomial for ReLU, Softmax approximation	CIFAR-10	CIFAR-10: 92.43% $\pm$ 2.65%	High memory requirements, complexity due to bootstrapping
Hong et al. (2022) [24]	Shallow Neural Network (SNN)	CKKS	Polynomial approximation of Softmax	Cancer Genome Atlas (TCGA)	TCGA: 85%	Challenges with polynomial approximation of Softmax
Xiong et al. (2020) [25]	Encrypted CNN Ensemble	BFV	Low-degree polynomial approximations for non-linear operations	MNIST	Accuracy: 96%-97% with CNN ensemble	High computational overhead and time intensive due to ensemble structure
Vizitiu et al. (2020) [26]	Deep neural network on encrypted medical data (classification tasks)	Matrix Operation for Randomization or Encryption (MORE)	Approximates non-linear layers with rational polynomial expansions or direct floating-point homomorphic support; focuses on ReLU-like replacements	MNIST, medical data (e.g., X-ray coronary angiography)	Comparable classification accuracy to plaintext; overhead grows with network depth but demonstrated feasible for real medical scenarios	Encryption overhead increases computation time; specialized scheme (MORE) not as widely adopted as CKKS/BFV; potential memory overhead
Izabachène et al. (2019) [27]	Fully masked Hopfield neural network for face recognition	GSW-type FHE variant	Avoids standard ReLU by using Hopfield-style network with discrete sign-like updates; no standard polynomial activation	Small face-recognition sets (e.g., ORL, Yale)	0.6s per recognition on standard CPU; secure evaluation feasible for face recognition	Specialized for Hopfield nets; limited scalability to large modern DNNs; specialized non-standard activation approach
Hesamifard et al. (2018) [28]	CryptoDL Framework	HELib (Leveled HE)	Polynomial approximations for Sigmoid, ReLU, Tanh function	UCI datasets, MNIST, CIFAR-10	MNIST: 99%, CIFAR-10: 91.5% with polynomial approximation	High computational overhead, limited to low-degree polynomials for practicality

ciphertext forms, the encryption and decryption processes, and homomorphic operations. During encryption, plaintext is converted to ciphertext using an encryption key; during decryption, a decryption key is used to recover the original plaintext. HE stands out by allowing arithmetic operations (such as addition and multiplication) to be performed directly on ciphertexts, producing encrypted results that, when decrypted, match those of plaintext computation. For instance, if $E(a)$ and $E(b)$ are encrypted versions of a and b, then $E(a) \times E(b)$ results in an encrypted representation of $a \times b$.

HE schemes can be divided into three main types based on the breadth of operations they enable: Partially Homomorphic Encryption (PHE), Somewhat Homomorphic Encryption (SHE), and Fully Homomorphic Encryption (FHE). With PHE, users can perform an unlimited number of operations of a single type—either repeated additions or repeated multiplications—on encrypted data. For instance, the Rivest-Shamir-Adleman (RSA) scheme supports unlimited multiplications, while the Paillier scheme supports unlimited additions. However, PHE cannot accommodate tasks requiring both addition and multiplication.

SHE takes this a step further by permitting both addition and multiplication, but only up to a limited number of operations. This constraint arises because each operation adds noise, which accumulates over time and can eventually render decryption unfeasible. Despite this limitation, SHE still offers more computational flexibility than PHE and paves the way for transitioning to FHE.

FHE addresses the limitations of both PHE and SHE by enabling unlimited number of additions and multiplications on encrypted data. This capability supports complex workloads, including training and inference for machine learning models. Nonetheless, FHE requires substantial computational resources and memory. While recent progress, such as bootstrapping, has made FHE more viable, its deployment in real-world scenarios remains an ongoing challenge. Understanding these distinctions is crucial for selecting the appropriate HE scheme based on the requirements of privacy-preserving machine learning tasks.

In this study, we employ the CKKS scheme for HE, a suitable scheme for computations involving real numbers and floating-point arithmetic. The CKKS scheme allows approximate arithmetic on encrypted data, making it ideal for applications in machine learning where precise calculations on non-integer values are essential, such as during the inference phase of neural networks with activation functions like Sigmoid and Tanh. For a complete understanding of the underlying HE scheme and its configuration with the mathematical foundations, please refer to [30].

B. CHALLENGES WITH NON-LINEAR FUNCTIONS IN HE

As noted above, HE schemes make it possible to carry out fundamental arithmetic operations—like addition and multiplication—directly on encrypted data without decrypting it. While these operations suffice for evaluating simple mathematical expressions, they present considerable difficulties in computing non-linear activation functions (e.g., Sigmoid, Tanh, and ReLU), which are essential for effective ANN performance [21].

The primary difficulty arises from the inherent complexity of non-linear functions. Unlike addition and multiplication, non-linear functions involve complex operations like exponentiation and division, which are not directly supported by most HE schemes. These functions require a series of transformations that accumulate noise with each operation, often leading to a rapid increase in the levels of ciphertext noise, making it difficult to maintain precision during decryption.

Furthermore, non-linear functions are essential for giving ANNs non-linearity, which helps these models identify intricate patterns and connections in the data. Without non-linear activation functions, ANNs become equivalent to linear models, significantly limiting their expressive power and making them unable to learn intricate decision boundaries. This means that applying HE directly to ANN models without an efficient approximation of these non-linear functions can result in poor model performance and reduced accuracy.

The challenge lies in finding an approximation method that balances accuracy with computational feasibility, ensuring that the encrypted computations remain efficient while preserving the model's predictive capabilities. This issue continues to be a critical focus of research in the field of PPML, as the ability to effectively approximate non-linear functions directly impacts the viability of using HE in real-world applications involving ANNs.

C. OVERVIEW OF THE CURRENT ACTIVATION FUNCTION APPROXIMATIONS

In this section, we explore the current methods for approximating non-linear activation functions, which are essential for the performance of neural networks. These approximations are particularly important when working with homomorphically encrypted data, where direct computation of non-linear functions is challenging. We focus on two widely used approaches: polynomial estimators and piecewise linear approximations. Each method provides a different balance between computational complexity and approximation accuracy, making them suitable for various encrypted computing scenarios.

1) POLYNOMIAL APPROXIMATION

This approach employs polynomial functions to approximate non-linear activation functions like Sigmoid and Tanh. Polynomial approximations are computationally efficient, avoiding complex operations such as exponentiation, which makes them well-suited for environments where managing the noise budget is crucial, such as HE. However, there is a trade-off between the polynomial's degree and the precision of its approximation. Higher-degree polynomials

can more accurately capture the behavior of non-linear functions, but they require more operations. In the context of HE, this increased complexity leads to longer computation times and greater noise accumulation in the encrypted data. Therefore, a common choice is a second-degree polynomial, which strikes a balance between simplicity and computational feasibility. Despite this, a second-degree polynomial may struggle to accurately approximate complex non-linear functions, resulting in reduced precision. Although using a higher-degree polynomial could improve the approximation, the associated noise growth during encrypted operations makes this approach impractical for maintaining decryption accuracy.

In this work, we used second-degree estimators of Sigmoid and Tanh, which is a common way used in the literature. For the Sigmoid activation function, we use the polynomial approximation $\text{Sigmoid}_{\text{poly}}(x) \approx 0.5 + 0.197x - 0.004x^2$, while for the Tanh activation function, the approximation is defined by the equation $\text{Tanh}_{\text{poly}}(x) \approx 0.0 + 1.0x - 0.165x^2$. These forms were chosen to provide a balance between simplicity and accuracy, allowing efficient computation under HE constraints while maintaining reasonable approximations of the Sigmoid and Tanh curves. They were determined through empirical analysis of the functions. Figure 1 and Figure 2 show the polynomial approximations for Sigmoid and Tanh and comparison with the correct values for these activation functions, respectively.

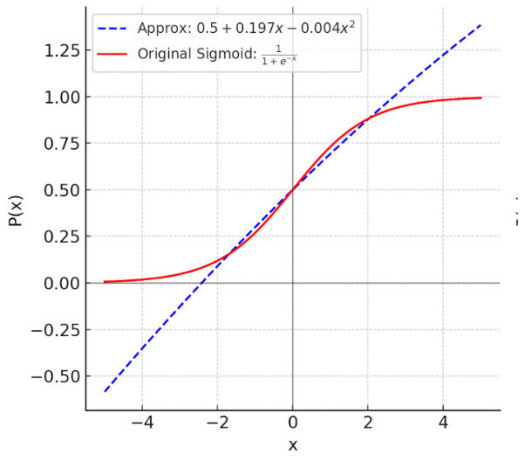

FIGURE 1. Sigmoid polynomial approximation and the original Sigmoid.

2) PIECEWISE LINEAR APPROXIMATION

This method approximates non-linear functions by dividing their range into segments and fitting a linear function within each segment. This allows for a piecewise linear approximation that can capture the overall shape of the function with relatively simple calculations, making it suitable for encrypted environments. It strikes a balance between accuracy and computational complexity, as each segment can adapt to different parts of the function's curve.

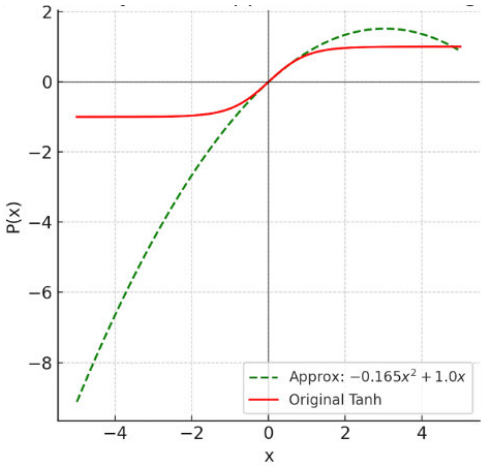

FIGURE 2. Tanh polynomial approximation and the original Tanh.

A key challenge of this approach is determining the appropriate range for each segment when working with homomorphically encrypted data. Since the data remains encrypted throughout the process, directly identifying ranges is not feasible, making it difficult to select optimal breakpoints for the piecewise approximation. To address this, one potential solution is to determine the ranges on plaintext data during the training phase and then apply those predetermined ranges during inference on encrypted data. Although this approach preserves approximation accuracy, it introduces additional complexity and requires meticulous calibration to maintain consistency throughout both training and inference stages.

Algorithm 1 details the computations of the non-linear activation functions using piecewise linear approximations.

Algorithm 1 Piecewise Linear Approximation for Nonlinear Activation Functions With HE

Require: Encrypted tensor x, breakpoints, and segment coefficients (a, b).

Ensure: Encrypted tensor with approximated nonlinear values.

1: Initialize encrypted result tensor $r \leftarrow 0$.
2: **for each segment** i **do**
3: Retrieve coefficients (a, b) and define the segment range.
4: Create a mask for x based on the segment range using approximate Heaviside functions.
5: Compute the segment approximation $s_i \leftarrow a \cdot x + b$.
6: Update $r \leftarrow r + \text{mask} \cdot s_i$.
7: **end for**
8: **Return** r.

The predefined breakpoints for the Sigmoid activation function were set to $[-6.0, -3.0, 0.0, 3.0, 6.0]$, while for the Tanh activation function, the breakpoints were $[-6.0, -3.0, 0.0, 3.0, 6.0]$. Algorithm 1 approximates the nonlinear activation functions using a piecewise linear

approach that is compatible with HE. The range of input values is divided into segments, each defined by a breakpoint range and a linear equation, which allows efficient approximation of the non-linear activation function when direct computation is not feasible due to the constraints of encrypted data processing.

The process starts by initializing an output tensor to store the results. For each segment, a linear approximation is applied to the input values within the defined range. Since the data remain encrypted, direct comparison operations are not possible. Instead, approximate Heaviside functions are used to create masks that identify which inputs belong to each segment.

For values less than or equal to the first breakpoint, the algorithm creates a mask to select those values. For values greater than the last breakpoint, a different mask is applied. For intermediate ranges, two masks are used to define the boundaries of the segment, ensuring that only values within the specified range are affected by the corresponding linear function.

Each linear approximation is calculated using simple arithmetic operations—multiplication and addition—that are supported by HE schemes. The masked segment is then added to the overall result, combining the contributions of each segment to produce the final output.

D. MOTIVATION FOR ANN-BASED ESTIMATORS

The motivation behind adopting ANN-based estimators for approximating non-linear activation functions lies in achieving a flexible, uniform framework for handling various activations, such as Sigmoid and Tanh, without altering the underlying model structure. Traditional approaches often rely on separate, function-specific approximation techniques (e.g., polynomial or piecewise linear), each with its own setup requirements and limited ability to accurately capture the complex behaviors of non-linear functions.

A key advantage of our ANN-based estimator is its simplicity: switching to a different activation function only requires retraining, with no need to modify the network architecture. Specifically, to approximate a new non-linear activation (e.g., ReLU or Softmax instead of Sigmoid/Tanh), we simply recollect the preactivation outputs (from the main ANN's hidden layer) and pair them with the corresponding target post-activation values for the new function. This effectively yields a new "training set" for the single-layer ANN estimator, reflecting how the activation function transforms inputs over its valid range. Because the estimator learns a general mapping from pre- to post-activation values, no special polynomial expansions, breakpoints, or manually tuned approximations are required. Only the reference dataset—i.e., pairs of hidden-layer outputs and their correct activation values—must be regenerated, making the process both flexible and efficient. This design also shortens the development cycle: adjusting the estimator for another activation function requires only standard regression training on the updated dataset, without complex modifications to the neural network architecture or encryption parameters.

ANN-based estimators can additionally capture subtle, nuanced behaviors of non-linear activation functions by learning adaptively from training samples over the full function range. This allows them to model smooth transitions and inflection points more precisely than simpler polynomial or piecewise linear methods might miss. Unlike low-degree polynomial approximations—which often struggle with accuracy—or piecewise linear approximations—whose precision can vary across breakpoints—the proposed estimator can learn closer to the actual shape and characteristics of the function. This adaptability is especially beneficial in HE settings, where consistent high-fidelity approximations with minimal noise accumulation are crucial.

IV. PROPOSED SOLUTION

A. OVERVIEW OF THE PROPOSED SOLUTION

To provide a clear understanding of the proposed solution, Figure 3 illustrates the dataflow diagram. The dataflow shows four preparatory steps required to obtain the ANN-based estimator. First, the range of input values for which the activation functions need to be approximated is determined. Next, the training dataset is prepared by computing activation function values for samples within the defined range. This dataset forms the basis for training the ANN estimator, which captures the behavior of the activation function over the relevant range. The ANN is trained in plaintext mode, thus avoiding the noise build-up that would occur with homomorphically encrypted calculations. Once trained, the ANN estimator is utilized during encrypted inference, effectively addressing the core challenge that this approach seeks to resolve.

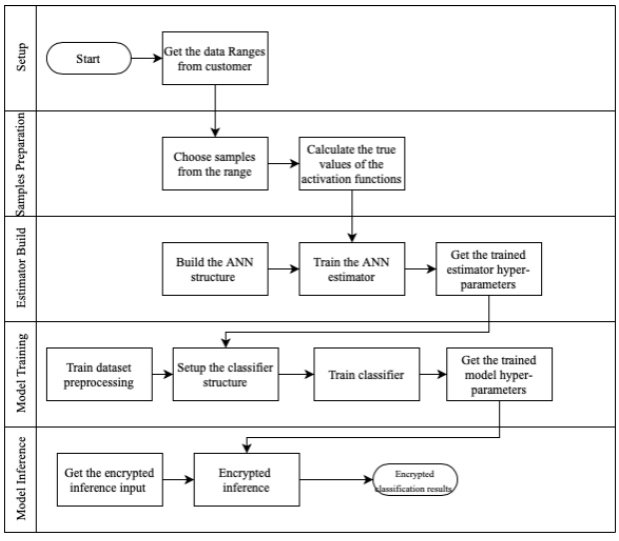

FIGURE 3. The dataflow diagram of the proposed work.

To illustrate a real-world implementation of the proposed solution, the communication diagram in Figure 4 presents a

healthcare example. The diagram features four main actors: the CSP, which offers MLaaS; the Key Management System (KMS), responsible for generating and managing encryption key pairs; the healthcare center; and the patient. During the training phase, the healthcare center preprocesses the training dataset and shares it with the CSP. The CSP then utilizes its computational resources to train the ANN model for the classification or regression task specified by the healthcare center. Following this, the CSP prepares the ANN estimator to handle activation functions in a homomorphically encrypted mode. This architecture is designed to resist common privacy threats such as Man-in-the-Middle (MitM) interception, model inversion, or inference-time data leakage since no raw data or decrypted model output is exposed to the cloud or external observers.

During the inference phase, the patient retrieves the encryption key pairs from the KMS. The public key is used to encrypt the input data for inference before it is transmitted to the CSP for processing. After doing the inference on the encrypted data, the CSP gives the patient the encrypted results, which they can decrypt using the private key to obtain the final classification or regression result.

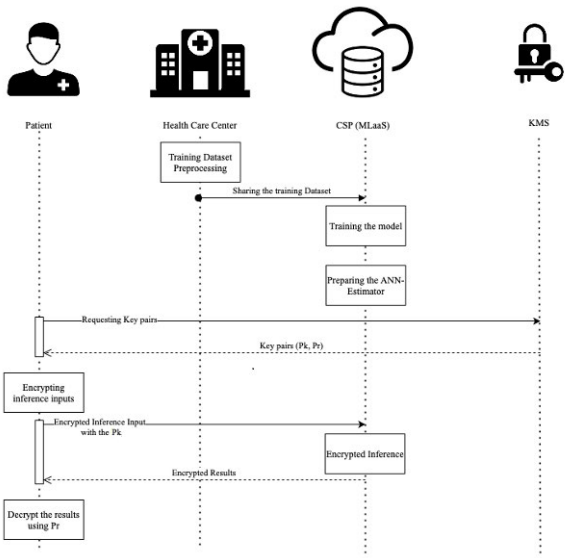

FIGURE 4. Communication diagram for implementation example in healthcare.

B. DESIGN AND TRAINING OF THE ANN ESTIMATOR

The proposed ANN-based estimator is implemented as a lightweight, single-layer fully connected network, where the input and output dimensions are set to match the size of the main ANN's hidden layer (64 units in our experiments on the MNIST dataset). The estimator serves to approximate the non-linear activation function within the encrypted inference pipeline. Its main components and training protocol are as follows:

- Input and Output: For each inference, the estimator receives the preactivation vector from the main ANN's hidden layer ($\mathbf{z}_1 \in \mathbb{R}^{64}$) and produces an output vector ($\mathbf{a}_1 \in \mathbb{R}^{64}$), thereby approximating the true activation function values (e.g., Sigmoid or Tanh).
- Architecture: The estimator consists of a single fully connected layer. This lightweight design is chosen to minimize computational overhead and avoid introducing additional noise in the HE context.
- Training Dataset: The training data is constructed by pairing the preactivation outputs from the main ANN's hidden layer with their corresponding true post-activation values, effectively forming input-output pairs for supervised regression.
- Training Procedure: Training is conducted in the plaintext domain as a regression task, where the estimator minimizes the MSE between its outputs and the reference activation values. The Adam optimizer is employed (learning rate $= 0.0005$, weight decay $= 1e^{-5}$) over 10 epochs, with a batch size consistent with the main ANN's protocol. Weight clipping with a threshold of 5.0 is applied after each update to ensure stable training and maintain compatibility with HE.
- Weight Initialization: Weights are initialized using the Xavier (Glorot) method [30], which is particularly effective for preventing vanishing or exploding gradients in networks approximating non-linear activations like Sigmoid and Tanh.

This architectural choice achieves a balance between approximation accuracy and computational efficiency, ensuring seamless integration into the encrypted inference pipeline without incurring significant computational overhead or excessive ciphertext noise. Empirical results demonstrate that the ANN estimator achieves high-fidelity approximation of non-linear activation functions in encrypted settings. Figures 5 and 6 illustrate the ANN estimator's performance in approximating the Sigmoid and Tanh functions, respectively.

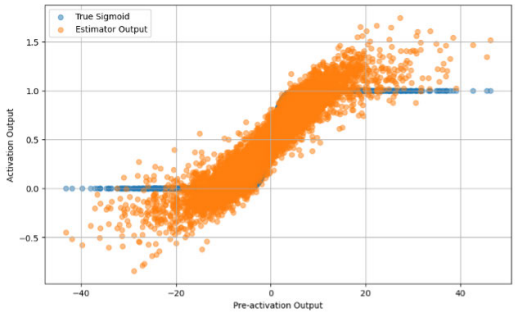

FIGURE 5. Sigmoid ANN estimator and the original Sigmoid.

C. DESIGN AND TRAINING OF THE MAIN ANN

The main ANN in this work is intentionally designed as a shallow model, serving as the primary network for classification or regression tasks, with a particular focus on the MNIST handwritten digit dataset. Its structure is kept minimal and fixed throughout all experimental phases,

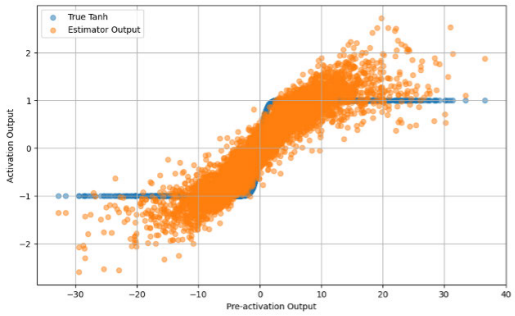

FIGURE 6. Tanh ANN estimator and the original Tanh.

prioritizing the evaluation of activation function estimators under HE rather than optimizing classification accuracy with deeper architectures. The ANN consists of the following layers:

- Input Layer: Configured to accept flattened input data—such as a 784-dimensional vector obtained by flattening each 28 × 28 grayscale MNIST image—the input layer can also be adapted to other structured datasets. This design choice provides versatility for different types of input while maintaining compatibility with HE requirements.
- Hidden Layer: Fully connected to the input layer, the hidden layer comprises 64 neurons. A non-linear activation function (either Sigmoid or Tanh, depending on the estimator under evaluation) is applied after this layer to enable the network to capture complex, non-linear relationships in the data. The choice of 64 units balances model expressiveness with computational efficiency, which is critical given the noise and processing constraints inherent to HE.
- Output Layer: The configuration of the output layer depends on the specific task. For classification tasks such as MNIST, it features 10 units, each corresponding to a digit class; for regression, the layer can be configured to produce continuous outputs.

We train the main ANN in plaintext mode before adapting it for encrypted inference. The model is implemented in PyTorch and trained using the Adam optimizer (learning rate 0.001) to minimize cross-entropy loss—a standard objective for classification tasks—over 10 epochs. Input features are standardized using normalization, and training is conducted with a batch size of 64. A validation subset is extracted from the training set to monitor model performance and convergence during training. Throughout this phase, the total training time is recorded. Once training is complete, the main ANN's weights are finalized and prepared for integration with the ANN estimator in the encrypted inference pipeline.

By maintaining a simple, single-hidden-layer architecture, the computational complexity is reduced, which ensures the model remains practical for encrypted inference. This simplicity also enables clearer insights into the performance of activation function estimators, avoiding additional complexity from deeper or more intricate network designs.

D. HOMOMORPHICALLY ENCRYPTED INFERENCE PHASE

Algorithm 2 summarizes the steps related to the homomorphically encrypted inference phase.

Algorithm 2 Homomorphically Encrypted Inference

Require: Encrypted input data x, trained ANN and ANN estimator parameters $(W_1, b_1, W_2, b_2, W_e, b_e)$, encryption context.

Ensure: Encrypted inference result y and computation time.

Step 1: Encode Model Parameters
Extract trained weights and biases:
    (W_1, b_1): First layer weights and biases of the main ANN.
    (W_2, b_2): Second layer weights and biases of the main ANN.
    (W_e, b_e): Weights and biases of the ANN estimator.
Encode each parameter as plaintext tensors compatible with the encryption context.
Step 2: Compute First Layer Transformation
Perform the first linear transformation:
    $z_1 \leftarrow W_1 \cdot x + b_1$ (element-wise operations on encrypted data).
z_1 represents the encrypted preactivation values of the hidden layer.
Step 3: Apply Activation Function Using ANN Estimator
Use the ANN estimator to approximate the activation function:
    $a_1 \leftarrow W_e \cdot z_1 + b_e$ (encrypted approximation of Sigmoid or Tanh).
a_1 represents the encrypted post-activation values of the hidden layer.
Step 4: Compute Second Layer Transformation
Perform the final linear transformation:
    $y \leftarrow W_2 \cdot a_1 + b_2$ (encrypted transformation for prediction).
y represents the encrypted final output of the ANN.
Step 5: Return Results
return Encrypted output y and total computation time.

Once the main ANN computes the *encrypted preactivation outputs* from the hidden layer in Step 2 of Algorithm 2, those ciphertext values z_1 are directly fed into our single-layer ANN estimator (Step 3). The estimator replaces the usual Sigmoid or Tanh by mapping z_1 to encrypted post-activation outputs a_1. Crucially, there is no decryption step at this stage; a_1 remains encrypted. Finally, a_1 is passed back into the main ANN for the second layer transformation Step 4), functioning as if it were a standard non-linear activation output in plaintext. Performing all operations on encrypted data ensures privacy throughout the inference process.

To further clarify these data flow steps, we now detail the underlying mathematical operations of the algorithm. The encrypted inference process begins with the encrypted input data, represented as X_{enc}, which is derived by encoding and encrypting the original plaintext input X using a HE scheme. This encrypted input X_{enc} allows computations to be performed directly on encrypted data without decryption, preserving the privacy of sensitive information throughout the inference process.

Next, the trained model parameters (weights and biases) of both the main ANN and the ANN estimator are converted to plaintext tensors (indicated with a "pt" superscript), rendering them compatible with HE operations. Let n represent the number of input features, h the number of hidden units in the main ANN, and o the number of output classes or continuous outputs. The first layer of the main ANN has weights $W_{\text{fc1}} \in \mathbb{R}^{h\times n}$ and biases $b_{\text{fc1}} \in \mathbb{R}^{h}$. These parameters are applied to the encrypted input X_{enc} to compute the encrypted preactivation values of the hidden layer, denoted by:

$$h_{\text{enc}} = X_{\text{enc}} \cdot W_{\text{fc1}}^{\text{pt}} + b_{\text{fc1}}^{\text{pt}}$$

The encrypted preactivation values h_{enc} are then processed through the ANN estimator, which approximates the activation function, which may be Sigmoid or Tanh based on the trained datasets. This involves applying the estimator weights $W_{\text{est}} \in \mathbb{R}^{h\times h}$ and biases $b_{\text{est}} \in \mathbb{R}^{h}$, resulting in encrypted post-activation values:

$$h_{\text{est\_enc}} = h_{\text{enc}} \cdot W_{\text{est}}^{\text{pt}} + b_{\text{est}}^{\text{pt}}$$

This representation effectively approximates the activation functions within the HE context, avoiding direct computation of the function and ensuring efficient encrypted operations.

Finally, the encrypted post-activation values $h_{\text{est\_enc}}$ are passed through the second layer of the main ANN, which uses weights $W_{\text{fc2}} \in \mathbb{R}^{o\times h}$ and biases $b_{\text{fc2}} \in \mathbb{R}^{o}$ to produce the encrypted final output:

$$y_{\text{enc}} = h_{\text{est\_enc}} \cdot W_{\text{fc2}}^{\text{pt}} + b_{\text{fc2}}^{\text{pt}}$$

This output, y_{enc}, represents the results of the ANN.

Throughout the encrypted inference phase, in is important to note that all user data—including the input features, hidden layer preactivation vectors, estimator outputs, and final network outputs—remain encrypted in ciphertext form at all times. The only components maintained in plaintext are the model parameters (weights and biases) of both the main ANN and the ANN-based estimator, which are encoded as plaintext tensors for computational efficiency within the HE framework. This design ensures that no intermediate activations or sensitive user information are exposed in plaintext during any step of inference. It is important to note that keeping the model parameters in plaintext does not compromise user privacy, as these parameters are independent of individual user data and are typically considered non-sensitive in privacy-preserving machine learning deployments.

For a deeper understanding of the inference process and how it is influenced by the bias-variance trade-off, including its impact on model performance and generalization, see [31].

E. ADAPTION OF HE

In this work, we adopted the CKKS scheme for HE due to its suitability for computations on real-valued data, which is essential for machine learning tasks involving continuous numbers, such as neural network inference. Unlike other encryption schemes that primarily handle integer data, CKKS enables approximate arithmetic on encrypted floating-point values, making it ideal for implementing activation functions and other mathematical operations directly on encrypted data without decryption. This flexibility enables efficient execution of complex computations directly on encrypted data, making it highly suitable for secure machine learning inference in cloud-based environments.

Managing noise accumulation in HE is critical for accurate computations. Each operation adds noise to the ciphertext, which must be controlled to prevent loss of accuracy or decryptability. To address this, we carefully configured the encryption parameters, such as the polynomial modulus degree and coefficient modulus sizes, to provide an optimal balance between noise tolerance and computational efficiency. Additionally, we set a high global scale factor in CKKS to enhance precision, allowing us to maintain a lower noise level during calculations. This setup minimizes the noise growth across layers of computation, ensuring that the results remain precise and decryptable at the end of the inference process.

V. EXPERIMENT DETAILS

In this section, we give a detailed account of the experiments designed to assess the proposed ANN estimator's ability to approximate activation functions—an essential step toward enabling accurate and efficient inference within a HE framework.

The dataflow diagram in Figure 7 illustrates the experimental setup. As shown, six different inference tests were performed, each aimed at assessing the effectiveness of polynomial, piecewise linear, and ANN estimators for both Sigmoid and Tanh activation functions. These tests were organized into two categories, one for Sigmoid and the other for Tanh activation functions, with each category using a distinct ANN structure optimized for approximating the corresponding activation function. This structured approach allowed us to thoroughly evaluate each estimator's performance across the two key non-linear activation functions.

A. EXPERIMENT SETUP

For our experiments, we utilized Google Colab, a cloud-based platform that provides robust computational resources suitable for machine learning and deep learning research. Specifically, we leveraged a Colab instance equipped with a TPU v2-8, which offered ample computational power and memory to efficiently run complex models and manage the

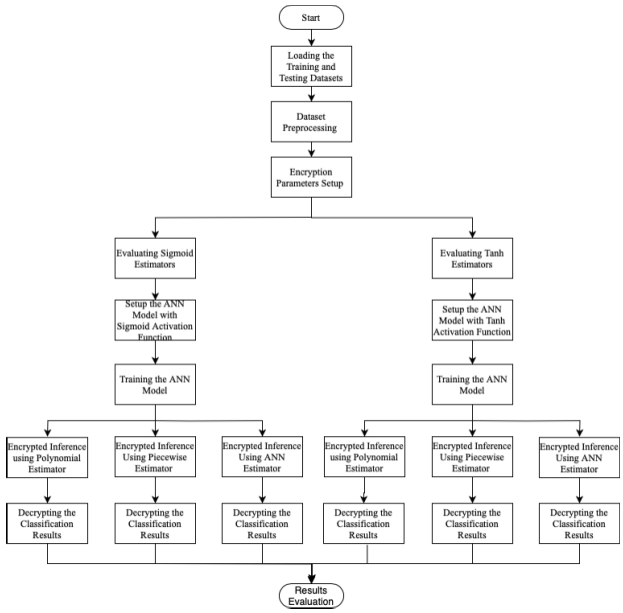

FIGURE 7. Experiment dataflow diagram.

intensive demands of HE operations, utilizing GPU and TPU acceleration as needed.

We utilized several libraries to implement and evaluate the proposed solution efficiently. PyTorch served as the primary deep learning framework, providing tools for building, training, and evaluating neural network models. Its flexibility and GPU support made it ideal for developing the main ANN and ANN estimator models. TenSEAL was employed to handle the HE aspects of our project. Specifically, TenSEAL allowed us to implement the CKKS scheme, perform encrypted computations, and manage encryption parameters, making it possible to execute neural network inference directly on encrypted data. NumPy was used for efficient array manipulation and mathematical operations, assisting in data preprocessing, tensor transformations, and performance evaluation. Scikit-learn provided tools for computing performance metrics, such as accuracy, F1-score, and sensitivity, which were essential for assessing the accuracy and effectiveness of the ANN estimator. Finally, Matplotlib was used for plotting and visualizing results, such as comparisons between the true activation functions and the outputs of the ANN estimators, allowing us to visually analyze the estimator's performance.

For our implementation, we configure the CKKS scheme using the TenSEAL library with the following parameters:

- Polynomial Modulus Degree: A value of 32,768 is chosen for the degree of polynomial modulus. This parameter controls the degree of the polynomial used in the encryption process, directly affecting the security level and computational efficiency of the scheme. A higher value like 32,768 provides a high level of security, which is important to maintain data privacy in cloud environments. Additionally, this degree ensures sufficient capacity to perform the operations required for our neural network without excessive noise growth.
- Coefficient Modulus Bit Sizes: We use the coefficient modulus sizes [60, 50, 50, 50, 50, 50, 60]. These values define the sizes of the coefficient moduli used in the scheme, impacting the precision of the computations and the noise budget available during encrypted operations. The chosen configuration balances the need for precision with the overall noise budget, enabling the execution of a series of arithmetic operations while maintaining decryption accuracy. The higher bit sizes (e.g., 60-bit and multiple 50-bits) ensure that the global scale remains high enough for accurate results.
- Global Scale: The global scale is set to 2^{50}. The global scale determines the precision of the encrypted computations and how values are represented during encryption. A higher global scale of 2^{50} provides more headroom for maintaining precision throughout the computations, reducing the risk of precision loss during encrypted inference. This is particularly crucial for our neural network estimators, as it helps to ensure that the encrypted outputs remain as accurate as possible.
- Galois Keys Generation: The configuration also includes the generation of Galois keys. Galois keys are necessary for enabling rotation operations on encrypted vectors, which are required for efficient computation in neural networks, such as during matrix multiplications or shifting elements within encrypted data. This step is essential for performing complex neural network operations while keeping the data encrypted.

This configuration of the CKKS scheme is carefully optimized to balance security, computational efficiency, and accuracy, making it well-suited for the specific demands of PPML tasks. The parameters were fine-tuned through extensive experimentation, ensuring that the chosen settings provide optimal performance for our encrypted inference process.

B. DATASET AND PREPROCESSING

For our experiments, we utilized the MNIST dataset, a well-established benchmark for machine learning models, particularly in image classification tasks. The MNIST dataset, introduced by LeCun et al. (1998), consists of grayscale images of handwritten decimal digits (0-9). This dataset is extensively used for its accessibility and standardized format, making it a prime choice for assessing machine learning solutions [32].

The MNIST dataset consists of 70,000 images in total, with 60,000 designated for training and 10,000 reserved for testing. Each image measures 28×28 pixels, producing 784 features when flattened, which serve as inputs for our ANN models. In this study, we used all 60,000 training images to train both the main ANN and the ANN estimator, ensuring broad coverage of the dataset and capturing the diverse characteristics of handwritten digits.

For testing, we utilized the standard MNIST test set of 10,000 images but applied a specific selection process during encrypted inference. To manage computation time and ensure a balanced evaluation, we limited the encrypted inference tests to a maximum of 100 samples per class. This subset choice enabled evaluation across all digit classes while managing computational demands, providing reliable information on the accuracy and effectiveness of the ANN estimator in previously unseen data.

Preprocessing of the MNIST dataset involved converting each image into a tensor and applying standard normalization. Specifically, each image was transformed to have a mean of 0.1307 and a standard deviation of 0.3081, following the typical normalization values for MNIST. This normalization step is crucial as it helps stabilize the training process by scaling the input values, allowing the neural networks to converge more efficiently. The images were flattened into 784-dimensional vectors to match the input layer's structure in the main ANN.

C. CASE STUDY: PREDICTING HEART DISEASE

We performed a case study utilising the UCI Heart Disease dataset to show the generalisability and practical applicability of our suggested ANN-based activation function estimators. This experiment investigates the performance of activation estimators in a real-world clinical dataset under plaintext inference conditions, but the main focus of this study has been on assessing them in the context of encrypted inference. By using a tabular medical dataset, we hope to demonstrate that the suggested estimator framework is not just applicable to image data (like MNIST) but can also be effectively applied in privacy-sensitive fields like healthcare, where decision support systems are increasingly using predictive modelling.

The UCI Heart Disease dataset is a widely used benchmark in medical machine learning studies. It contains 303 patient records with 13 input features that include clinical and demographic indicators such as age, sex, chest pain type, resting blood pressure, cholesterol level, and electrocardiographic results [35]. The purpose is to predict the presence of cardiac disease in a patient, which is expressed as a binary classification task. This dataset is especially useful for testing the resilience of machine learning models in healthcare because of its real-world properties, class imbalance, and small sample size.

Although the UCI Heart Disease dataset does not contain direct identifiers such as names or Social Security Numbers (SSNs), it retains quasi-identifiers—age, sex, resting blood pressure, and cholesterol levels—that are widely recognized as potential indirect identifiers in privacy literature. These features, when combined, can be exploited for re-identification attacks through data linkage techniques. For instance, a simple k-anonymity check on the four-attribute set (age, sex, cholesterol, and resting blood pressure) reveals that 10 out of 303 records (3.3%) are unique, underscoring real-world privacy risks associated with de-identified medical records. This mirrors prior findings that (i) up to 28% of individuals in HIPAA-compliant discharge data were re-identified via newspaper linkage [36], and (ii) 99.98% of U.S. residents are unique when just 15 demographic fields are known [37]. Copula-based risk models on comparable clinical datasets further demonstrate that even small sets of quasi-identifiers can pose substantial re-identification danger [38].

It is important to note that our primary aim is not to address direct disclosure risks (i.e., identifier leakage), but rather to evaluate the secure inference capability of our encrypted ANN model, which operates solely over ciphertext—even when input data carries such quasi-identifiers. This approach emphasizes the significance of enabling privacy-preserving inference over encrypted quasi-identifier data, rather than relying exclusively on de-identification or data sanitization.

The UCI Heart Disease dataset underwent preprocessing to ensure good data quality and compliance with the neural network training methodology. To ensure that all records were preserved, missing feature values were replaced with the mean of each column. The features were transformed to a uniform numerical representation, and the class labels were simplified into a binary classification task by assigning a label of 1 to each incidence of heart disease and 0 otherwise. To achieve uniform feature contribution, all features were standardised with zero mean and unit variance. The dataset was then divided into training and testing sets in an 80/20 ratio to ensure balanced class distributions. Finally, the data was organised into batches to facilitate processing during model training and evaluation.

The main ANN designed for this case study consists of a fully connected input layer, a single hidden layer with 512 neurons, and an output layer configured for binary classification. A dropout layer with a rate of 0.3 was incorporated after the activation function to improve generalization. Three different activation functions—Sigmoid, ReLU, and Tanh—were tested in the hidden layer to assess their effect on classification performance. To facilitate encrypted inference, we trained an enhanced ANN-based estimator designed to approximate the chosen activation function using a simple two-layer fully connected architecture. During training, we first trained the main ANN using the cross-entropy loss function with an adaptive optimizer and a learning-rate scheduler over 300 epochs. Subsequently, the estimator was trained in a regression setup to map preactivation values to their corresponding post-activation outputs, effectively learning the activation behavior. For each activation function, the experiment was repeated independently, and results were compared with plaintext inference to evaluate accuracy, loss, and generalization metrics.

To maintain consistency and comparability, the HE configuration used here mirrors that of the earlier experiments.

VI. RESULTS

This section presents the experimental results that validate our primary objectives: to demonstrate that ANN-based

estimators can accurately approximate non-linear activation functions (Sigmoid and Tanh) within homomorphic encrypted inference, while balancing accuracy with computational feasibility. A detailed interpretation of these results and their explicit connection to the study's objectives and broader implications are provided in the Discussion section.

A. PERFORMANCE METRICS

After obtaining the encrypted inference results, they are decrypted and compared against the true values from the test set. To assess the performance of the estimators under study, we utilize the evaluation metrics outlined in Table 2. These metrics provide a comprehensive view of each estimator's accuracy, sensitivity, and overall effectiveness. For further details on these evaluation metrics, please refer to [33].

All results in this work are rounded to four decimal places for consistency and clarity.

TABLE 2. Evaluation metrics and their descriptions.

Metric	Description	Formula
Accuracy	Measures the ratio of instances classified correctly to the total number of instances.	$\text{Accuracy} = \frac{\text{number of correct predictions}}{\text{total number of predictions}}$
MSE (Loss)	Represents the average of the squared differences between the predicted and the actual values. It is commonly used to quantify prediction errors in regression tasks.	$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2$
F1-score	The harmonic mean of precision and recall.	$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$
Sensitivity (Recall)	The fraction of actual positives that the model correctly recognizes as positives.	$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives}+\text{False Negatives}}$
Specificity	The fraction of actual negatives that the model correctly classifies as negatives.	$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives}+\text{False Positives}}$
AUC	The area under the ROC curve, indicating the model's ability to distinguish between classes across various thresholds.	$\text{AUC} = \int \text{ROC Curve}$

Since encryption and decryption times are consistent across experiments, we focus solely on evaluating encrypted inference efficiency. This approach ensures that our analysis accurately reflects the inference process's computational demands within the HE framework.

B. ENCRYPTED INFERENCE PERFORMANCE ON MNIST USING ANN CLASSIFIERS

1) RESULTS USING SIGMOID ACTIVATION FUNCTION ESTIMATORS

Table 3 presents the results of the homomorphic inferences using the Sigmoid activation function and its respective estimators within the HE framework. Figures 8, 9, and 10 display the ROC curves along with the AUC values for the homomorphic inferences conducted with the polynomial, piecewise linear, and ANN-based Sigmoid estimators, respectively.

TABLE 3. Homomorphic inferences using different estimators for sigmoid activation function.

Method	Accuracy	Sensitivity	Specificity	F1-score	Avg Loss	AUC
Polynomial Estimator	0.8540	0.8540	0.9838	0.8538	0.8392	0.9863
Piecewise Linear Approx.	0.1000	0.1000	0.9000	0.0182	2.3130	0.4927
ANN Estimator	0.8710	0.8710	0.9857	0.8703	0.4238	0.9887

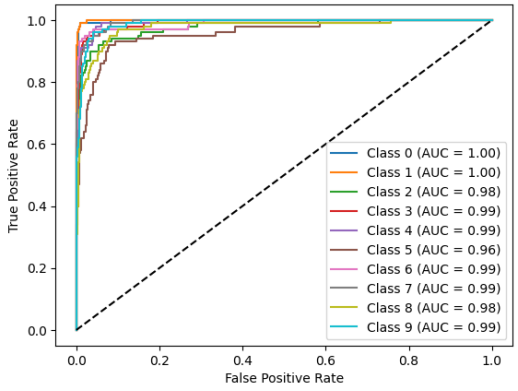

FIGURE 8. ROC curves and AUC values for the homomorphic inference using Sigmoid polynomial estimator.

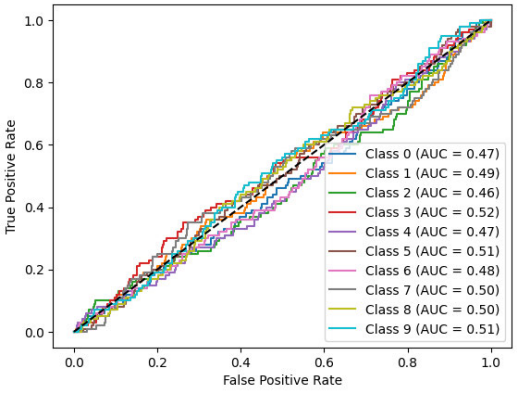

FIGURE 9. ROC curves and AUC values for the homomorphic inference using Sigmoid piecewise linear approximation.

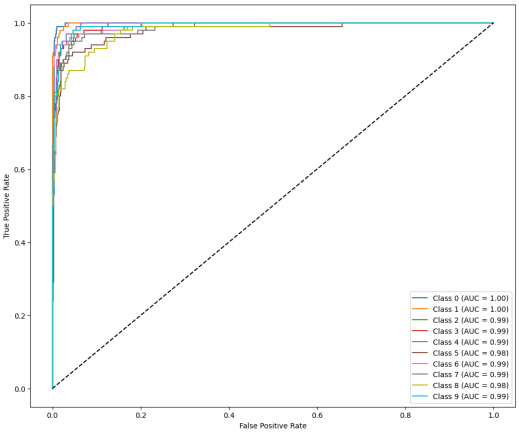

FIGURE 10. ROC curves and AUC values for the homomorphic inference using Sigmoid ANN estimator.

2) RESULTS USING TANH ACTIVATION FUNCTION ESTIMATORS

Table 4 presents the results of the homomorphic inferences using the Tanh activation function and its respective estimators within the HE framework. Figures 11, 12, and 13

display the ROC curves along with the AUC values for the homomorphic inferences conducted with the polynomial, piecewise linear, and ANN-based Sigmoid estimators, respectively.

TABLE 4. Homomorphic inferences using different estimators for tanh activation function.

Method	Accuracy	Sensitivity	Specificity	F1-score	Avg Loss	AUC
Polynomial Estimator	0.4950	0.4950	0.9439	0.4548	11.2686	0.9395
Piecewise Linear Approx.	0.1000	0.1000	0.9000	0.0182	2.3124	0.4938
ANN Estimator	0.8560	0.8560	0.9840	0.8549	0.4748	0.9876

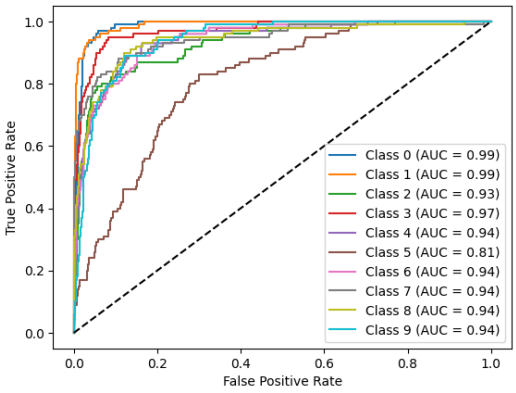

FIGURE 11. ROC curves and AUC values for the homomorphic inference using Tanh polynomial estimator.

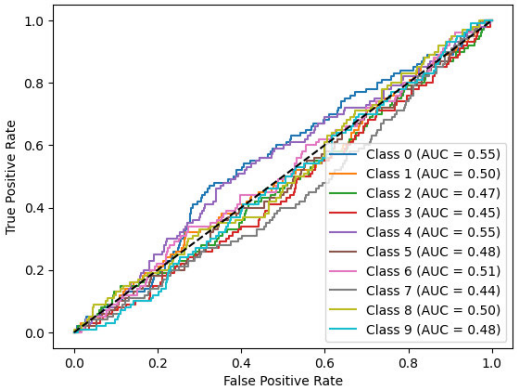

FIGURE 12. ROC curves and AUC values for the homomorphic inference using Tanh ANN estimator.

3) ENCRYPTED INFERENCE TIME

Since the homomorphically encrypted inference was tested on a dataset containing 1,000 points (100 from each class), the average computation time was calculated for each method. The encryption and decryption times were excluded, as they remain consistent across all tests. Our focus was to compare the computational differences among the non-linear estimators for the activation functions. The computational times were grouped in alignment with the experimental

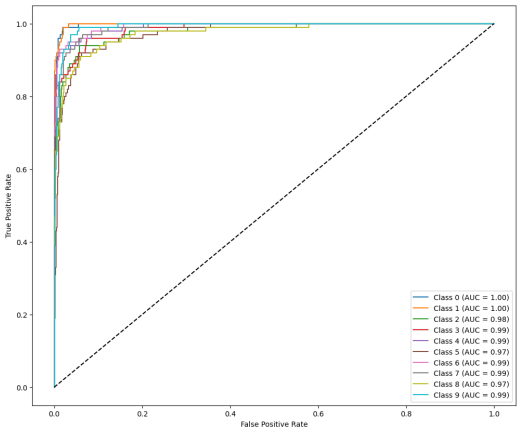

FIGURE 13. ROC curves and AUC values for the homomorphic inference using Tanh ANN estimator.

setup outlined in Figure 7. Implementation time details are provided in Table 5.

TABLE 5. Average implementation time for the encrypted inferences based on the type of estimators to Sigmoid and Tanh.

Method	Sigmoid [Seconds]	Tanh [Seconds]
Polynomial Estimator	1.9145	1.9388
Piecewise Linear Approximation	2.8261	2.9069
ANN Estimator	5.6938	5.6947

C. COMPARATIVE ANALYSIS

A substantial body of previous research (refer to Section II) utilizes CNNs as the backbone for performing inference on homomorphically encrypted data. To enable a fair and comprehensive comparison, we extend our evaluation by incorporating the proposed ANN-based estimator into a CNN framework. This subsection presents a comparative analysis covering CNN architecture, encryption security level (in bits), classification accuracy, and implementation time. By comparing our method with existing approaches based on polynomial or piecewise approximations, we demonstrate that our ANN estimator maintains its effectiveness and flexibility even within deeper, more complex networks. Additionally, this analysis highlights the trade-offs between accuracy, security, and computational efficiency. A summary of this comparison is provided in Table 6.

The security level for CKKS and BFV HE schemes is primarily determined by two critical parameters: the polynomial modulus degree (N) and the coefficient modulus bit sizes. As outlined by Furka et al. [39], the polynomial modulus degree N significantly influences both security and computational complexity, where larger N values generally provide higher security but increased computational demands. The total sum of bits used for the coefficient modulus (denoted as Q) is then compared against standardized security bounds provided by the Homomorphic Encryption Security Standard.

Specifically, the maximum permissible coefficient modulus bit-length ($Q_{\max}$) for achieving a target security level, such as 128 bits, is predefined for various polynomial degrees—for example, for $N = 8192$, the maximum allowed bit-size sum for 128-bit security is 218 bits. Hence, ensuring the total bits in the coefficient modulus are below this threshold guarantees the desired level of security in bits [39].

D. CASE STUDY RESULTS: HEART DISEASE PREDICTION

The performance results of the ANN classifier and activation function estimators on the UCI Heart Disease dataset are presented in Table 7. The table compares three activation functions—Sigmoid, Tanh, and ReLU—under both plaintext and ciphertext inference.

VII. DISCUSSION

This section presents the evaluation results of the proposed ANN-based estimators for Sigmoid and Tanh activation functions within PPML frameworks. As shown in Tables 3 and 4, the ANN-based estimators demonstrated robust performance in approximating non-linear activation functions, particularly under HE. By examining the performance of Sigmoid and Tanh estimators across polynomial, piecewise linear, and ANN-based methods, it was evident that ANN estimators achieved the highest levels of accuracy and sensitivity. For instance, the ANN estimators recorded an AUC of 0.9887 for Sigmoid and 0.9876 for Tanh, outperforming both polynomial and piecewise linear approximations. These findings suggest that ANN-based approaches can more accurately capture the complexities of non-linear activation functions, a critical requirement for effective encrypted inference for machine learning.

One key advantage of ANN-based estimators is their ability to approximate complex non-linear behaviors without the need for separate approximation techniques for each activation function. This flexibility, grounded in the Universal Approximation Theorem, allows ANN estimators to generalize across different types of non-linear functions by merely adjusting the training dataset rather than altering the model architecture itself. Compared to polynomial approximations, which may require high degrees to achieve similar accuracy, and piecewise linear methods, which struggle with precise curve fitting, ANN estimators offer a balanced solution for maintaining both accuracy and computational feasibility. This adaptability also makes ANN-based estimators particularly promising for applications in MLaaS, where different types of activation functions may be required across various encrypted models.

The piecewise linear approximation method for estimating Sigmoid and Tanh (Tables 3 and 4) yielded significantly lower accuracy compared with other methods. This decrease in performance can be attributed to the inherent limitations of the piecewise linear approach in capturing the non-linear characteristics of complex activation functions like Sigmoid and Tanh. While the piecewise linear method approximates the activation function by dividing the input space into intervals and applying linear transformations within each interval, this approach fails to account for the smooth and continuous curvature present in these activation functions. As a result, the model loses essential non-linear information that is crucial for accurate predictions, particularly in datasets where the relationships between features are highly non-linear.

Despite their accuracy, the computational demands of ANN-based estimators are notably higher than those of polynomial and piecewise linear methods. As mentioned in Table 5, the encrypted inference time for the ANN estimator reached an average of 5.69 seconds for Sigmoid and 5.69 seconds for Tanh, compared to just 1.91 seconds for the polynomial approach. However, this increase in computation time is offset by the improvement in model performance, which may justify the trade-off in scenarios where accuracy takes priority over speed. The added computation time may be a worthwhile investment, especially in sensitive sectors like healthcare and finance, where precise and privacy-preserving inferences are crucial.

If we see the comparative analysis in Table 6, it is evident that while some previous works, such as Lin et al. [19], achieve a high accuracy of 0.9870 using two-stage polynomial activations, and Khan and Michalas [21] secure 0.9850 accuracy with an intricate CNN framework that incurs over 150 seconds of inference time, these approaches come with increased complexity and computational overhead. Similarly, Nguyen et al. [22] and Xiong et al. [25] deliver competitive performance with fast inference times, yet often depend on complex network designs and advanced noise-management techniques—and some even operate at a lower 80-bit security level. In contrast, our approach leverages a lightweight, single-layer ANN-based estimator that not only achieves a commendable accuracy of 0.9770 but also registers an outstanding AUC of 0.9997. This high AUC value is particularly significant as it demonstrates the model's superior ability to distinguish between classes under encrypted conditions, which is critical in high-stakes applications where precision is essential. Moreover, our solution maintains robust 128-bit security, comparable to most of the high-security benchmarks in the literature, while delivering an encrypted inference time of approximately 21.87 seconds. This balanced design underscores the practical benefits of our method—combining reduced complexity, strong security, and exceptional discriminative performance—making it especially suitable for real-world PPML applications.

Related to the case study results, the findings in Table 7 confirm that HE inference performs closely to plaintext inference across all tested activation functions. Although HE inference introduces a noticeable increase in implementation time due to encryption overhead, the model's classification performance remains consistent. Accuracy, sensitivity, specificity, and AUC values show minimal degradation, indicating that the proposed ANN estimator can effectively approximate activation functions in the encrypted domain.

TABLE 6. Summary of CNN architectures and performance metrics in homomorphic encrypted inference for MNIST dataset.

Ref.	Model Design	Security Level	Performance	Implementation Time
Zhang et al. (2024) [19]	Conv Layer: (Kernel=5×5, Stride=2, Padding=2), Channels: 1→5, polynomial activation (degree=2); Fully Connected: 980→100, polynomial activation (degree=2); Fully Connected: 100→10	128-bit (Microsoft SEAL 3.7)	Accuracy: 0.9870	6.7s
Shi and Zhao (2023) [20]	Input Layer: 784 neurons; 6 Hidden Layers: 512→256→128→64→32→16 (ReLU activation); Output Layer: 10 neurons (SoftMax activation)	Efficient integer vector-based HE (specific security bits not explicitly stated)	Accuracy: 0.8639 – 0.8879	Not explicitly stated, but reported 58× faster than traditional CNN
Khan and Michalas (2023) [21]	Conv: Input 28×28, kernel 5×5, stride=1, Channels: 1→5, ReLU; Mean Pooling: 2×2, stride=1; Conv: kernel 5×5, stride=1, Channels: 5→10, ReLU; Mean Pooling: 2×2, stride=1; FC: 490→128, ReLU; FC: 128→10, Softmax	Fan-Vercauteren Somewhat HE (Security bits not explicitly stated)	Accuracy: 0.9850	153.3172s
Nguyen et al. (2023) [22]	Conv: Kernel=5×5, stride=2, padding=2, Channels: 1→5, ReLU; FC: 980→100, ReLU; FC: 100→10	HE-based (Security bits not explicitly stated)	Accuracy: 0.9831 for simple architecture (HeFUNs) and 0.9916 for more complex model (HeFUNl)	1.374s (HeFUNs), 1.501s (HeFUNl)
Xiong et al. (2020) [25]	LeNet-style: Conv → Pool → FC (for MNIST/CIFAR-10)	BFV scheme (based on RLWE), 80-bit	Accuracy: 0.9600 – 0.9700 (Encrypted Ensemble)	< 7 min for 50,000 images; Layer-wise: Conv ∼2.9s, Dense ∼6.8s
Hesamifard et al. (2018) [29]	Fully connected neural network with 1–5 hidden layers, using low-degree polynomial approximations (typically Chebyshev or custom) to replace standard activation functions, implemented for both training and inference phases on encrypted data via leveled HE.	80-bit security	The highest accuracy was scored using Polynomial approximation of Sigmoid: 99.15%	∼320s
This work	Conv (3×3, 32→64) → MaxPool ×2 → FC (3136→64→10), Sigmoid ANN Estimator	CKKS (poly_mod_degree= 32768, 128-bit equiv.)	Accuracy: 0.9770, F1: 0.9770, AUC: 0.9997, Spec: 0.9974	CNN Train: 162.61s Estimator Train: 137.57s Encrypted Inference: 21.8737s

TABLE 7. Performance comparison of activation function estimators on the UCI heart disease dataset.

Activation Function	Inference Type	Time (s)	Accuracy	Sensitivity	Specificity	F1-score	Loss	AUC
Sigmoid	Plaintext	0.0200	0.8525	0.8582	0.8582	0.8525	0.3272	0.9502
	Ciphertext	5.7030	0.8525	0.8582	0.8582	0.8525	0.3500	0.9459
Tanh	Plaintext	0.0200	0.8033	0.8155	0.8155	0.8019	0.4238	0.9437
	Ciphertext	5.7164	0.7705	0.7798	0.7798	0.7699	0.7048	0.9123
ReLU	Plaintext	0.0200	0.8033	0.8101	0.8101	0.8032	0.7537	0.9297
	Ciphertext	5.7208	0.8525	0.8582	0.8582	0.8525	1.1677	0.9343

Among the activation functions, ReLU and Sigmoid achieved the highest accuracy during HE inference, each reaching 0.8525, matching their plaintext counterparts. This demonstrates the estimator's ability to accurately replicate activation function behavior in encrypted inference settings. While Tanh exhibited slightly lower performance under encryption, it still produced reasonable results, affirming the robustness of the overall approach for medical data analysis in secure environments.

The performance of our ANN models on the UCI Heart Disease dataset aligns with previous studies using the same data. For example, Mohan et al. achieved 0.8901 accuracy with a hybrid ANN model [40], Srinivasan et al. reported 0.9478 using a Naïve Bayes network [41], and Narasimhan and Victor observed ANN accuracy of about 0.8361 [42]. These results support the validity of our approach, demonstrating that our models maintain competitive accuracy even under HE constraints.

Despite the promising results, the proposed approach has certain limitations that must be addressed in future work. First, although the piecewise linear method is computationally efficient, it demonstrated significantly lower accuracy and sensitivity. This restricts its practical use in HE-based scenarios, where maintaining high precision is crucial. Second, noise accumulation in HE remains a critical bottleneck, especially when employing ANN estimators that involve multiple encrypted operations. As the complexity of the model or the dimensionality of the data increases, so does the risk of accuracy degradation due to noise growth. These limitations highlight the need to further optimize both the neural network architecture and encryption parameters. Techniques such as ciphertext relinearization, bootstrapping,

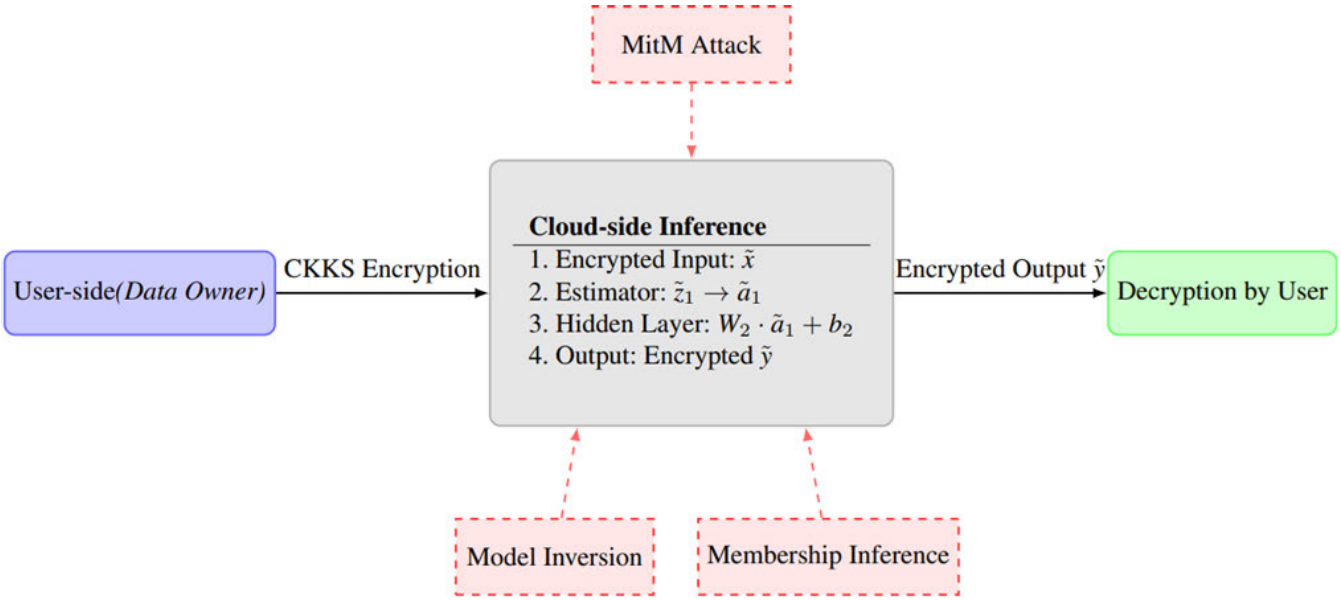

FIGURE 14. Threat model of the proposed encrypted inference system. The system operates entirely on ciphertexts using CKKS-based HE, thereby minimizing exposure to privacy attacks such as MitM, model inversion, and membership inference. Dashed red arrows represent potential attack vectors under the honest-but-curious cloud adversary model. Notation: $\tilde{x}$ - Encrypted input;$\tilde{z}_1$ - Pre-activation;$\tilde{a}_1$ - Post-activation output;$W_2 \cdot \tilde{a}_1 + b_2$ - Encrypted hidden layer;$\tilde{y}$ - Encrypted prediction.

or lightweight ANN designs may help reduce overhead and improve scalability for real-world encrypted inference tasks.

To evaluate the privacy-preserving capabilities of our design, we conducted a threat analysis based on a ciphertext-only model under the honest-but-curious adversary assumption. While the current study does not include empirical attack simulations, the system architecture leverages semantically secure CKKS-based HE to mitigate known threats in MLaaS settings.

As illustrated in Figure 14, the entire inference process operates on ciphertexts, ensuring that raw input features ($\tilde{x}$), intermediate activations ($\tilde{z}_1$, $\tilde{a}_1$), and model parameters (W_2, b_2) are never exposed to the cloud environment. This ciphertext-only design inherently mitigates common attack surfaces associated with MLaaS deployments.

The diagram also highlights potential adversarial vectors, denoted with dashed red arrows within an honest-but-curious threat model, where the cloud executes the protocol as expected but may attempt to extract sensitive information. These include:

- MitM: Interception of encrypted communication between the data owner and the cloud.
- Model inversion: Attempts to reconstruct sensitive input data from model outputs.
- Membership inference: Attempts to determine whether a specific data point was used during training.

Because the protocol operates solely on ciphertexts and avoids any form of decryption in untrusted environments, the system provides strong theoretical guarantees of confidentiality. An informal attack surface analysis confirmed that no inference-time leakage points exist in the design.

Beyond the architectural guarantees, the privacy relevance of the datasets used—including the UCI Heart Disease dataset, which served as a representative case study—strengthens the threat model. The UCI Heart Disease dataset, in particular, includes quasi-identifiers such as age, sex, cholesterol, and blood pressure—attributes known to enable re-identification through linkage with external datasets. Even in the absence of explicit identifiers, prior studies have shown that such features can compromise anonymity. Therefore, performing inference entirely over encrypted quasi-identifiers is not just a theoretical demonstration of HE capabilities, but a practical defense mechanism against real-world adversarial risks, including statistical reconstruction and auxiliary-data correlation.

Nevertheless, the lack of empirical adversarial testing remains a limitation. As part of future work, we aim to simulate targeted attacks such as adaptive model extraction, ciphertext correlation, and membership inference, in order to empirically validate the resilience of our encrypted inference framework. This will strengthen both the practical security and the credibility of the proposed privacy-preserving solution.

VIII. CONCLUSION AND FUTURE WORKS

This study presented a novel approach for approximating non-linear activation functions, specifically Sigmoid and Tanh, using ANN-based estimators within a homomorphic encryption (HE) framework. By addressing the computational challenges associated with encrypted inference, the proposed method maintains model accuracy and computational efficiency. Experimental results demonstrated that

ANN-based estimators consistently outperform traditional polynomial and piecewise linear methods in terms of accuracy, sensitivity, and AUC, highlighting their practical viability for secure machine learning inference in cloud-based environments. These findings confirm that ANN-based estimators effectively approximate complex non-linear behaviors, offering a promising solution for enhancing encrypted inference without compromising precision or feasibility.

Despite these demonstrated advantages, the computational cost associated with ANN estimators highlights an area for optimization. Compared to polynomial approximations, which are computationally efficient but less accurate, ANN-based estimators require more processing time, potentially limiting their use in latency-sensitive applications. Nevertheless, the accuracy benefits of ANN estimators, particularly in critical fields like healthcare and finance, underscore their importance in settings where secure inference and model performance are essential.

To further validate the approach, we conducted a case study on the UCI Heart Disease dataset. The results reaffirmed the effectiveness of the proposed ANN-based activation function estimators in maintaining high classification accuracy under HE. Sigmoid and ReLU activation functions yielded the best results in encrypted inference, closely matching their plaintext counterparts with minimal additional computation time. These outcomes demonstrate the feasibility of achieving accurate and secure machine learning inference. Moreover, our ANN models performed comparably to existing studies using the same dataset, further validating the approach.

Additionally, we extended our evaluation to a CNN architecture, showing that the proposed estimator generalizes well to more complex networks. The CNN results under encrypted inference remained competitive with state-of-the-art solutions, achieving a balanced trade-off between accuracy, efficiency, and secure inference capability. These findings support the practicality of integrating the proposed estimator across a wide range of secure machine learning applications.

Overall, these results highlight the advantages of using ANN-based estimators for non-linear activations under HE, achieving higher accuracy than polynomial or piecewise linear methods. By closely approximating the true Sigmoid and Tanh functions, our estimator maintains the expressive power of neural networks and enables precise encrypted inference. The single-layer design also simplifies integration into existing workflows, further emphasizing the feasibility of secure inference solutions in cloud environments demanding robust data protection and accuracy.

Looking ahead, future work may include the following directions:

- Optimizing the ANN architecture for HE, potentially by exploring more efficient network structures that reduce computation time while preserving accuracy,
- Applying computational efficiency techniques, such as model pruning or quantization, to make ANN-based estimators more feasible without significant loss in performance,
- Extending the estimator to additional activation functions, including ReLU and Softmax, which are widely used in deep learning models,
- Evaluating the system under adversarial conditions, such as MitM interception, model inversion, and membership inference attacks. We aim to construct a comprehensive threat model and simulate such attacks to empirically assess inference-time leakage and data reconstruction threats, further reinforcing the robustness of the encrypted inference framework,
- Integrating the estimator with complementary privacy-preserving techniques, such as federated learning or TEEs, to enable secure, distributed learning scenarios.

By combining the strengths of HE with these complementary methods, future research may create more scalable, versatile, and practical PPML solutions.

REFERENCES

[1] A. Wood, M. Altman, A. Bembenek, M. Bun, M. Gaboardi, J. Honaker, K. Nissim, D. R. O'Brien, T. Steinke, and S. Vadhan, "Differential privacy: A primer for a non-technical audience," *Vanderbilt J. Entertainment Technol. Law*, vol. 21, no. 1, pp. 209–276, Jan. 2018.

[2] F. Zhu, F. Hu, Y. Zhao, B. Chen, and X. Tan, "A secure and fair federated learning framework based on consensus incentive mechanism," *Mathematics*, vol. 12, no. 19, p. 3068, Sep. 2024.

[3] J. Park and H. Lim, "Privacy-preserving federated learning using homomorphic encryption," *Appl. Sci.*, vol. 12, no. 2, p. 734, Jan. 2022.

[4] M. R. Abou Harb and B. Celiktas, "Efficient estimation of sigmoid and tanh activation functions for homomorphically encrypted data using artificial neural networks," *TechRxiv*, Jan. 2025, doi: 10.36227/TECHRXIV.173579549.98951880/V1.

[5] T. Aremu and K. Nandakumar, "PolyKervNets: Activation-free neural networks for efficient private inference," in *Proc. IEEE Conf. Secure Trustworthy Mach. Learn. (SaTML)*, Feb. 2023, pp. 593–604.

[6] E. Sarkar, E. Chielle, G. Gürsoy, O. Mazonka, M. Gerstein, and M. Maniatakos, "Fast and scalable private genotype imputation using machine learning and partially homomorphic encryption," *IEEE Access*, vol. 9, pp. 93097–93110, 2021.

[7] K. T'Jonck, C. R. Kancharla, B. Pang, H. Hallez, and J. Boydens, "Privacy preserving classification via machine learning model inference on homomorphic encrypted medical data," in *Proc. 31st Int. Sci. Conf. Electron. (ET)*, vol. 1, Sep. 2022, pp. 1–6.

[8] N. Almutairi, F. Coenen, and K. Dures, "PPNNBP: A third party privacy-preserving neural network with back-propagation learning," *IEEE Access*, vol. 11, pp. 31657–31675, 2023.

[9] A. Marcel, D. Miu, and A. Rancea, "Privacy preserving neural network models based homomorphic encryption: A case study for diabetes prediction," in *Proc. IEEE 19th Int. Conf. Intell. Comput. Commun. Process. (ICCP)*, Oct. 2023, pp. 269–275.

[10] D. Natarajan, A. Loveless, W. Dai, and R. Dreslinski, "Chex-mix: Combining homomorphic encryption with trusted execution environments for oblivious inference in the cloud," in *Proc. IEEE 8th Eur. Symp. Secur. Privacy*, Jul. 2023, pp. 73–91.

[11] M. Baryalai, J. Jang-Jaccard, and D. Liu, "Towards privacy-preserving classification in neural networks," in *Proc. 14th Annu. Conf. Privacy*, Dec. 2016, pp. 392–399.

[12] Y. Yao, Z. Zhao, X. Chang, J. Mišić, V. B. Mišić, and J. Wang, "A novel privacy-preserving neural network computing approach for e-health information system," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2021, pp. 1–6.

[13] D. Lei, C. Hu, and J. Dong, "NPNNL: A non-interactive privacy-preserving neural network learning scheme," in *Proc. IEEE Int. Conf. Metaverse Comput.*, vol. 8, Jun. 2023, pp. 129–133.

[14] O. L. Usman, R. C. Muniyandi, K. Omar, M. Mohamad, A. A. Owoade, and M. A. Kareem, "HoRNS-CNN model: An energy-efficient fully homomorphic residue number system convolutional neural network model for privacy-preserving classification of dyslexia neural-biomarkers," *Brain Informat.*, vol. 12, no. 1, p. 11, Dec. 2025.
[15] V. D. Allavarpu, V. S. Naresh, and A. K. Mohan, "Neural network-driven privacy-preserving credit risk analysis: A homomorphic encryption approach," *Contemp. Math.*, vol. 6, pp. 1051–1075, Feb. 2025.
[16] J.-H. Lee, "Efficient polynomial approximations for non-arithmetic functions in inference on fully homomorphically encrypted data," Ph.D. dissertation, Seoul Nat. Univ., Seoul, South Korea, 2024.
[17] J. Xiong, J. Chen, J. Lin, D. Jiao, and H. Liu, "Enhancing privacy-preserving machine learning with self-learnable activation functions in fully homomorphic encryption," *J. Inf. Secur. Appl.*, vol. 86, Nov. 2024, Art. no. 103887.
[18] C. Song and X. Shi, "ReActHE: A homomorphic encryption friendly deep neural network for privacy-preserving biomedical prediction," *Smart Health*, vol. 32, Jun. 2024, Art. no. 100469.
[19] Y. Lin, T. Zhang, Y. Mao, and S. Zhong, "CrossNet: A low-latency MLaaS framework for privacy-preserving neural network inference on resource-limited devices," *IEEE Trans. Dependable Secure Comput.*, vol. 22, no. 2, pp. 1265–1280, Mar. 2025.
[20] J. Shi and X. Zhao, "Anti-leakage method of network sensitive information data based on homomorphic encryption," *J. Intell. Syst.*, vol. 32, no. 1, Mar. 2023, Art. no. 20220281.
[21] T. Khan and A. Michalas, "Learning in the dark: Privacy-preserving machine learning using function approximation," in *Proc. IEEE 22nd Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Nov. 2023, pp. 62–71.
[22] D. T. K. Nguyen, D. H. Duong, W. Susilo, Y.-W. Chow, and T. A. Ta, "HeFUN: Homomorphic encryption for unconstrained secure neural network inference," *Future Internet*, vol. 15, no. 12, p. 407, Dec. 2023.
[23] J.-W. Lee, H. Kang, Y. Lee, W. Choi, J. Eom, M. Deryabin, E. Lee, J. Lee, D. Yoo, Y.-S. Kim, and J.-S. No, "Privacy-preserving machine learning with fully homomorphic encryption for deep neural network," *IEEE Access*, vol. 10, pp. 30039–30054, 2022.
[24] S. Hong, J. H. Park, W. Cho, H. Choe, and J. H. Cheon, "Secure tumor classification by shallow neural network using homomorphic encryption," *BMC Genomics*, vol. 23, no. 1, p. 284, Dec. 2022.
[25] A. Xiong, M. Nguyen, A. So, and T. Chen, "Privacy preserving inference with convolutional neural network ensemble," in *Proc. IEEE 39th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Nov. 2020, pp. 1–8.
[26] A. Vizitiu, C. I. Niţă, A. Puiu, C. Suciu, and L. M. Itu, "Applying deep neural networks over homomorphic encrypted medical data," *Comput. Math. Methods Med.*, vol. 2020, pp. 1–26, Apr. 2020.
[27] M. Izabachène, R. Sirdey, and M. Zuber, "Practical fully homomorphic encryption for fully masked neural networks," in *Proc. Int. Conf. Cryptol. Netw. Secur.*, Jan. 2019, pp. 24–36.
[28] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service," *Proc. Privacy Enhancing Technol.*, vol. 2018, no. 3, pp. 123–142, Apr. 2018.
[29] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–376, Jan. 1989.
[30] J. H. Cheon, A. Costache, R. C. Moreno, W. Dai, N. Gama, M. Georgieva, S. Halevi, M. Kim, S. Kim, K. Laine, Y. Polyakov, and Y. Song, *Protecting Privacy through Homomorphic Encryption*, vol. 1, Jan. 2021, pp. 3–28.
[31] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Stat.*, Mar. 2010, pp. 249–256.
[32] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[33] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[34] D. M. W. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," 2020, *arXiv:2010.16061*.
[35] A. Janosi, W. Steinbrunn, M. Pfisterer, and R. Detrano, 1989, "Heart disease [dataset]," *UCI Machine Learning Repository*. [Online]. Available: https://doi.org/10.24432/C52P4X
[36] L. Sweeney, J. S. Yoo, L. Perovich, K. E. Boronow, P. Brown, and J. G. Brody, "Re-identification risks in HIPAA safe harbor data: A study of data from one environmental health study," *Technol. Sci.*, vol. 2017, Jan. 2017, Art. no. 2017082801.
[37] L. Rocher, J. M. Hendrickx, and Y.-A. de Montjoye, "Estimating the success of re-identifications in incomplete datasets using generative models," *Nature Commun.*, vol. 10, no. 1, p. 3069, Jul. 2019.
[38] Y. Jiang, L. Mosquera, B. Jiang, L. Kong, and K. E. Emam, "Measuring re-identification risk using a synthetic estimator to enable data sharing," *PLoS ONE*, vol. 17, no. 6, Jun. 2022, Art. no. e0269097.
[39] M. Furka, M. Kalúz, M. Fikar, and M. Klaučo, "Guidelines for secure process control: Harnessing the power of homomorphic encryption and state feedback control," *IEEE Access*, vol. 11, pp. 110328–110341, 2023.
[40] S. Mohan, C. Thirumalai, and G. Srivastava, "Effective heart disease prediction using hybrid machine learning techniques," *IEEE Access*, vol. 7, pp. 81542–81554, 2019.
[41] S. Srinivasan, S. Gunasekaran, S. K. Mathivanan, P. Jayagopal, and G. T. Dalu, "An active learning machine technique based prediction of cardiovascular heart disease from UCI-repository database," *Sci. Rep.*, vol. 13, no. 1, Aug. 2023, Art. no. 13588.
[42] G. Narasimhan and A. Victor, "A hybrid approach with metaheuristic optimization and random forest in improving heart disease prediction," *Sci. Rep.*, vol. 15, no. 1, Mar. 2025, Art. no. 10971.

MHD RAJA ABOU HARB received the B.S. degree in computer technology engineering from Damascus University, in 2014, and the M.S. degree in cybersecurity from Uskudar University, in 2021. He is currently pursuing the Ph.D. degree in computer engineering with FMV Isik University. His current research interests include cybersecurity, privacy preserving machine learning, homomorphic encryption algorithms, and machine learning.

BARIS CELIKTAS received the B.S. degree in systems engineering from National Defense University, in 2008, the M.S. degree in international relations from Karadeniz Technical University, in 2016, the M.S. degree in applied informatics from Istanbul Technical University, in 2018, and the Ph.D. degree in cybersecurity engineering and cryptography from the Institute of Informatics, Istanbul Technical University, in 2022. He is currently an Assistant Professor with the Computer Engineering Department and the Director of the Cybersecurity Graduate Program, Isik University. Besides, he is a Cybersecurity Consultant and an Architect, specializing in enterprise cybersecurity and cryptography solutions, cloud security, risk management, and governance. He holds numerous industry-recognized certifications, including CISSP, CCSP, CISM, CISA, CRISC, SSCP, CCNP, Sec+, CySA+, C|EH, and ISO 27001, 22301, 20000, 27701, and 42001, as a Lead Auditor and a Lead Implementer, along with GDPR DPO and NIST Cybersecurity Consultant. His research interests include cybersecurity, network security, cloud computing, cryptography, malware analysis, risk management, and security applications.