SentinelTouch: A Lightweight Privacy-Preserving Biometric-Fingerprinting Authentication and Identification System Based on Neural Networks and Homomorphic Encryption

Nges Brian Njungle
STAM Center, Arizona State University
nnjungle@asu.edu

Eric Jahns
STAM Center, Arizona State University
jjahns@asu.edu

Mishel Jyothis Paul
STAM Center, Arizona State University
mpaul16@asu.edu

Michel A. Kinsy
STAM Center, Arizona State University
mkinsy@asu.edu

Abstract

Biometric fingerprint authentication and identification systems are increasingly deployed, yet widespread adoption in cloud and server-based platforms remains hindered by privacy and security concerns. Unlike passwords, compromised fingerprints are immutable, making their secure storage and computation paramount. Homomorphic Encryption (HE) offers strong privacy guarantees for fingerprint data processing by enabling computation directly on encrypted data. However, the high dimensionality of fingerprint images and the complexity of the neural networks needed for accurate recognition creates significant bottlenecks, which hinder the practical deployment of HE in this domain.

We introduce SentinelTouch, an open-source framework for privacy-preserving fingerprint authentication and identification that delivers both efficiency and accuracy in HE environments. Our key insight is a twofold optimization: (1) a preprocessing pipeline that reduces fingerprint image dimensions to as low as 28×28 while preserving most of its discriminative details, and (2) the design of a lightweight, HE-friendly neural network that generalizes effectively on this compact data. We evaluate two deployment pipelines: (1) a *full-privacy* pipeline, where encrypted images are processed entirely under HE settings, achieving user identification in a one-to-many setting in just 16 seconds. (2) A *hybrid* pipeline, where only encrypted embeddings are processed under HE settings, achieving one-to-many user identification in 284 milliseconds. Our results show a 10× and 2.5× speedup over the current state-of-the-art results in both pipelines, respectively. Across the SOKOTO and PolyU datasets, SentinelTouch achieves Rank-1 accuracies within ±0.1% of the leading encrypted systems. This work demonstrates the practicality of end-to-end privacy-preserving fingerprint identification and authentication systems, offering HE security guarantees and utilizing neural networks, without compromising accuracy.

Keywords

Privacy-preserving, Biometric Fingerprinting System, Homomorphic Encryption, Neural Networks

Proceedings on Privacy Enhancing Technologies 2026(2), 143–156

https://doi.org/10.56553/popets-2026-0041

1 Introduction

As technology advances, various approaches to authentication and identification in digital systems have emerged. Traditional password-based methods remain the most widely used; however, they suffer from inherent limitations, including the need for users to remember specific pass-strings and vulnerabilities arising from passwords being forgotten, shared, or observed [13]. To address these shortcomings, biometric authentication and identification systems have been developed, leveraging physiological and behavioral traits such as fingerprints, facial features, iris patterns, and voice to authenticate and identify users [4]. These systems have been pervasively deployed across multiple domains and industries, including consumer electronics (e.g., smartphones and laptops), enterprise applications (e.g., biometric login and payment authorization), law enforcement, finance, and healthcare services [27, 35].

Biometric fingerprint authentication is widely regarded as one of the most reliable methods for user identification due to its distinctiveness, permanence, and resistance to forgery. In most authentication and identification systems, they offer an optimal balance between accuracy and user convenience [40]. When properly collected and processed, fingerprint templates are unique and inherently non-fungible, making them highly effective for secure and accurate user identity verification. Unlike password-based systems, these systems remove the need to remember or protect credentials, thereby reducing the risks of compromise. Furthermore, any device equipped with a touch-sensitive interface, such as a smartphone screen, laptop touchpad, or even a door panel, can be readily adapted into a dependable fingerprint scanner [15].

However, as the adoption of biometric fingerprint systems continues to expand, concerns regarding data privacy, security breaches, and the potential misuse of personal information have intensified. This is particularly critical for biometric fingerprint data because it is permanent and cannot be altered. Unlike passwords, compromised fingerprint templates cannot be revoked or replaced. Consequently, an attacker who obtains such data may be able to exploit it indefinitely and across multiple systems. In fact, the adoption of fingerprint-based authentication and identification in cloud platforms and web applications has remained very limited. This is primarily due to serious security and privacy concerns surrounding the storage and processing of biometric data [12]. A well-known demonstration of this issue is the 2016 US Office of Personnel Management data breach, where over 5.6 million individuals' fingerprints were

stolen. This incident highlights the critical need for secure storage, processing, and management of fingerprint data [19].

Homomorphic Encryption (HE) has emerged as the most promising approach for the secure storage and processing of biometric fingerprint data in server and cloud environments [21]. This is because HE enables computations to be performed directly on encrypted data without decryption [18]. It ensures that fingerprint data remains encrypted at rest, in communication, and during computation. Thus, it provides end-to-end privacy and security across the entire biometric fingerprint data lifecycle. However, the significant computational overhead associated with HE operations often results in slow fingerprint matching and search times, posing a major challenge to the practical deployment of such systems.

Numerous existing works have employed HE in the design and development of fingerprint authentication and identification systems. We categorize these works into two distinct classes. The first class includes works such as [21, 41]. These works extract fingerprint data points such as minutiae and ridges to create fingerprint templates. These templates are encrypted, stored, and later matched in the encrypted domain against newly extracted and encrypted templates. Works in this class generally have very simple designs as well as offer a full-privacy pipeline for authentication and identification of users. On the downside, these works suffer from significant performance and memory overheads. These overheads arise from complex fingerprint template representations, which are computationally demanding and easily affected by perturbations during the encrypted matching and identification processes.

The second class of works utilizes deep neural networks for fingerprint image processing prior to matching. These neural networks reduce the storage requirements by transforming and extracting compact embeddings from fingerprint data. These embeddings are significantly smaller than the templates used in the first class, thus reducing the storage requirements and complexity of matching. The integration of neural networks further enhances identification accuracy, as they leverage sophisticated and robust machine learning–based feature extraction techniques. +The size of fingerprint images typically requires large neural network models with millions of parameters for effective generalization. However, such models are computationally expensive to evaluate in the encrypted domain. For instance, the current state-of-the-art encrypted inference on a ResNet-20 model reports a best runtime of 260 seconds for an input of size 3×32×32 [34], a latency that is clearly impractical for real-time authentication scenarios. Moreover, the referenced input size for this model is much smaller than that used in typical fingerprint authentication and identification systems. Scaling this input into fingerprint image dimensions would require substantially larger HE parameters and model configurations, leading to a proportional increase in computational cost and further degrading performance.

In 2019, DeepPrint [16] introduced an efficient hybrid pipeline for the efficient use of HE in privacy-preserving fingerprint systems. In this pipeline, the feature extraction layers of the neural network are processed on the client-side and in plain text to generate user embeddings. The fingerprint matching task is then moved to the server-side, where the encrypted user's embeddings are compared against stored encrypted users' embeddings. DeepPrint was able to achieve privacy-preserving one-to-many user identification using HE in 3.4 seconds. Works such as [11, 12, 38] have adopted this hybrid pipeline, creating very efficient privacy-preserving biometric fingerprint identification systems that work in real-time. While this approach leads to high-performing, privacy-preserving fingerprint identification systems, it also presents some limitations. Firstly, their practical applications are limited to settings where the neural network models can be deployed on the local device. This setting is not always feasible in the real world, as most Machine Learning as a Service (MLaaS) application scenarios only depend on models deployed remotely. Secondly, it assumes that the client devices have enough computational resources to run the feature extraction layers of the neural network in real-time. While this assumption holds in many scenarios, a large portion of biometric fingerprinting clients rely on the Internet of Things and Edge Devices, which often have constrained computational resources [37]. In resource-constrained devices, minimizing computational and memory demands is always essential. The lighter and simpler their workloads, the better their overall efficiency, responsiveness, and energy sustainability [25].

The application of HE in privacy-preserving fingerprint works currently face two main limitations. The first stems from the high dimensionality of fingerprint images, which leads to heavy computational and storage demands. The second arises from the need for large, complex neural network models capable of effectively learning and generalizing from these high-dimensional inputs.

In this work, we address both challenges through a twofold strategy. First, we introduce a data preprocessing step that significantly reduces the dimensions of fingerprint images to just 28×28 images while preserving nearly all discriminative information in the images. Second, we leverage these compact representations to design a lightweight yet highly efficient neural network that generalizes effectively on the reduced data, achieving nearly perfect accuracies in one-to-many identification tasks. Building on these contributions, we present SentinelTouch, an open-source, privacy-preserving fingerprint authentication and identification framework based on HE. This proposed system implements and evaluates two distinct pipelines for deploying HE privacy-preserving fingerprint identification and authentication models, demonstrating significant improvements in efficiency without compromising accuracy.

Our first approach involved designing and developing an efficient, lightweight, and HE-friendly privacy-preserving neural network model on the server-side. At the client-side, the user applies an image preprocessing technique to the biometric fingerprint image, encrypts the transformed image, and sends the encrypted image to the server. The server then performs feature extraction on the encrypted image using the HE-friendly neural network and an encrypted matching task on the resulting encrypted embeddings. Our experimentation setup takes about 16 seconds for encrypted feature extraction, matching, and identification of a user in a one-to-many identification task. This is a 10x improvement over the state-of-the-art end-to-end privacy-preserving HE biometric fingerprint identification system. To the best of our knowledge, this is the first work to employ a full-privacy pipeline using a HE-friendly neural network for authenticating and identifying users based on biometric fingerprint images. Our second approach also followed the hybrid pipeline proposed in DeepPrint [16] and adopted by many recent related works. In a one-to-many identification task, our hybrid pipeline is able to identify users in just 284 milliseconds, approximately 2.5 times faster than Blind-Match [11].

To achieve SentinelTouch, we also introduce a novel, complete, and efficient algorithm for computing the cosine similarity function in the encrypted domain. This function is very accurate in biometric matching tasks, thus suitable for our scenarios. We evaluate SentinelTouch on the SOKOTO and the PolyU fingerprint datasets. Our results show a Rank-1 accuracy within ±0.1% of state-of-the-art works. Concretely, the core contributions of this work are:

- The design and development of SentinelTouch, an efficient open-source privacy-preserving biometric fingerprint authentication and identification system built on HE, using data preprocessing and a lightweight neural network.
- We propose a novel and highly efficient encrypted domain cosine similarity function algorithm. This algorithm is designed to perform complete fingerprint embedding matching directly and effectively in the encrypted domain.
- We introduce a novel and efficient architecture for HE-based privacy-preserving biometric fingerprint data processing. This architecture leverages a preprocessing stage for compact representation of biometric fingerprint data, thereby decreasing the complexity and size of the neural network required for efficient authentication and identification.
- We evaluate our work using two different pipelines with PolyU and Sokoto datasets. First, we build a full privacy-preserving pipeline where the model is deployed on the server-side. Secondly, we also employ the hybrid pipeline proposed in DeepPrint where the model is deployed on the client-side. Our work outperforms state-of-the-art works by 10x and 2.5x in the full and hybrid pipelines, respectively.

2 Related Works

Homomorphic Encryption (HE) is among the foremost privacy-preserving techniques currently employed to secure sensitive data. Its application in fingerprint identification and authentication systems can be broadly categorized into two main classes.

The first class involves leveraging various HE schemes to develop full privacy-preserving pipelines for fingerprint authentication and identification systems. This approach primarily relies on traditional similarity metrics measurement, such as Euclidean distance and Hamming distance, for user authentication and identification. In these systems, fingerprint templates are typically transformed into encrypted feature vectors or binary representations before being outsourced to the server or cloud. The server then performs distance computations directly on the encrypted data using homomorphic operations, enabling secure matching without ever decrypting the underlying biometric information. Although these approaches ensure full-pipeline data privacy scenarios, they often face challenges related to scalability and accuracy in matching tasks, especially on large biometric databases. Furthermore, the reliance on basic distance metrics limits their ability to exploit the complex and highly discriminative features of fingerprint data. Examples of works in this class include: [3, 21, 32, 41]. Kim et al. [21] stands out as the most efficient work in this class, presenting a privacy-preserving fingerprint authentication system that uses the TFHE scheme to compute encrypted Euclidean distance measurements of biometric fingerprint templates. This system performed a one-to-one biometric fingerprint matching task in about 162 seconds.

The second class of privacy-preserving fingerprint authentication systems utilizes neural networks to generate compact, fixed-length representations, enabling efficient and discriminative matching within the encrypted domain. In 2019, Engelsma et al. introduced DeepPrint [16], a hybrid deep neural network that produces a 192-dimensional embedding from a fingerprint image. The feature extraction layers of the neural network were placed at the client-side, while the matching and user identification tasks were moved to the server. Under HE constraints, DeepPrint achieved a 98.8% accuracy while using approximately 3.4 seconds to identify a user in a one-to-many setting. The authors of DeepPrint demonstrated the compatibility of their approach to both the CKKS and BFV homomorphic encryption schemes. However, DeepPrint suffers from large ciphertext sizes of about 62 MB per query. To address its limitations, Choi et al. proposed Blind-Touch [12], a distributed architecture for fingerprint matching that partitions inference between the client and server. It applies ciphertext compression and leverages parallelism to reduce the ciphertext size to approximately 0.8 MB, enabling full-cycle HE-based one-to-one matching in 650ms. Blind-Touch achieved an accuracy of 93.6% and 98.2% on the PolyU and SOKOTO datasets, respectively. Building on Blind-Touch, Blind-Match [11] proposed a one-to-many identification system using an optimized version of the cosine similarity function. It achieved a 99.68% accuracy on the PolyU dataset with a runtime of just 740ms, while using up to a 128-dimensional embedding feature vector. More recently, Sumalatha et al. [38] advanced the one-to-one setting by leaving only the matching task on the server-side. Using Euclidean distance measurement for similarity, their system attained 99.19% accuracy on the SOKOTO dataset. Blind-Touch and Sumalatha et al. utilize large, custom neural networks designed for high-performance one-to-one matching, each containing millions of parameters. On the other hand, Blind-Match adopts a ResNet-18 architecture with approximately 12 million parameters to handle one-to-many matching tasks.

Unlike the aforementioned systems, our work adopts a fundamentally different approach to privacy-preserving fingerprint authentication and identification, focusing on one-to-many identification tasks, as these are more realistic than one-to-one identification tasks. Instead of directly applying HE to high-dimensional fingerprint images or using neural networks for feature extraction, we introduce a dedicated data preprocessing stage that reduces the dimensionality of the fingerprint images while preserving most of the discriminative information contained in the original image. This compression step not only minimizes storage and transmission overhead but also significantly decreases the computational cost in the encrypted domain. From these compact representations, we develop a lightweight neural network that efficiently learns and generalizes from the compressed fingerprint images. We used the classic principal components analysis and LeNet-5 neural network model for our work. This lightweight yet effective model enables faster one-to-many identification while maintaining high discriminative power, making it particularly suitable for integration with HE schemes. In extensive evaluations, our method achieved a Rank-1 accuracy of 99.849% on the SOKOTO dataset and a Rank-1 accuracy of 99.521% on the PolyU dataset in one-to-many identification tasks. These results demonstrate competitive performance with state-of-the-art systems while offering improved efficiency and

Table 1: Comparison between the Hybrid Pipeline and the Full-Privacy Pipeline in SentinelTouch

Aspect	Hybrid Pipeline	Full-Privacy Pipeline
Application Scenario	Suitable only when the model can be deployed on the client device.	Applicable in all MLaaS and shared-model scenarios. No model is required on the client device.
Performance	Faster, enabling real-time identification of users in about 284ms	Slower, taking about 16 seconds to identify a user.
Client Computation	Higher, as the client handles the model inference workload	Lower, as the model inference workload is outsourced to the cloud.
Client Trusted Compute Base	Larger, since additional plaintext model inference increases the amount of client-side code.	Smaller, as encrypted model inference is fully executed on the untrusted server.

scalability in privacy-preserving environments. We implement a full privacy-preserving neural network pipeline based on the CKKS scheme, which requires only 16 seconds for user identification. We also implemented the hybrid pipeline proposed in DeepPrint where the model performs feature extraction on the client-side. We then outsourced the user identification tasks to the server, which is done in the encrypted domain, and achieved user identification in just 284 milliseconds. SentinelTouch presents a new, privacy-preserving fingerprint authentication and identification system that is realistic for different application settings, utilizing HE for privacy guarantees. In Table 1, we show a comparison of the two pipelines available in SentinelTouch.

3 Background

3.1 Homormophic Encryption

Homomorphic Encryption (HE) is a cryptographic technique that allows computation to be performed directly on encrypted data, producing an encrypted result which, when decrypted, matches the outcome of operations performed on the corresponding plaintexts. This property enables privacy-preserving computation in untrusted environments, such as remote servers or cloud platforms, as sensitive data is never revealed in its raw form. If m_1 and m_2 are messages, Enc denotes the encryption function, and f and f' represent computationally feasible functions operating on ciphertext and plaintext inputs, respectively, then HE ensures:

$$f(\text{Enc}(m_1), \text{Enc}(m_2)) = \text{Enc}(f'(m_1, m_2)) \tag{1}$$

Since the introduction of HE in 1978 as *Privacy Homomorphism* [33], researchers have developed various HE algorithms to enable secure computation on encrypted data. Among these, the Paillier scheme is an early and widely used additive HE method, allowing secure addition of encrypted integers but not supporting multiplicative operations [31]. In contrast, following Gentry's groundbreaking work of 2009, which introduced the first fully homomorphic encryption (FHE) scheme [18], later schemes such as BGV (Brakerski, Gentry, and Vaikuntanathan) [5] and BFV (Brakerski/Fan, and Vercauteren) [17] extended capabilities to support both addition and multiplication on encrypted integers. These schemes strategically improved upon Gentry's construction by introducing different noise management techniques like modulus switching and key switching, which significantly reduced the reliance on costly bootstrapping operations and improved the efficiency of encrypted computations. Other advancements include the TFHE (Fast Fully Homomorphic Encryption over the Torus) scheme [9, 10], designed for fast gate-by-gate encrypted bit computations, and the CKKS (Cheon-Kim-Kim-Song) Scheme [7, 8], which enables encrypted homomorphic operations on floating point arithmetic data.

While our work is compatible with all SIMD-like HE schemes, such as BGV and BFV, we have chosen the CKKS scheme for this work. The CKKS scheme supports efficient approximate arithmetic over real numbers, making it particularly well-suited for privacy-preserving machine learning and statistical computations. Its ability to handle floating-point operations directly aligns with the requirements of fingerprint image processing and identification tasks. These properties enable more natural and scalable processing of real-world data compared to integer-based or bit-focused schemes. Additionally, CKKS offers a good balance between performance and accuracy, which is also critical for achieving practical usability in fingerprint identification works, like SentinelTouch.

The CKKS scheme processes data in batches, where the batch size is referred to as the number of slots. The maximum number of allowable multiplications, known as the depth, is fixed during key generation. Exceeding this depth may compromise the accuracy of the decrypted result. To overcome this limitation, bootstrapping can be applied to refresh ciphertexts and refresh the noise levels, enabling unlimited computations at the expense of additional computational overhead. CKKS supports three core homomorphic operations on ciphertexts: Addition, Multiplication, and Rotation. Each operation can be performed between ciphertexts or between a ciphertext and a plaintext (constant). Let N be the number of slots, and $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^N$ be real vectors. Denote by $C(\mathbf{v})$ the ciphertext encrypting vector $\mathbf{v}$. The operations are formally defined as:

$$\text{Add}\big(C(\mathbf{v}_1), x\big) = C(\mathbf{v}_1 \oplus x) \tag{2}$$

$$\text{Mul}\big(C(\mathbf{v}_1), x\big) = C(\mathbf{v}_1 \otimes x) \tag{3}$$

$$\text{Rot}\big(C(\mathbf{v}), r\big) = C\big(v_r, v_{r+1}, \ldots, v_{N-1}, v_0, \ldots, v_{r-1}\big) \tag{4}$$

where x is either a plaintext vector $\mathbf{v}_2$ or a ciphertext $C(\mathbf{v}_2)$, and $\oplus, \otimes$ denote element-wise addition and multiplication, respectively. The rotation operation cyclically shifts the slots by a non-zero integer r either to the right or to the left. We utilize the CKKS implementation from the OpenFHE library [2], which provides a mature, efficient, and actively maintained open-source implementation. OpenFHE supports optimized CKKS operations, bootstrapping, and has extensive documentation, making it an ideal choice for building scalable privacy-preserving applications like SentinelTouch.

4 Data Preprocessing

Data preprocessing is an important stage in many machine learning pipelines. It serves to reduce redundancy, highlight discriminative

features, and standardize input data for effective learning [28, 43]. The quality and efficiency of model training are often determined by the quality of data used, and thus the robustness of the preprocessing techniques applied. Most fingerprint authentication and identification systems employing neural networks, such as [11, 12], operate on minimally processed images. While this approach simplifies the machine learning pipeline, it often leads to inefficiencies; thus, models must learn on noisy and redundant pattern information, which increases the complexity of learning. Furthermore, the absence of a structured preprocessing stage results in suboptimal utilization of computational resources as larger models are required for efficient feature extraction and generalization.

The SOKOTO fingerprint dataset provides images of size 96×108, whereas the PolyU dataset contains images of size 350×225. The SOKOTO dataset is captured using a low-cost optical Hamster plus (HSDU03PTM) and SecuGen SDU03PTM sensor scanners [36]. On the other hand, the PolyU dataset is acquired using high-resolution capacitive scanners based on contactless technology, which captures much cleaner images in 2D/3D at 384×384 resolution [22, 24]. These datasets both undergo very basic preprocessing steps, such as region-of-interest (ROI) extraction and scaling, to ensure consistent ridge spacing and orientation across samples. While these operations normalize image size, remove irrelevant borders, and reduce noise, they are not feature-aware transformations; thus, they are not based on the statistical structure of the information present in the images. While higher-resolution inputs are rich in detail, they also dramatically increase the dimensionality of the data. In a neural network, this translates directly into higher neuron counts, higher memory requirements, and longer training and inference times. In most conventional computing environments, these costs are manageable due to the abundance of computational resources. However, in the encrypted domain, such increases are significantly more problematic. Larger input dimensions require more slots in the ciphertext, leading to a steep (often exponential) growth in the cost of homomorphic operations, thus image processing. Similarly, an increase in model size drastically inflates the runtime of encrypted model inference. Consequently, the biometric fingerprint image size and the appropriate neural network architecture required for efficient learning become critical constraints when developing HE-based privacy-preserving fingerprint systems.

Fingerprint images represent structured data points characterized by repeated ridge–valley patterns, where much of the pixel-level information is spatially correlated. These correlations mean that large portions of the fingerprint images carry redundant information that does not significantly contribute to identity discrimination. Our hypothesis in this work is that if these images are carefully preprocessed, We can drastically reduce their dimensionality by orders of magnitude while preserving the majority of discriminative features. This is possible because the essential identity-bearing information in fingerprints, such as ridge flow and minutiae points, occupies a lower-dimensional manifold within the original high-dimensional image space. By leveraging advanced data preprocessing techniques, such as Principal Component Analysis (PCA), we can analyze the variance of discriminative information in fingerprint images and use that information to produce compact, noise-robust representations of the images. Such smaller representations are not only easier for neural networks to learn, but also reduce the required model size, improve training and inference speed, and lower computational cost. With these changes, we can address the current limitations of privacy-preserving fingerprint identification systems that rely on HE.

4.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a linear dimensionality reduction technique that projects high-dimensional data onto a lower-dimensional subspace while preserving the directions of maximum variance [1]. For fingerprint images, PCA can compactly represent ridge–valley structures by emphasizing the components that capture the most discriminative information. Based on our understanding of this information, we can create a compact and more efficient representation of the fingerprint images without redundancies. Let a fingerprint image be represented as a vector $\mathbf{x} \in \mathbb{R}^n$, where n is the number of pixels. Given a dataset of m images $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$, PCA computes the covariance matrix as:

$$C = \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top, \tag{5}$$

where $\bar{\mathbf{x}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i$ is the mean image. The principal components are given by the top k eigenvectors of C, forming a transformation matrix $W \in \mathbb{R}^{k \times n}$. Each image is then projected onto this mean transformative representation as:

$$\mathbf{y} = W\mathbf{x}, \quad k \ll n, \tag{6}$$

reducing the dimensionality from n to k while retaining the majority of the discriminative information through the variance.

When applying PCA to our datasets, we first analyzed the variance distribution of the fingerprint images to determine the number of principal components needed to retain most of the discriminative information. Based on this analysis, we computed the cumulative variance explained by the principal components and plotted the percentage coverage for each dataset, as shown in Figure 1. This evaluation enabled us to examine the trade-off between variance retention and dimensionality reduction, thereby deriving a compact yet informative representation. Our goal was to maintain sufficient discriminative power for identity recognition while minimizing computational and memory costs, which are essential factors, especially in the HE setting.

The lightweight LeNet-5 neural network architecture is a well-established and thoroughly studied architecture for classifying grayscale images of relatively low dimensionality [6, 23]. The standard architecture uses input images of 28x28 dimensions (784 pixels), thus a natural reference point for our PCA-based dimensionality reduction. For the SOKOTO dataset, we observed that 784 principal components capture approximately 83.65% of the total variance, whereas for the PolyU dataset, 784 principal components cover 97.93% of the variance within the images. This indicates that, despite the difference in image resolution, the intrinsic dimensionality of the fingerprint features is significantly lower than that of the raw pixel space; thus, compact representations of these fingerprint images can be easily generated for this lower dimension.

By projecting the fingerprint datasets onto 784 principal components, we align the reduced representation with the canonical input structure used by this model. Although this projection significantly lowers the input dimensionality, the retained variance confirms

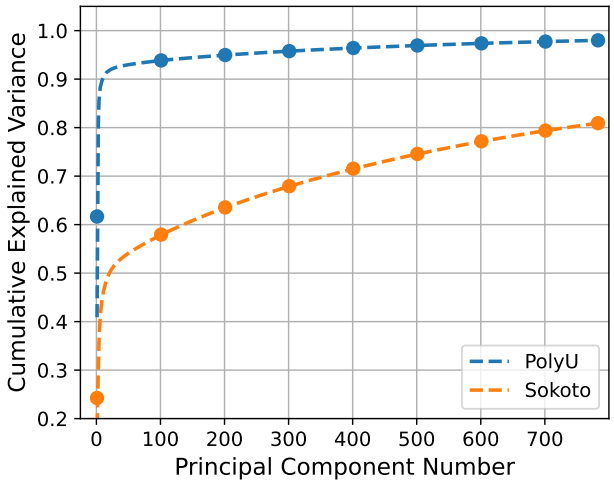

Figure 1: Cumulative explained variance by principal component for the PolyU and Sokoto datasets, illustrating how much percentage of discriminative information is covered by how many principal components in each dataset.

that the majority of discriminative, identity-relevant information is preserved. From the perspective of HE, the choice of dimensionality is particularly critical as higher-dimensional inputs require larger ciphertexts, more packing slots, and consequently higher polynomial modulus degrees, which directly increase both memory usage and computational overhead. In contrast, constraining the representation to 784 components reduces ciphertext dimensions, enabling smaller modulus sizes and thereby improving efficiency, scalability, and inference speed in the encrypted domain. Thus, the selection of 784 components strikes a balance between maintaining discriminative information, ensuring compatibility with the standard LeNet-5 architecture, and achieving substantial efficiency gains for encrypted-domain computation. This balance between compression and fidelity is key. PCA not only reduces computational complexity but also optimizes the feasibility of deploying privacy-preserving models in practice. Our principal component selection strategy, therefore, serves both algorithmic and cryptographic requirements, retaining sufficient discriminative power while lowering HE parameterization costs. Figure 2 shows a sampled image from the PolyU dataset with its transformed versions after PCA analysis, while Figure 3 shows the same information for a randomly sampled fingerprint image from the SOKOTO dataset.

4.2 LeNet-5 Model

Originally introduced by LeCun et al. (1998) for handwritten digit recognition, LeNet-5 employs compact convolutional filters that are well suited to capture ridge-based local patterns in our compact fingerprint images [23]. Its shallow depth results in a lightweight model, making it especially suitable for the computational constraints of HE. Compared to modern deep architectures, LeNet-5 contains only about forty thousand parameters, which is orders of magnitude fewer than the custom CNNs and ResNet-18 models employed in related works. This compact size provides an effective

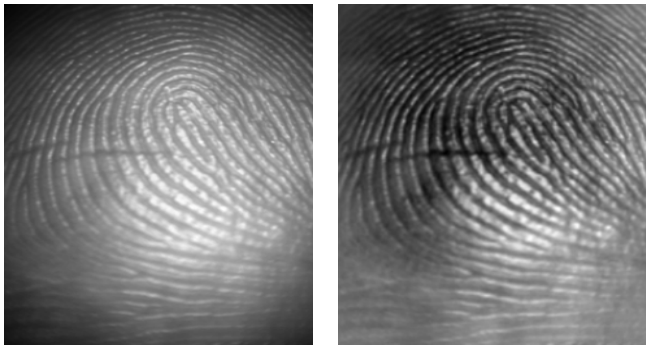

Figure 2: Comparison of original fingerprint image from PolyU (left) and PCA reduced image (right). After resizing the PCA-reduced image, the image size decreases by approximately 99%, but all the relevant ridge-valley structures remain extremely visible.

Figure 3: Comparison of original fingerprint image from Sokoto (left) and PCA reduced image (right). After resizing the PCA-reduced image, the image size decreases by approximately 92.44%, yet all the patterns remain clearly visible.

trade-off between accuracy and computational efficiency, making it particularly suitable for deployment in the encrypted domain.

In our design, we retain the sequential feature extraction layers of LeNet-5 but replace the original final classifier with an embedding projection layer. This modification maps each fingerprint image into a fixed-dimensional embedding vector, providing a compact and discriminative representation of identity. These embeddings are encrypted, and recognition is then performed by comparing encrypted embeddings rather than relying on direct classification. This separation of feature learning from decision making enables more flexible and scalable matching strategies, user registration, and deletions on the fingerprint system. Figure 4 shows the LeNet-5 architecture employed in this work.

5 Threat Model

Consistent with prior work on privacy-preserving computation using HE, our system assumes a trusted client that exclusively holds the private key in the protocol. The client is responsible for executing the data preprocessing stage in plaintext. The client is also responsible for encrypting the preprocessed inputs before transmitting them to the server. Finally, the client decrypts the results returned from the server and interprets them. We assume that the client maintains secure key management practices throughout the system's lifecycle. The confidentiality guarantees of our design rely

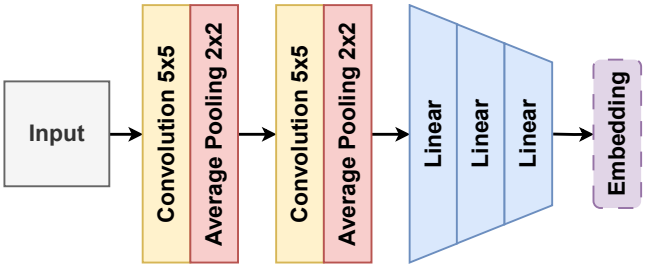

Figure 4: The standard LeNet-5 architecture employed in this work with 5x5 kernels for the convolution layers and 2x2 kernels for the pooling layers.

fundamentally on the secrecy and integrity of the client's private key; therefore, this key is assumed to be accessible only to the client and should never be shared.

The server is considered untrusted but is modeled under a semi-honest, or honest-but-curious, adversarial setting. In this model, the server strictly follows the protocol, performing the computations exactly as specified, but may attempt to extract additional information from the ciphertexts, metadata, or computation artifacts it observes. While the server does not actively manipulate the inputs, outputs, or intermediate computations to subvert the process, it remains curious and seeks to learn any possible information about the client's data. This setting captures a realistic cloud-computing scenario in which the service provider is assumed to be computationally powerful and protocol-compliant, yet untrusted with respect to data privacy. All server-side computations are performed entirely in the encrypted domain using the CKKS scheme with IND-CPA security as implemented by the OpenFHE library. During user registration, the client encrypts its processed input, transmits it to the server, and the server stores the resulting embeddings exclusively as ciphertexts. During identification, the client captures a new biometric sample, processes it, and encrypts it before sending it to the server for verification. The only plaintext data present at the server are the model weights, which are not user-specific and can be trained using publicly available datasets. They should be owned by the server provider, further ensuring that they reveal no sensitive information about enrolled users' private data.

SentinelTouch supports two deployment scenarios with differing client-side requirements. In the full privacy-preserving pipeline, the client encrypts preprocessed data and sends it to the server, which performs all subsequent feature extraction and matching in the encrypted domain. This approach minimizes client-side plaintext computation, thereby reducing the Trusted Computing Base (TCB). In the hybrid pipeline, the client additionally performs feature extraction locally before encrypting and transmitting the feature vector for server-side matching. While this reduces the encrypted server-side workload and improves overall performance, it increases client-side computation and expands the TCB, since additional code is required for model inference on the trusted client. Although both pipelines share the same threat model for the client and server devices, the full privacy-preserving pipeline offers a stronger security posture due to fewer plaintext computations. Reduced code on the client means a smaller TCB and thus a lower risk of compromise.

HE provides strong confidentiality guarantees for biometric fingerprint processing by supporting computation directly on encrypted data, but practical deployments must consider potential information leakage through side channels. Effective key management remains essential, as private key compromise would undermine the confidentiality promise of computations and secure storage of embeddings. Under these assumptions, SentinelTouch ensures end-to-end confidentiality of biometric fingerprint processing during both user enrollment and identification.

6 SentinelTouch System Design and Overview

The objective of SentinelTouch is to deliver a lightweight and efficient privacy-preserving fingerprint authentication and identification system that operates within the constraints of HE. To achieve this, SentinelTouch combines an efficient fingerprint image preprocessing stage with a compact neural network architecture, while carefully tuning HE parameters to balance accuracy and performance in the encrypted domain. The pipeline begins with secure key generation performed by the system administrator. The client device is provided with the set of private and public keys either through a secure communication channel or directly generated on the device. The private key should be stored in a secure enclave or equivalent trusted environment. The corresponding evaluation keys required for the encrypted computations are uploaded to the server. This separation ensures that the client retains full control of sensitive decryption material, while the server only operates on encrypted data using public and evaluation keys. A neural network is first trained, and the resulting weights are exported. In the full-privacy pipeline, an equivalent lightweight, HE-friendly model is constructed and deployed on the server using these weights. The model's design is carefully optimized to minimize computational overhead while maintaining accuracy and privacy. In the hybrid pipeline, an equivalent plaintext model is deployed on the client device for local feature extraction.

During authentication, the client preprocesses its fingerprint input (including PCA projection and resizing). In the full-privacy pipeline, the preprocessed fingerprint is encrypted using the client's public key and transmitted to the server. The HE-friendly model then performs inference entirely in the encrypted domain, producing an encrypted embedding that is matched against encrypted user embeddings on the server. In the hybrid pipeline, the client performs both preprocessing and feature extraction locally, encrypts the resulting feature vector, and transmits it to the server for encrypted-domain matching. In both pipelines, the server operates exclusively on ciphertexts and never gains access to any biometric fingerprint data or intermediate plaintext features. Figure 5 illustrates SentinelTouch 's architecture for fingerprint registration and matching in the full privacy-preserving pipeline, while Figure 6 depicts the corresponding hybrid pipeline. (These Figures are not drawn to scale).

6.1 User Registration Phase

The registration phase initializes each user's encrypted fingerprint template in the system database. Its primary objective is to securely populate the enrollment database with reference embeddings while minimizing ciphertext growth and computational overhead. Each

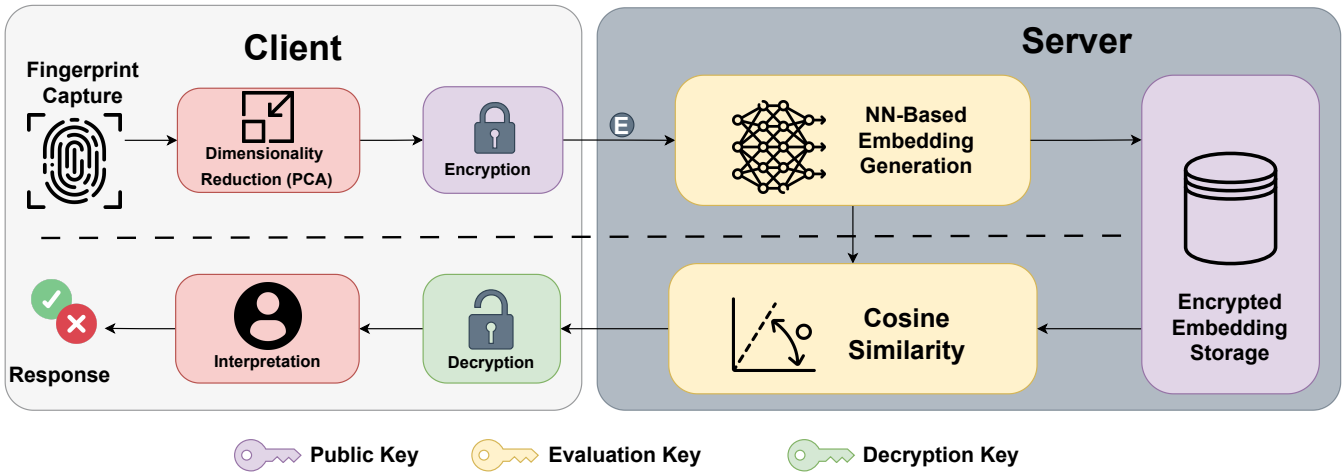

Figure 5: System architecture of SentinelTouch showing the various stages in the end-to-end fingerprint matching pipeline. Generated embeddings from the HE-based neural network are stored only during enrollment, denoted by E. The bottom half of the pipeline is used for authentication and is inactive during enrollment.

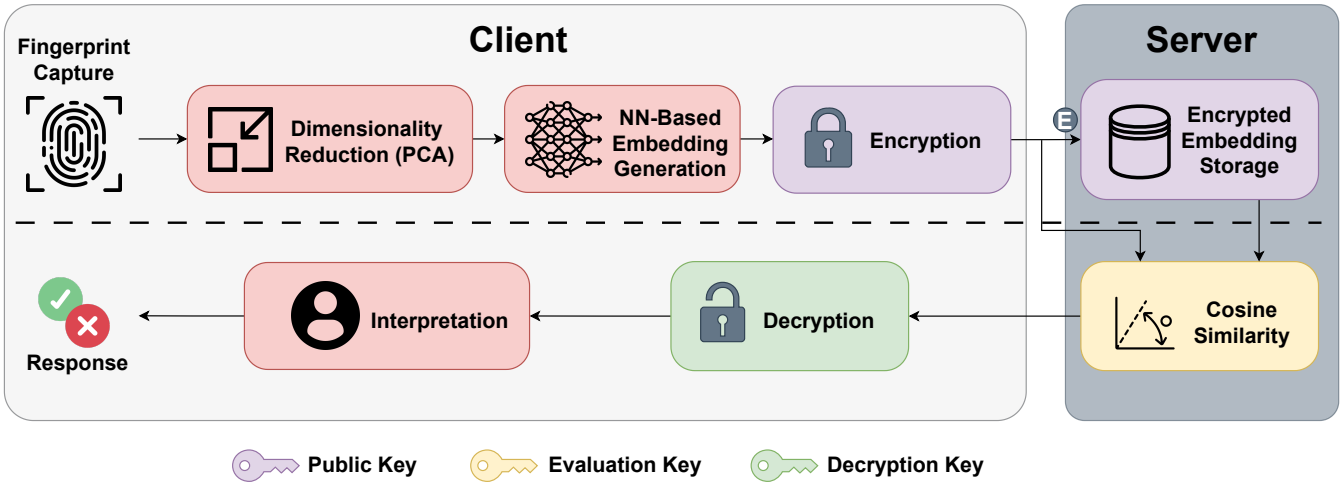

Figure 6: System architecture of SentinelTouch showing the various stages in the end-to-end fingerprint matching pipeline. Generated embeddings from the HE-based neural network are stored only during enrollment, denoted by E. The bottom half of the pipeline is used for authentication and is inactive during enrollment.

user submits their fingerprint sample through the client device. Local preprocessing is performed on the client side, including normalization, PCA-based dimensionality reduction, and resizing, to produce the compact input representation expected by the lightweight neural network model. In the hybrid pipeline, this input is further processed on the client side, and the resulting embeddings are normalized prior to encryption, similar to the approach used in Blind-Match [12]. Normalizing the embeddings is important in this pipeline because it simplifies the encrypted cosine similarity computation by transforming the vector magnitudes to 1. On the other hand, in the full-privacy pipeline, the resulting input image is flattened and encrypted using the client's public key, and then transmitted to the server. Upon receiving the encrypted image, the server executes the encrypted inference pipeline, producing a dimensional encrypted embedding for each fingerprint image.

SentinelTouch leverages the SIMD-style packing capability of the CKKS scheme to efficiently manage fingerprint embeddings in the encrypted domain. We achieve this by constructing and storing embeddings using a compact vertical representation approach. Let the embedding dimension be D, and each ciphertext provides S slots; the database is organized as a set of D ciphertexts, where the i-th ciphertext stores the i-th coordinate of all user embeddings, with each slot corresponding to a unique user ID. To register a new user with ID u, the client first provides an encrypted embedding packed in a single ciphertext. For each embedding dimension, the ciphertext is rotated so that the desired coordinate is brought into the first slot, which is then isolated using a clean mask. The extracted value is subsequently rotated into the correct slot corresponding to the new user ID, producing a new set of D ciphertexts that contain the user's embedding components aligned at the appropriate indices.

Algorithm 1 User Registration in SentinelTouch

Require: Embedding dimension D, slot capacity S, user ID u, encrypted embedding E
Ensure: Updated encrypted fingerprint database

Create cleaning mask $M \leftarrow$ create_index_mask$(0, S)$
Initialize empty list of ciphertexts $\{U_0, \ldots, U_{D-1}\}$
for $i \leftarrow 0$ to $D-1$ **do**
  $T \leftarrow \text{EvalRotate}(E, i)$ ▷ Bring coordinate i to slot 0
  $T \leftarrow \text{EvalMult}(T, M)$ ▷ Isolate coordinate
  if $u > 0$ **then**
    $T \leftarrow \text{EvalRotate}(T, -u)$ ▷ Place into user slot u
  end if
  $U_i \leftarrow T$
end for
if $u > 0$ **then**
  $\{C_0, \ldots, C_{D-1}\} \leftarrow$ Deserialize stored ciphertext C
  for $i \leftarrow 0$ to $D-1$ **do**
    $C_i \leftarrow \text{EvalAdd}(C_i, U_i)$
  end for
  Serialize updated database $\{C_0, \ldots, C_{D-1}\}$
else
  Serialize $\{U_0, \ldots, U_{D-1}\}$ as first user
end if
return success (0) if storage succeeds, else failure (1)

Once this new set is constructed, the database is deserialized from storage, and each new ciphertext is added to it's corresponding ciphertext in the database, thereby inserting the user's embedding without affecting existing entries. Finally, the updated database is serialized and stored.

Algorithm 1 illustrates the user registration process in SentinelTouch. By constructing vertical representations of user embeddings in the database, SentinelTouch achieves both storage efficiency and computational simplicity. Each ciphertext corresponds to a single embedding dimension and contains the values for all users across its slots. This layout enables the addition of new users without disrupting existing entries. Moreover, this structure enables highly efficient matching. Since embeddings are stored vertically, cosine similarity computations can be performed dimension-wise across all users simultaneously. This not only reduces the number of homomorphic operations required but also leverages the SIMD-style packing inherent to the CKKS scheme, thereby maximizing parallelism and minimizing overhead.

6.2 User Authentication and Identification

When a registered user u attempts to authenticate, the client captures a fresh fingerprint sample and performs identical preprocessing steps as in the registration phase, including PCA projection, resizing, and feature extraction. The resulting compact feature vector encrypted under the client's public key is matched against the encrypted database. Since the database already stores encrypted reference embeddings, authentication reduces to a secure similarity search problem. To this end, we adopt the widely used cosine similarity function, commonly employed in modern biometric systems such as CosFace [39], ArcFace [14], and SphereFace [26]. The cosine similarity between two embeddings $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ is described mathematically by Equation 7:

$$\text{cosine}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2}. \tag{7}$$

To enable parallel comparison of a query embedding against all stored embeddings, the server first replicates the query embedding across ciphertext slots using homomorphic rotations. In SentinelTouch, each embedding dimension is stored in a separate ciphertext, so the replication process is performed independently for every dimension. This produces a set of ciphertexts where each dimension of the query embedding occupies multiple slots, allowing the server to compute cosine similarity with multiple users in parallel across all dimensions. Rather than replicating slot by slot, which would be inefficient, we implement an algorithm that iteratively doubles the ciphertext content using powers-of-two rotations and additions. At each step, the current ciphertext is rotated and added to itself, effectively doubling the number of slots containing the query embedding in that dimension. This process is repeated until the number of replicated slots meets or exceeds the total number of enrolled users. Algorithm 2 illustrates the embedding replication process in SentinelTouch, showing how each embedding dimension is expanded across ciphertext slots to enable parallel, SIMD-style matching against all enrolled users.

Algorithm 2 Query Embedding Replication in SentinelTouch

Require: Encrypted query embedding E, number of users S, embedding dimension D
Ensure: Set of replicated ciphertexts $\{U_0, \ldots, U_{D-1}\}$

Create ones mask $M \leftarrow$ create_index_mask$(0, S)$
Initialize empty list of ciphertexts $\{U_0, \ldots, U_{D-1}\}$
for $i \leftarrow 0$ to $D-1$ **do**
  $T \leftarrow \text{EvalRotate}(E, i)$ ▷ Bring coordinate i to slot 0
    ▷ Replicate embedding across all user slots
  $T \leftarrow \text{EvalMult}(T, M)$
  $T \leftarrow \text{EvalAdd}(T, \text{EvalRotate}(T, -1))$
  for $k \leftarrow 1$ to $\lfloor \log_2 S \rfloor - 1$ **do**
    $p \leftarrow 2^k$
    $T \leftarrow \text{EvalAdd}(T, \text{EvalRotate}(T, -p))$
  end for
  $U_i \leftarrow T$ ▷ Store replicated ciphertext for dimension i
end for
return $\{U_0, \ldots, U_{D-1}\}$

After the embedding replication process, the dot product computation is performed on the generated query ciphertexts against the encrypted database. The dot product is computed by multiplying corresponding elements of the two vectors and summing the results. In SentinelTouch, this dot product calculation is highly efficient because user embeddings are stored vertically, allowing the dot product to be computed for all users simultaneously. This process is detailed in Algorithm 3.

In the hybrid pipeline, we adopt the optimized approach proposed in Blind-Match. Since all embeddings are unit-magnitude normalized in plaintext before being transmitted to the server, the cosine similarity between two embeddings $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ simplifies to a dot product computation as shown in Equation 8

$$\text{cosine}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{1}\|_2 \|\mathbf{1}\|_2}. \tag{8}$$

Algorithm 3 Dot Product Computation in SentinelTouch

Require: Query embeddings $\{U_0, \ldots, U_{D-1}\}$, database embeddings $\{C_0, \ldots, C_{D-1}\}$, embedding dimension D
Ensure: Cosine dot product for all users
Initialize empty list of ciphertexts $V \leftarrow []$
for $i \leftarrow 0$ to $D-1$ **do**
  $T \leftarrow \text{EvalMult}(U_i, C_i)$
  Append T to V
end for
$S \leftarrow \text{EvalAddMany}(V)$ ▷ Sum all dimension-wise products
return S ▷ Contains dot products for all users

In the full-privacy pipeline, both the magnitudes of the vectors $\mathbf{x}$ and $\mathbf{y}$ and the division by their product must be computed directly in the encrypted domain. Since HE supports only a limited set of arithmetic operations efficiently, computing cosine similarity requires a carefully adapted algorithm tailored to these constraints. To this effect, we propose a novel and highly efficient algorithm for performing complete cosine similarity computations within the encrypted domain that can be used for our matching task.

The magnitude of a vector is defined as the square root of the sum of the squares of its elements, as shown in Equation 9:

$$\|\mathbf{x}\|_2 = \sqrt{\sum_i x_i^2}, \quad \|\mathbf{y}\|_2 = \sqrt{\sum_i y_i^2}. \tag{9}$$

However, the square root is a non-linear operation and cannot be directly computed in HE. To overcome this, we reformulate the cosine similarity function to minimize the number of non-linear operations needed in it's evaluation. First, we square each element of the vectors and sum them, obtaining the squared components of the magnitudes. Instead of computing each square root separately, we exploit the commutative property of multiplication of square roots as shown in Equation 10.

$$\|\mathbf{x}\|_2 \, \|\mathbf{y}\|_2 = \sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2} = \sqrt{\left(\sum_i x_i^2\right)\left(\sum_i y_i^2\right)}. \tag{10}$$

This allows us to combine the two magnitude computations into a single inverse square root operation. The cosine similarity can then be expressed as shown in Equation 11

$$\text{cosine}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y}) \cdot \frac{1}{\sqrt{\left(\sum_i x_i^2\right)\left(\sum_i y_i^2\right)}}. \tag{11}$$

With this reformulation, only one non-linear operation, the inverse square root, needs to be performed in the encrypted domain. To compute this efficiently under the CKKS scheme, we adopted and implemented the Newton-Raphson approximation under HE constraints [42]. Our Newton-Raphson approximation iteratively refines the estimate of the inverse square root while remaining compatible with homomorphic operations.

Algorithm 4 summarizes the homomorphic magnitude calculations used in SentinelTouch. The algorithm first computes the dot product $\mathbf{x} \cdot \mathbf{y}$ as described in Algorithm 3. To obtain the magnitudes of the inputs in the encrypted domain, each vector is squared element-wise and summed to yield $\sum_i x_i^2$ and $\sum_i y_i^2$. These results are then combined following Equation 10, after which the inverse square root is efficiently approximated using Newton-Raphson iterations as shown in Algorithm 5. Finally, the dot product is multiplied by this inverse square root to compute the cosine similarity, as defined in Equation 11.

We use the Newton–Raphson method to approximate the inverse square root because, with a suitable initial guess, it achieves high accuracy using only a few iterations. In SentinelTouch, using an initial guess of 0.5 with just two iterations worked perfectly well for all our experimental scenarios. Its update relies solely on additions and multiplications, making it efficient and compatible with homomorphic arithmetic while minimizing noise growth. This approach substantially reduces computational complexity and mitigates noise growth, making full-privacy cosine similarity evaluation both feasible and efficient for encrypted fingerprint embeddings. It is also worth noting that, due to the structural representation of users in our encrypted database, this computation is executed in parallel across all users simultaneously, thereby significantly reducing the computational overhead and enabling the one-to-many matching to be carried out in one shot.

Algorithm 4 Homomorphic Magnitude and Cosine Similarity

Require: Encrypted query embeddings $\{U_0, \ldots, U_{D-1}\}$, database embeddings $\{C_0, \ldots, C_{D-1}\}$, embedding dimension D, precomputed dot product d (from Algorithm 3)
Ensure: Encrypted cosine similarity for all users
Initialize empty lists $S_U \leftarrow []$, $S_C \leftarrow []$
for $i \leftarrow 0$ to $D-1$ **do**
  $su \leftarrow \texttt{EvalSquare}(U_i)$ ▷ square query dimension
  append su to S_U
  $sc \leftarrow \texttt{EvalSquare}(C_i)$ ▷ square database dimension
  append sc to S_C
end for
$s_x \leftarrow \texttt{EvalAddMany}(S_U)$ ▷ compute $\sum_i x_i^2$
$s_y \leftarrow \texttt{EvalAddMany}(S_C)$ ▷ compute $\sum_i y_i^2$
$m \leftarrow \texttt{EvalMult}(s_x, s_y)$ ▷ magnitude product, Eq. (10)
$r \leftarrow \texttt{approx\_rsqrt}(m)$ ▷ inverse square root (Newton–Raphson)
$c \leftarrow \texttt{EvalMult}(d, r)$ ▷ cosine similarity, Eq. (11)
return c

Algorithm 5 Homomorphic Inverse Square Root via Newton-Raphson Approximation

Require: Encrypted input z, encrypted initial guess y_0, iteration count d
Ensure: Encrypted approximation of $1/\sqrt{z}$
$y \leftarrow y_0$
for $n \leftarrow 0$ to $d-1$ **do**
  $y^2 \leftarrow \texttt{EvalSquare}(y)$
  $g \leftarrow \texttt{EvalMult}(z, y^2)$
  $t \leftarrow \texttt{EvalSub}(1.5, \texttt{EvalMult}(0.5, g))$
  $y \leftarrow \texttt{EvalMult}(y, t)$
end for
return y

The results of the homomorphic cosine similarity computation are returned to the client, who then decrypts the ciphertext to obtain and interpret the similarity scores.

6.3 User Deletion Phase

The user deletion process in SentinelTouch allows a registered user's encrypted fingerprint embeddings to be securely and efficiently removed from the enrollment database without compromising the privacy or integrity of other users. This operation is essential for user revocation compliance and maintaining overall database consistency. When a deletion request is issued, the server first verifies its authenticity using a digitally signed request from the system administrator or an authorized revocation authority. This step ensures that unauthorized or malicious deletion attempts cannot be executed by an adversary. Once verified, the server constructs a binary mask with 0 at the slot corresponding to the target user and 1s in all other positions. The database is then deserialized, and each relevant ciphertext C_{cIdx} is multiplied homomorphically by this mask. This operation zeroes out the target user's embedding while leaving all other users' embeddings intact. The updated ciphertexts are then serialized and stored back in the database. If the operation succeeds, SentinelTouch returns 0; otherwise, it returns 1. Importantly, since the operation is performed entirely in the encrypted domain, no plaintext embeddings are ever exposed. For efficient storage management, the client maintains a record of freed embedding slots. When registering new users, these available slots can be reused before allocating new ciphertexts, preventing database fragmentation and optimizing resource utilization. The full deletion procedure is summarized in Algorithm 6.

Algorithm 6 User Deletion in SentinelTouch

Require: User ID u, embedding dimension D, encrypted database ciphertexts $\{C_0, \ldots, C_{D-1}\}$
Ensure: Updated fingerprint database with user u removed
1: Create deletion mask $M \leftarrow$ create_index_mask(u, num_users) ▷ 0 at user u, 1 elsewhere
2: **for** $i \leftarrow 0$ to $D-1$ **do**
3: $C_i \leftarrow \text{EvalMult}(C_i, M)$ ▷ Zero out target user
4: **end for**
5: Serialize updated database $\{C_0, \ldots, C_{D-1}\}$
6: **return** success (0) if storage succeeds, else failure (1)

7 Experiments

In this study, we conducted experiments on a system equipped with an Intel Core i7-14700K 3.4GHz 20-core processor and 32GB of RAM. This work was developed using OpenFHE v1.3.1 (Released July 11, 2025). Our plaintext baseline models were developed in Python using PyTorch. The HE-friendly Convolution and Pooling layers were developed based on the vector encoding approach proposed in Gazelle, which has been widely adopted by HE-friendly neural networks [20] using FHEON [29]. We also employed the Chebyshev polynomial approximation for evaluating the non-linear ReLU function, which has also been widely adopted by HE-friendly neural networks [30]. SentinelTouch is openly available on GitHub at https://github.com/stamcenter/sentineltouch.

7.1 HE Security Parameters

For all experiments, we used the following security parameters. In the full-privacy pipeline, we employed a polynomial degree of 2^{13} with 2^{12} slots (4096), a multiplicative depth of 27, a first modulus size of 54, a rescaling factor of 50, 4 digits for key switching, and the `flexibleauto` rescaling strategy. For the hybrid pipeline, we used a polynomial degree of 2^{12} with 2^{11} slots (2048), multiplicative depth of 16, a first modulus size of 26, a rescaling factor of 20, 2 digits for key switching, and the `flexibleauto` rescaling strategy. The settings in the hybrid pipeline were inspired by the configurations of Blind-Match to ensure a fair comparison, while the settings for the full-privacy pipeline were adopted from FHEON [29].

7.2 Datasets

Our experiments were conducted using two publicly available fingerprint datasets: the SOKOTO and PolyU datasets.

Sokoto Coventry Fingerprint Dataset (SOCOFing or SOKOTO). The Sokoto dataset is a large-scale fingerprint dataset created for biometric research, comprising more than 6,000 images from 600 African subjects. The fingerprints were captured using an optical sensor, with each subject contributing impressions from all ten fingers. For experimentation, we partitioned the data into 3, 600 training, 1, 200 validation, and 1, 200 test images. Additionally, we created 8,400 imposter pairs, maintaining a 1:7 ratio between genuine and imposter datasets, similar to the standards used in most related works. This dataset is particularly useful for analyzing the resilience and generalization of fingerprint recognition systems.

PolyU High-Resolution Fingerprint Dataset (PolyU). The PolyU dataset, developed by the Hong Kong Polytechnic University, is a widely adopted benchmark for fingerprint recognition. It provides high-resolution images captured at 1200 dpi, offering fine ridge and minutiae details that support precise recognition and enhancement studies. PolyU includes both contact-based and contactless-2D data; in this work, we use the processed contactless subset. This subset consists of two sessions: Session I contains 336 subjects (each with 6 images), while Session II contains 160 subjects (each with 6 images), totaling 496 participants. Following the dataset protocol, we assigned 296 subjects for training and 200 for testing. The evaluation set comprises 3, 000 genuine and 19, 900 imposter pairs, providing a rigorous basis for evaluation, similar to most related works.

Together, these two datasets provide complementary evaluation settings: SOKOTO introduces variability from altered and distorted samples, while PolyU emphasizes high-quality minutiae-rich fingerprints. Using both datasets allows for a balanced assessment of system performance in both challenging and controlled conditions.

7.3 Rank-1 Accuracy

The accuracy measurements for SentinelTouch were done for varying embedding feature vector sizes in both full-privacy and hybrid pipelines using the different datasets. To evaluate the impact of embedding feature dimensionality and privacy-preserving configurations on SentinelTouch performance, we conducted experiments on both the PolyU and SOKOTO datasets using feature vectors of 16 and 32 embedding dimensions. Our results demonstrate that the choice of feature vector size has only a marginal effect on the Rank-1 accuracy. Specifically, across all settings, the observed difference between the 16 and 32 feature vectors was bounded within 0.25%, further supporting the findings of Blind-Match that substantial

compression of feature vectors is possible without significantly sacrificing recognition accuracy. Importantly, smaller feature vectors reduce the size of the encrypted database required to store users' embeddings. It also reduces the latency of encrypted matching in SentinelTouch, and directly affects the storage requirements.

On the PolyU dataset, SentinelTouch achieved a 99.512% Rank-1 accuracy with a 32-dimensional embedding, compared to 99.397% when using 16 dimensions. A similar trend was observed on the SOKOTO dataset, where we achieved 99.85% Rank-1 accuracy with a 32-dimensional embedding and 99.80% with a 16-dimensional embedding. These results confirm that feature vector dimensionality has only a negligible influence on overall accuracy; thus, 16-dimensional embeddings are recommended for real-world adoption, as they save storage space. Table 2 shows the Rank-1 ranked accuracies obtained on SentinelTouch on both the SOKOTO and PolyU datasets.

Table 2: Rank-1 accuracy of SentinelTouch using LeNet-5 under the 16 and 32 dimensional embeddings.

Dataset	16-dim embedding	32-dim embedding
PolyU	99.397%	99.512%
SOKOTO	99.800%	99.849%

7.4 Execution Time

We evaluate the execution time of SentinelTouch across different embedding feature vector dimensions and pipeline settings. We report the latencies, including; (i) PCA preprocessing and encryption, (ii) feature extraction, (iii) user identification, (iii) decryption and interpretation of results. The latency recorded here is an average of multiple runs of a random sample from the validation set.

The specific dataset does not affect the latency of SentinelTouch. The additional runtime impact from 16-dimensional embedding to 32-dimensional embedding is only visible in the matching and storage phases and comes from the additional number of identical operations required for the space. Doubling the embedding dimensions from a 16-dimensional embedding to 32-dimensional embedding also doubles the matching time. Using the LeNet-5 model and 16-dimensional embedding space, the full-privacy pipeline required approximately 16 seconds for a complete user identification task, while the hybrid pipeline required only 284 milliseconds for the same one-to-many task. While this slowdown is nontrivial, the fully privacy-preserving pipeline remains within a practical range for real-world authentication deployments that prioritize strong privacy guarantees. Table 3 and 4 present an ablation study of execution times across various deployment pipelines for the different operations as well as phases.

Table 3: End-to-end execution time for operations in SentinelTouch using LeNet-5 for different embedding dimensions. *ms* is milliseconds, *s* is seconds. *D* is the vector dimension.

Operation	Hybrid (ms)		Full-Privacy (s)	
	16D	32D	16D	32D
Registration	57	114	14	14
Deletion	36	70	0.036	0.070
Identification	284	542	16	16

Table 4: Detailed runtime of SentinelTouch using LeNet-5 for hybrid and full-privacy pipelines with different embedding dimensions.

Operation	Hybrid(ms)		Full-Privacy(s)	
	16D	32D	16D	32D
PCA Preprocessing	≈ 0	≈ 0	≈ 0	≈ 0
Encryption	4	8	0.02	0.04
Feature Extraction	≈ 0	≈ 0	14	14
Homomorphic Matching	278	542	1.4	2.6
Decryption & Interpretation	2	4	0.06	0.12
Total	284	554	15.5	16.76

Scalability in privacy-preserving fingerprint identification with HE can be achieved through two complementary approaches, both of which are compatible with SentinelTouch. The first approach involves increasing the size of the HE security parameters. Specifically, enlarging the ciphertext slot count enables the system to accommodate more users within a single ciphertext. However, this comes with a direct trade-off in performance, as doubling the number of slots approximately doubles the runtime while proportionally increasing storage overhead. This scaling method works well for moderate dataset sizes but becomes very expensive for extremely large databases, such as those exceeding 100,000 users, because of the significant increase in computation and memory requirements associated with the required HE parameters. The second approach, adopted by Blind-Touch and Blind-Match, introduces a distributed server architecture where multiple instances of the server process disjoint subsets of the encrypted dataset in parallel, and the results are securely aggregated. This distributed scaling strategy maintains privacy guarantees while improving throughput by leveraging parallel computation. SentinelTouch is compatible with both scaling paradigms; thus, the choice of pipeline for large-scale deployments should balance database size, available computation resources, and target system latency.

7.5 Memory Consumption

In addition to runtime performance, we evaluate the memory footprint of SentinelTouch across different pipeline configurations. Memory consumption is measured on the server side and includes all ciphertext storage, intermediate homomorphic evaluation buffers, and any temporary structures required for user registration, deletion, inference, and matching. These measurements reflect the peak memory usage during end-to-end execution, providing a realistic estimate for deployment requirements. The datasets do not impact the server's memory usage. Instead, it depends mainly on the deployment pipeline, the embedding feature vector size, and the HE security parameters needed to evaluate the encrypted model.

The hybrid pipeline consistently consumes less memory than the full-privacy pipeline, as the feature extraction operations are performed in plaintext and do not require additional ciphertext buffers and evaluation keys for the convolution and pooling layers. Nevertheless, the memory requirements remain within the capacity of consumer-grade systems even for the most expensive setting, using a maximum of 5.2GB of RAM. Table 5 summarizes the observed peak memory usage across our models and pipeline settings.

Table 5: Peak memory consumption (GB) of SentinelTouch for hybrid and full-privacy pipelines with different embedding dimensions.

Pipeline	Hybrid (GB)		Full-Privacy (GB)	
	16D	32D	16D	32D
Memory Usage	0.8	1.1	4.1	5.2

7.6 Comparison with Related Works

To contextualize the performance of SentinelTouch, we compare our results with existing approaches in encrypted fingerprint matching. To the best of our knowledge, SentinelTouch represents the first full-privacy pipeline for fingerprint identification and matching using neural networks. In the full-privacy setting, SentinelTouch outperforms state-of-the-art methods by approximately 10× in end-to-end identification latency, as summarized in Table 6.

In the hybrid setting, we compare SentinelTouch against prominent fingerprint recognition solutions, including Blind-Match, Blind-Touch, and DeepPrint. Our results show that SentinelTouch achieves up to 2.5× faster processing time for end-to-end, one-to-many matching tasks compared to Blind-Match using 16-dimensional embeddings, as reported in Table 7. These efficiency gains are achieved without sacrificing accuracy, demonstrating that SentinelTouch provides an optimal balance of privacy, latency, and accuracy.

Table 6: Comparison of end-to-end latency (in seconds) for full-privacy biometric fingerprint identification systems.

System	method	Latency (s)
Kim et al. [21]	Euclidean Distance	162
Pradel et al. [32]	Euclidean Distance	3133536
Yang et al. [41]	Hamming Distance	252
SentinelTouch	LeNet-5	16

Processor Summary:
Hardware used:
Kim et al.: Intel Core i5-7500 CPU @ 3.40GHz ×4, 64-bit Ubuntu 16.04 LTS.
Pradel et al.: Intel Core i3-6100 CPU @ 3.70GHz ×4, 8GB RAM.
Yang et al.: AMD FX-8370 Eight-Core Processor @ 4.01GHz, 24GB RAM.
SentinelTouch: Intel Core i7-14700K @ 3.40GHz, 20 cores, 32GB RAM.

Table 7: Comparison of end-to-end identification latency for hybrid fingerprint matching systems and their one-to-many rank-1 accuracy on the PolyU dataset. Acc.: Accuracy

System	Model	Parameters	Latency (ms)	Rank-1 Acc.
Blind-Touch	Custom Arch.	$\approx 2.0M$	650	59.17%
Blind-Match	ResNet-18	$\approx 11.2M$	720	99.55%
DeepPrint	Custom Arch.	$\approx 67.4M$	3,400	–
SentinelTouch	LeNet-5	$\approx 44k$	284	99.51%

Processor Summary:
Hardware used:
DeepPrint: Intel Core i9-7900X @ 3.30 GHz, 32GB RAM
Blind-Match and Blind-Touch: Five NAVER Cloud standard-g2 instances, Intel Xeon Gold 5220 @ 2.20 GHz, 8GB RAM each
SentinelTouch: Intel Core i7-14700K @ 3.4 GHz, 20 cores, 32GB RAM

To evaluate our hypothesis under realistic hardware constraints, we performed inference experiments on a Raspberry Pi 3 Model B+ equipped with a 1.4 GHz 64-bit quad-core processor. The ResNet-18 model used in Blind-Match, comprising approximately 11.2 million parameters, required an average of 13.9 seconds to process a single image. In contrast, our PCA + LeNet-5 pipeline achieved substantially lower latency: PCA construction and image resizing took only 0.001 seconds, while inference on the LeNet-5 model with about 44,000 parameters required 0.0131 seconds, resulting in a total pre-encryption runtime of 0.0132 seconds. This clearly confirms our hypothesis, showing that our hybrid PCA + LeNet-5 approach remains far more efficient than large neural network models such as the ResNet-18. Moreover, the drastic reduction in model size and computation cost makes our design highly suitable for deployment on most resource-constrained devices. These results highlight the practicality of lightweight models in achieving efficient and scalable privacy-preserving fingerprint image processing on the edge.

8 Conclusion

In this work, we present SentinelTouch, a novel system for privacy-preserving fingerprint identification and authentication that leverages homomorphic encryption and neural networks. SentinelTouch supports both full-privacy and hybrid pipelines, enabling strong privacy guarantees while maintaining practical performance for real-world deployment. We introduce a preprocessing stage that reduces the fingerprint image dimensions to as low as 28x28 for efficient neural network processing. We then develop a lightweight neural network that provides high Rank-1 accuracy in multiple one-to-many identification tasks. Through extensive evaluation on the PolyU and SOKOTO datasets, we demonstrated that SentinelTouch achieves high recognition accuracy across different embedding feature vector dimensions. In full privacy-preserving mode, our baseline LeNet-5 model outperforms existing solutions by up to 10 times in latency, while in hybrid configurations, it achieves up to 2.5 times the improvement. These results establish SentinelTouch as the first full-privacy-preserving, practical, and high-performance neural network system for identification and authentication on biometric fingerprint images.

In future work, we plan to extend SentinelTouch to support additional biometric modalities and multi-factor authentication. We also aim to deploy SentinelTouch on dedicated fingerprint-scanner hardware and evaluate its suitability in real-world authentication and identification settings. Overall, SentinelTouch shows that strong privacy, high accuracy, and real-time performance can indeed coexist in biometric fingerprint verification, providing a robust foundation for next-generation secure biometric identification systems.

References

[1] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.
[2] Ahmad Al Badawi, Jack Bates, Flavio Bergamaschi, David Bruce Cousins, Saroja Erabelli, Nicholas Genise, Shai Halevi, Hamish Hunt, Andrey Kim, Yongwoo Lee, Zeyu Liu, Daniele Micciancio, Ian Quah, Yuriy Polyakov, Saraswathy R.V., Kurt Rohloff, Jonathan Saylor, Dmitriy Suponitsky, Matthew Triplett, Vinod Vaikuntanathan, and Vincent Zucca. Openfhe: Open-source fully homomorphic encryption library. Cryptology ePrint Archive, Paper 2022/915, 2022. https://eprint.iacr.org/2022/915.
[3] Mauro Barni, Tiziano Bianchi, Dario Catalano, Mario Di Raimondo, Ruggero Donida Labati, Pierluigi Failla, Dario Fiore, Riccardo Lazzeretti, Vincenzo

Piuri, Alessandro Piva, et al. A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates. In *2010 fourth IEEE international conference on biometrics: theory, applications and systems (BTAS)*, pages 1–7. IEEE, 2010.

[4] Debnath Bhattacharyya, Rahul Ranjan, Farkhod Alisherov, Minkyu Choi, et al. Biometric authentication: A review. *International Journal of u-and e-Service, Science and Technology*, 2(3):13–28, 2009.

[5] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

[6] Leiyu Chen, Shaobo Li, Qiang Bai, Jing Yang, Sanlong Jiang, and Yanming Miao. Review of image classification algorithms based on convolutional neural networks. *Remote Sensing*, 13(22):4712, 2021.

[7] Jung Hee Cheon, Kyoohyung Han, Andrey Kim, Miran Kim, and Yongsoo Song. A full RNS variant of approximate homomorphic encryption. Cryptology ePrint Archive, Paper 2018/931, 2018.

[8] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *International conference on the theory and application of cryptology and information security*, pages 409–437. Springer, 2017.

[9] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Tfhe: fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1):34–91, 2020.

[10] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Tfhe: Fast fully homomorphic encryption over the torus. Cryptology ePrint Archive, Paper 2018/421, 2018. https://eprint.iacr.org/2018/421.

[11] Hyunmin Choi, Jiwon Kim, Chiyoung Song, Simon S Woo, and Hyoungshick Kim. Blind-match: efficient homomorphic encryption-based 1: n matching for privacy-preserving biometric identification. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4423–4430, 2024.

[12] Hyunmin Choi, Simon S Woo, and Hyoungshick Kim. Blind-touch: Homomorphic encryption-based distributed neural network inference for privacy-preserving fingerprint authentication. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 21976–21985, 2024.

[13] A. Conklin, G. Dietrich, and D. Walz. Password-based authentication: a system perspective. In *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*, pages 10 pp.–, 2004.

[14] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019.

[15] Krishna Dharavath, Fazal A Talukdar, and Rabul H Laskar. Study on biometric authentication systems, challenges and future trends: A review. In *2013 IEEE international conference on computational intelligence and computing research*, pages 1–7. IEEE, 2013.

[16] Joshua J Engelsma, Kai Cao, and Anil K Jain. Learning a fixed-length fingerprint representation. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):1981–1997, 2019.

[17] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *Cryptology ePrint Archive*, 2012.

[18] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.

[19] Stephanie Gootman. Opm hack: The most dangerous threat to the federal government today. *Journal of Applied Security Research*, 11(4):517–525, 2016.

[20] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. {GAZELLE}: A low latency framework for secure neural network inference. In *27th USENIX security symposium (USENIX security 18)*, pages 1651–1669, 2018.

[21] Taeyun Kim, Yongwoo Oh, and Hyoungshick Kim. Efficient privacy-preserving fingerprint-based authentication system using fully homomorphic encryption. *Security and Communication Networks*, 2020(1):4195852, 2020.

[22] A Kumar. The hong kong polytechnic university 3d fingerprint images database version 2.0,", 2015.

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

[24] Chenhao Lin and Ajay Kumar. Matching contactless and contact-based conventional fingerprint images for biometrics identification. *IEEE Transactions on Image Processing*, 27(4):2008–2021, 2018.

[25] Hou-I Liu, Marco Galindo, Hongxia Xie, Lai-Kuan Wong, Hong-Han Shuai, Yung-Hui Li, and Wen-Huang Cheng. Lightweight deep learning for resource-constrained environments: A survey. *ACM Computing Surveys*, 56(10):1–42, 2024.

[26] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017.

[27] Lynette I Millett and Joseph N Pato. Biometric recognition: Challenges and opportunities. 2010.

[28] Nazri Mohd Nawi, Walid Hasen Atomi, and Mohammad Zubair Rehman. The effect of data pre-processing on optimized training of artificial neural networks. *Procedia Technology*, 11:32–39, 2013.

[29] Nges Brian Njungle, Eric Jahns, and Michel A Kinsy. Fheon: A configurable framework for developing privacy-preserving neural networks using homomorphic encryption. *arXiv preprint arXiv:2510.03996*, 2025.

[30] Nges Brian Njungle and Michel A Kinsy. Activate me!: Designing efficient activation functions for privacy-preserving machine learning with fully homomorphic encryption. In *International Conference on Cryptology in Africa*, pages 51–73. Springer, 2025.

[31] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pages 223–238. Springer, 1999.

[32] Gaëtan Pradel and Chris Mitchell. Privacy-preserving biometric matching using homomorphic encryption. *arXiv preprint arXiv:2111.12372*, 2021.

[33] Ronald L Rivest, Len Adleman, Michael L Dertouzos, et al. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[34] Lorenzo Rovida and Alberto Leporati. Encrypted image classification with low memory footprint using fully homomorphic encryption. Cryptology ePrint Archive, Paper 2024/460, 2024.

[35] Zhang Rui and Zheng Yan. A survey on biometric authentication: Toward secure and privacy-preserving identification. *IEEE Access*, 7:5994–6009, 2019.

[36] Yahaya Isah Shehu, Ariel Ruiz-Garcia, Vasile Palade, and Anne James. Sokoto coventry fingerprint dataset, 2018.

[37] S. Sreehari and S.M. Anzar. Touchless fingerprint recognition: A survey of recent developments and challenges. *Computers and Electrical Engineering*, 122:109894, 2025.

[38] U. Sumalatha, K. Krishna Prakasha, Srikanth Prabhu, and Vinod C. Nayak. Touch of privacy: A homomorphic encryption-powered deep learning framework for fingerprint authentication. *IEEE Access*, 13:59057–59073, 2025.

[39] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018.

[40] W Yang, S Wang, J Hu, G Zheng, and C Valli. Security and accuracy of fingerprint-based biometrics: A review. symmetry, 11 (2), 141, 2019.

[41] Wencheng Yang, Song Wang, Kan Yu, James Jin Kang, and Michael N Johnstone. Secure fingerprint authentication with homomorphic encryption. In *2020 Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–6. IEEE, 2020.

[42] Tjalling J Ypma. Historical development of the newton–raphson method. *SIAM review*, 37(4):531–551, 1995.

[43] Carlos Vladimiro González Zelaya. Towards explaining the effects of data preprocessing on machine learning. In *2019 IEEE 35th international conference on data engineering (ICDE)*, pages 2086–2090. IEEE, 2019.