HEFT: Homomorphically Encrypted Fusion of Biometric Templates

Luke Sperling[†] Nalini Ratha[‡] Arun Ross[†] Vishnu Naresh Boddeti[†]
[†]Michigan State University [‡]University at Buffalo

Abstract

This paper proposes a non-interactive end-to-end solution for secure fusion and matching of biometric templates using fully homomorphic encryption (FHE). Given a pair of encrypted feature vectors, we perform the following ciphertext operations, i) feature concatenation, ii) fusion and dimensionality reduction through a learned linear projection, iii) scale normalization to unit ℓ_2-norm, and iv) match score computation. Our method, dubbed HEFT (Homomorphically Encrypted Fusion of biometric Templates), is custom-designed to overcome the unique constraint imposed by FHE, namely the lack of support for non-arithmetic operations. From an inference perspective, we systematically explore different data packing schemes for computationally efficient linear projection and introduce a polynomial approximation for scale normalization. From a training perspective, we introduce an FHE-aware algorithm for learning the linear projection matrix to mitigate errors induced by approximate normalization. Experimental evaluation for template fusion and matching of face and voice biometrics shows that HEFT (i) improves biometric verification performance by 11.07% and 9.58% AUROC compared to the respective unibiometric representations while compressing the feature vectors by a factor of 16 (512D to 32D), and (ii) fuses a pair of encrypted feature vectors and computes its match score against a gallery of size 1024 in 884 ms. Code and data are available at https://github.com/human-analysis/encrypted-biometric-fusion

1. Introduction

Feature-level fusion is a commonly employed technique in multi-biometric recognition systems, especially in large-scale deployments. Template fusion helps to overcome the limitations of unibiometric systems in terms of improving recognition performance and population coverage. However, utilizing multiple biometric signatures also enhances the security risks associated with attacks on such systems. In fact, there is growing evidence that the templates contain sufficient information to either reconstruct the raw biometric signature [30] or leak sensitive soft-biometric information [29]. Thus, it is imperative to devise template fusion and matching schemes that secure the biometric signatures of users across all modalities and help protect their privacy. Realizing this goal is the primary focus of this paper.

Cryptosystems based on Fully Homomorphic Encryption [20] (FHE) are an attractive solution for protecting biometric templates through encryption. FHE schemes such as BFV [10, 19] and CKKS [13], theoretically, allow for computations directly on encrypted data without the need for decryption. Recent work [9, 18] has demonstrated that FHE is exceptionally effective and scalable for securing biometric templates, allowing for encrypted matching and search against a gallery of 100 Million.

Template-level fusion and matching typically involve the following operations: feature concatenation, linear/non-linear projection, scale normalization of resulting feature, and finally matching score computation. Operations in existing approaches for feature-level fusion are all presumed to be performed in plaintext (unencrypted domain) and therefore run into limitations when performed on ciphertext (encrypted domain). For example, non-arithmetic operations, such as division and square root required for scale normalization, are not supported by FHE schemes for direct computation on ciphertexts. Furthermore, operations on ciphertext are significantly more computationally expensive, both in terms of latency and memory requirements, than the same operations on the corresponding plaintext.

To overcome the aforementioned limitations, we propose HEFT, a biometric template fusion and matching scheme that operates directly on encrypted templates. Given a pair of encrypted templates, HEFT performs the following ciphertext operations: feature concatenation, projection, scale normalization to unit ℓ_2-ball, and matching score computation. This process is illustrated in Fig. 1.

From an *inference perspective*, the salient features of HEFT include, i) fusing unibiometric templates of different dimensionalities, ii) fusion and compression of concatenated templates through linear projection to ease the steep computational burden of downstream ciphertext matching operations, and iii) approximating the non-arithmetic ℓ_2 normalization operation through composite polynomials. From a *learning perspective*, we introduce FHE-aware

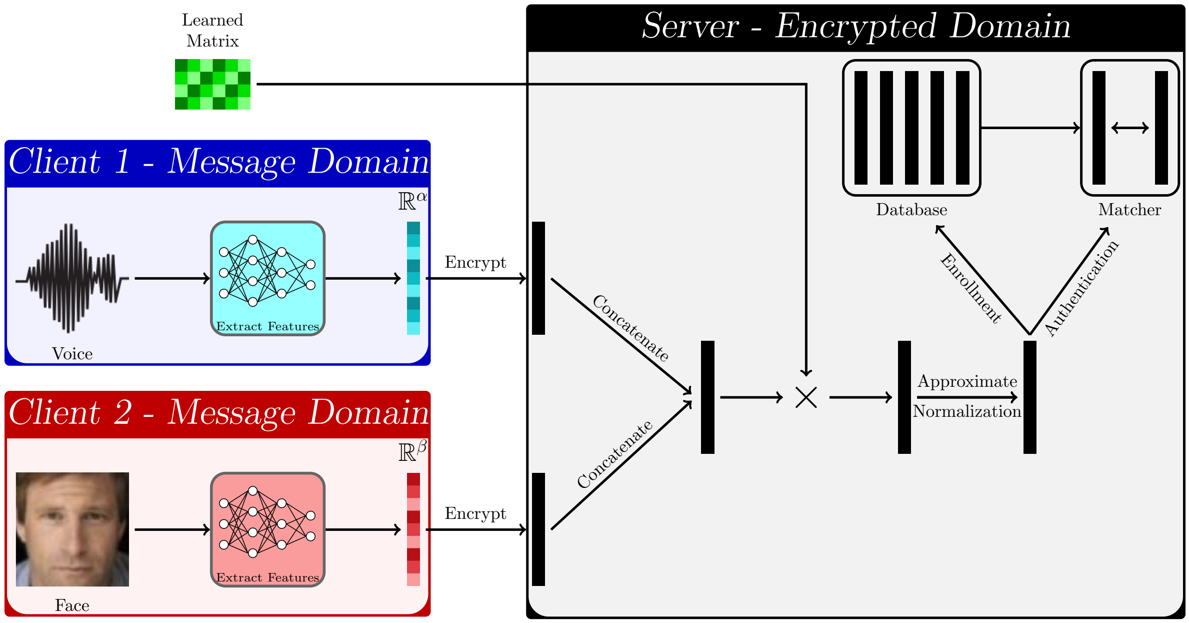


Figure 1: **Overview:** End-to-end biometric template fusion and matching using fully homomorphic encryption (FHE). Given feature representations extracted from two different modalities of an individual, the client encrypts and transmits the features to our system. We concatenate the two encrypted vectors and perform a matrix-vector multiplication with a learned plaintext projection matrix. The resulting ciphertext represents the fused encrypted vector. We normalize the encrypted vector using an approximation to overcome the constraints imposed by FHE. During enrollment, this template is stored in the database of encrypted templates. During authentication, match scores are computed between the probe and templates from the encrypted database and sent to the client for decryption and further processing.

learning of fusion model parameters to mitigate performance loss from approximating the normalization process.

From a *practical perspective*, we carefully analyzed the effect of various design choices on the trade-off between accuracy and efficiency (memory and latency) of biometric fusion and match score computation. These include data encoding schemes, matrix-multiplication methods, and approximation schemes for normalization. Through our analysis, we identify the optimal options (in terms of memory and latency) under both small-scale and large-scale settings, w.r.t. feature dimension and gallery size.

In summary, we present the first practically feasible homomorphic multibiometric feature-level fusion and matching algorithm. Experimental evaluation on a combination of encrypted face and voice biometric signatures demonstrates appreciable gains in matching performance over the unibiometric counterparts while taking 884 ms to fuse a pair of biometric templates and compute match scores against a gallery of size 1024.

2. Related Work

Privacy-Preservation in Biometrics: Many methods have been devised over the years to secure biometric templates and preserve user privacy. Early biometric cryptosystems based on image processing [41, 42] and fuzzy vaults [23] were employed for protecting both iris [26] and fingerprint [46] data. Such systems, however, suffered from a loss in matching performance. Cryptosystems such as Goldwasser-Micali encryption have also been used for authentication scenarios [11], but they do not protect the templates at matching and are, therefore, vulnerable to attacks.

Homomorphic encryption (HE) is an attractive option for privacy-preserving biometrics applications due to its ability to enable computations on encrypted data without the need to decrypt. Early biometric systems driven by HE were based on partially homomorphic encryption (PHE) schemes [21]. They were applied to numerous biometric modalities [6], including face recognition [45], iris recognition [47, 48, 8] and fingerprint recognition [5]. The opportunity to design robust biometrics cryptosystems came to the fore with the development of the first fully homomorphic encryption (FHE) scheme [20]. Since then, there have been many application scenarios for biometrics exploiting the privacy afforded by FHE without substantial performance drawbacks. Gomez-Barrero et al. [22] developed a general framework for template-level fusion based on homomorphic encryption. This framework relies on performing fusion before encryption and does not support template fusion directly in the encrypted domain. Boddeti [9] demonstrated the ability to match face templates in the encrypted domain. Engelsma et al. [18] proposed an efficient way to search encrypted templates by combining a novel encoding scheme with feature compression. By using a tree search structure created by fusing similar templates, Drozdowski et al. [16] developed a method for faster biometric indexing and retrieval. In contrast to this body of work, in this paper, we leverage fully homomorphic encryption for end-to-end template fusion and match score computation and devise an FHE-aware learning algorithm for feature projection.

Feature-Level Biometric Fusion: Fusion at the feature-level leverages information from multiple templates to improve performance. Early techniques focused on selecting features from each template to be fused [35]. Sarangi et al. [36] combined face and ear templates by concatenating templates compressed through classical dimensionality reduction techniques. Feature-level fusion has also been performed on face, fingerprint, and finger vein modalities [50]. Coupled mapping techniques have been devised to match samples between domains, with a maximum-margin approach [37] and with a marginal fisher analysis approach [38]. Lately, learning-based approaches have been used. Silva et al. [39] performed feature selection using Particle Swarm Optimization. Tiong et al. [44] proposed a method of information fusion via extracting features from raw biometric data using a CNN and then combining them with a series of fully connected layers. Other deep learning approaches have been proposed recently [7, 51, 2, 27, 43]. Contrasting these methods, we opt for a linear projection-based approach to limit the multiplicative depth of the circuit and decrease computational complexity, which is important for creating a practical solution in FHE.

3. Approach

We propose HEFT for template fusion and matching. It is designed for maximizing performance and efficiency at inference over ciphertexts. Given an encrypted multi-biometric dataset, i.e., a pair of encrypted feature vector matrices, HEFT performs the following series of ciphertext operations, (i) *concatenation* of the feature vectors, (ii) *linear projection* using a learned matrix to a new lower-dimensional feature space, (iii) *approximate normalization* of the features by projecting them onto a unit ℓ_2-ball for fast match score computation, and (iv) *match score computation* of the fused features against an encrypted gallery of fused features. Finally, to compensate for the errors[1] induced by approximate normalization at inference, we propose an FHE-aware training process that takes the approximate normalization into account.

3.1. Problem Setup

Biometric Fusion: Consider a multibiometric system that comprises n features vectors from two sources $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n] \in \mathcal{R}^{\alpha \times n}$ and $\boldsymbol{Y} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_n] \in \mathcal{R}^{\beta \times n}$. These features are fused into a new space $\boldsymbol{Z} = [\boldsymbol{z}_1, \dots, \boldsymbol{z}_n] \in \mathcal{R}^{\gamma \times n}$. While we restrict ourselves to fusing a pair of biometric features, our solution can readily be applied to the fusion of a multitude of biometric features.

In this paper, we consider a linear projection operation to fuse the feature vectors, i.e., $\boldsymbol{Z} = \boldsymbol{P}\tilde{\boldsymbol{X}}$, where $\tilde{\boldsymbol{X}} = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_n \\ \boldsymbol{y}_1 & \boldsymbol{y}_2 & \cdots & \boldsymbol{y}_n \end{bmatrix} \in \mathcal{R}^{\delta \times n}$ is a matrix of concatenated features and $\boldsymbol{P} \in \mathcal{R}^{\gamma \times \delta}$ is the projection matrix that maps into a common γ dimensional space, and $\delta = \alpha + \beta$.

The fused templates can be used for any downstream tasks, such as matching. A common metric that is adopted for template matching is the cosine similarity $d(\boldsymbol{x}, \boldsymbol{y}) = 1 - \frac{\boldsymbol{x}^T\boldsymbol{y}}{\|\boldsymbol{x}\|_2\|\boldsymbol{y}\|_2} = 1 - \tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{y}}$ where $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$ are scale normalized versions of $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, and are obtained by projecting $\boldsymbol{x}$ and $\boldsymbol{y}$ onto the unit ℓ_2-ball.

Secure Biometric Fusion: Our goal in this work is to devise a cryptographic solution to secure the multibiometric templates and prevent unauthorized access to any private user information during the template fusion process, as well as any desired downstream tasks. This can be achieved through a parameterized function that transforms the multibiometric features $(\boldsymbol{x}, \boldsymbol{y})$ into an alternate space $(\mathcal{E}(\boldsymbol{x}), \mathcal{E}(\boldsymbol{y}))$ such that $\mathcal{E}(\boldsymbol{x}) = f(\boldsymbol{x}; \boldsymbol{\theta}_{pk})$, $\boldsymbol{x} = g(\mathcal{E}(\boldsymbol{x}); \boldsymbol{\theta}_{sk})$ are encryption and decryption functions with $\boldsymbol{\theta}_{pk}$ and $\boldsymbol{\theta}_{sk}$ being the public and secret keys respectively. By executing all the fusion operations, namely, *concatenation*, *projection*, *normalization* and *match score computation* directly over the ciphertexts, i.e., without decrypting them, we can prevent unauthorized access to sensitive information, and hence preserve user privacy.

Fully homomorphic encryption (FHE) is a class of encryption algorithms that allows arithmetic computations directly over ciphertexts and is ideally suited to realize our goal. Even if a malicious attacker can gain access to the multibiometric features during any part of the fusion or matching process, without access to the secret key $\boldsymbol{\theta}_{sk}$ the attacker cannot reconstruct the underlying biometric sample or extract any other information present in the features.

3.2. Protocols: Template Fusion and Authentication

We use the Cheon-Kim-Kim-Song (CKKS) scheme [13] as the underlying FHE scheme for template fusion and match score computation. We first give an overview of this scheme and describe the enrollment and authentication protocols for template fusion next.

The **CKKS encryption scheme** allows operations over encrypted vectors of complex numbers [13]. Its mathematical basis lies in modular arithmetic over polynomial rings, and its security lies in the hardness of the Ring Learning with Errors problem. CKKS offers post-quantum security for an appropriate choice of encryption parameters [3].

[1] leads to performance degradation in downstream tasks like matching.

Plaintexts are polynomials within the polynomial ring $R = \mathbb{Z}[x]/(x^N + 1)$. Therefore, complex vectors $C^{N/2}$ must be encoded into this space to perform encryption. After encoding, the plaintext polynomial is encrypted via a secret key into a set of two polynomials, $R_q^2 = \mathbb{Z}_q[x]/(x^N + 1)$ where R_q denotes polynomials of coefficients modulo q and degree less than N. This will serve as the ciphertext.

CKKS has three keys, a secret key sk, a public key pk, and an evaluation key evk for homomorphic multiplication. Its protocol comprises the following functions, i) *Key Generation:* Generates the keys, ii) *Encryption:* Given a plaintext polynomial and the public key, output two polynomials representing the ciphertext, iii) *Decryption:* Given a ciphertext comprised of two polynomials, apply the secret key and retrieve a plaintext polynomial, iv) *Addition:* A simple sum of the ciphertexts translates to homomorphic addition, v) *Multiplication:* Multiplication of ciphertexts is polynomial multiplication which results in three polynomials. To restrict the size of resultant ciphertexts, relinearization is needed, vi) *Relinearization:* Given three polynomials representing a ciphertext product, the evaluation key is used to reduce the size of the ciphertext from three to two polynomials, and vii) *Rotation:* Ciphertexts may be cyclically rotated using an optionally generated set of Galois keys.

Encrypted Template Fusion Protocol at Enrollment: Consider two sets of biometric templates $\boldsymbol{X} \in \mathcal{R}^{\alpha\times n}$ and $\boldsymbol{Y} \in \mathcal{R}^{\beta\times n}$ that we seek to fuse along with their class labels $\boldsymbol{I} \in \mathcal{Z}^n$. Each set of templates are encrypted using the data encoding scheme requested by the cloud server. After receiving the encrypted templates, the cloud server performs the following operations: i) for each class label c, create all pairs of templates $\{(\boldsymbol{x}_i, \boldsymbol{y}_j)|\forall(i,j) \in \boldsymbol{I}_c \times \boldsymbol{I}_c, \boldsymbol{I_c} \subseteq \boldsymbol{I}\}$, where $\boldsymbol{I}_c$ are the indices of samples belonging to class c, ii) fuse the pairs of templates created, i.e., *concatenation*, *projection* and *normalization*, and iii) add the fused templates to the current gallery $\boldsymbol{G}$.

Encrypted Template Fusion Protocol at Authentication: A client sends a sample of encrypted multibiometric templates $\boldsymbol{x} \in \mathcal{R}^\alpha$ and $\boldsymbol{y} \in \mathcal{R}^\beta$. This pair of templates is fused, i.e., *concatenation*, *projection* and *normalization* to create a probe template $\boldsymbol{z} \in \mathcal{R}^\gamma$. For identification, i.e., $1 : N$ comparisons, match score (e.g., cosine similarity) is computed between the probe and the entire gallery $\boldsymbol{G}$. For verification with a claimed identity c i.e., $1 : 1$ comparison, match score (e.g., cosine similarity) is computed between the probe and the samples in the gallery $\boldsymbol{G}$ corresponding to the identity c. The encrypted scores are sent back to the client for decryption and further processing.

3.3. Encrypted Template Fusion and Matching

We now describe the various components of template fusion and match score computation. This includes (i) choice of the data encoding scheme, (ii) concatenating two ciphertexts, (iii) efficient ciphertext matrix-plaintext matrix multiplication for linear projection, and (iv) efficient and accurate approximate normalization.

3.3.1 Input Encoding and Vector Packing

Input Encoding: Before any computation can be performed on encrypted data, an encoding scheme must be selected to enable encryption and arithmetic operations on the resulting ciphertext. The efficiency of ciphertext operations is critically dependent on the encoding scheme chosen to represent features. As such, outline two different encoding schemes for the feature vectors, each of which is better suited for operating either at a small or large scale. *Dense:* Encodes each feature vector as a plaintext before encryption, thereby resulting in n ciphertexts. *SIMD:* Encodes each dimension of the feature vector as a plaintext before encryption, resulting in δ ciphertexts.

Vector Packing: FHE schemes such as CKKS support arithmetic operations directly on vectors by packing multiple numbers into different slots within a single polynomial. And, in most practical applications, the dimensionality of feature vectors is much less than the number of available polynomial slots. In such cases, multiple feature vectors can be batched into a single polynomial. The batching allows for SIMD (single instruction multiple data) operations and helps amortize runtime across multiple feature vectors.

Suppose we wish to encode n vectors into polynomials with m slots each. In the dense encoding scheme, $\lceil \frac{n}{\lfloor \frac{m}{\delta} \rfloor} \rceil$ many polynomials are needed if rotation operations are not needed. However, ciphertext template fusion requires rotation operations. So, we pack an extra copy of each vector to simulate the "wrapping" effect of rotation. Therefore, $\lceil \frac{n}{\lfloor \frac{m}{2\delta} \rfloor} \rceil$ polynomials are needed. In the SIMD encoding scheme, a single dimension of the n vectors can be packed into a single polynomial. In this scheme, $\delta\lceil \frac{n}{m} \rceil$ polynomials are needed to represent n δ-dimensional vectors.

3.3.2 Concatenating Ciphertexts

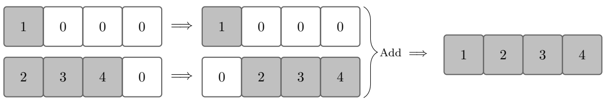


Figure 2: Ciphertext concatenation via rotation and addition for the dense encoding scheme. The second ciphertext (bottom) is right-rotated α slots and added to the first ciphertext (top).

The concatenation mechanism depends on our choice of data encoding scheme. *Dense:* In this case, each vector in the multibiometric dataset $(\boldsymbol{X}, \boldsymbol{Y})$ is zero-padded before encryption to a dimensionality of δ. Now, concatenation can be done in the encrypted domain by right-rotating each ciphertext in $\boldsymbol{Y}$ by α slots and adding to the corresponding ciphertext in $\boldsymbol{X}$. *SIMD:* As each dimension of the query is packed into a single ciphertext, there is no need to concatenate the features. Instead, simply storing the ciphertexts in a single ordered array is sufficient in this representation.

3.3.3 Encrypted Linear Projection

Executing fusion through linear projection requires a matrix-matrix multiplication. Since we learn our projection matrix in the unencrypted domain, the multiplication is a plaintext-ciphertext multiplication, which is considerably more efficient than a ciphertext-ciphertext multiplication. Next, we outline two matrix-vector multiplication techniques, one that is better suited for small-scale datasets and the other for large-scale datasets. However, due to our ciphertext packing scheme, these methods functionally become matrix-matrix algorithms and can be treated as such. Furthermore, we note that it is desirable for the fused representations to be as compact as possible, i.e., γ should be small to ease the computational burden of any downstream tasks that are performed directly on the ciphertexts. Hence, the projection matrix $\boldsymbol{P} \in \mathcal{R}^{\gamma \times \delta}$ is rectangular.

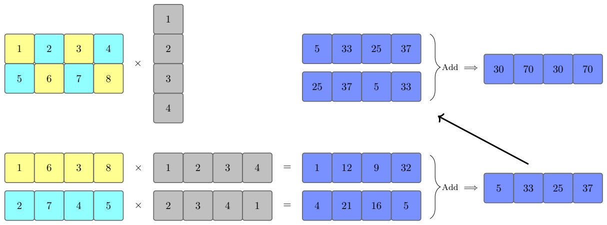


Figure 3: **Hybrid:** The efficiency of matrix-vector multiplications can be improved through a diagonal encoding scheme for the projection matrix ($\boldsymbol{P}$). The query is rotated once and multiplied with each diagonally encoded component of $\boldsymbol{P}$. The sum of these results is rotated and added with itself to obtain the final output.

Hybrid: When the query vectors are encoded using the dense scheme, the projection matrix can be encoded through a diagonal encoding scheme for efficient matrix-vector products. This scheme, shown in Fig. 3, was introduced by Juvekar et al. [24] and is specialized for short and wide rectangular matrices, i.e., $\gamma < \delta$. These diagonals are multiplied by rotated versions of the query vector, and the resultant vectors can simply be additively combined to yield the desired matrix-vector multiplication result. This method is best suited for cases where n is small.

SIMD: When the query vectors are encoded using the SIMD scheme, the projection matrix can also be in a repeated SIMD manner to support scalable matrix-vector products for large n. The scheme, shown in Fig. 4, was adopted by Engelmsa [18] for scaling search over an encrypted database. This method takes $\gamma\delta$ plaintext-ciphertext multiplications for a single matrix-vector multiplication but admits greater ciphertext packing potential, making it a computationally more efficient solution when $n >> \gamma\delta$. This method also negates the need for any expensive ciphertext rotations. The SIMD scheme, however, is more memory intensive due to the need for loading many plaintexts and ciphertexts in memory as seen in Fig. 6c.

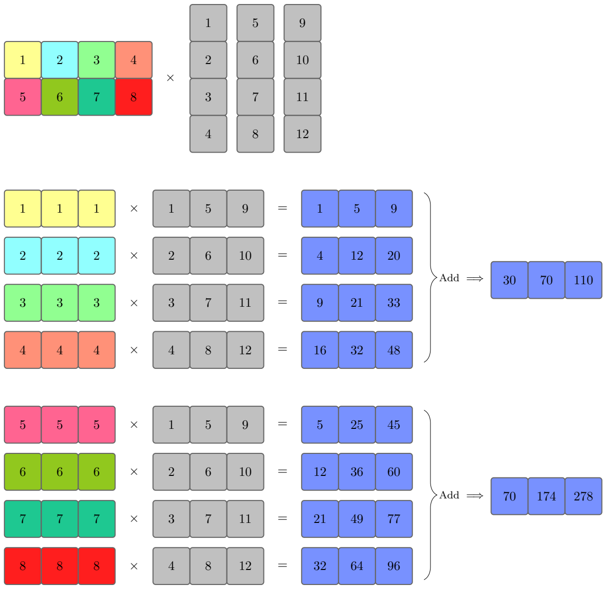


Figure 4: **SIMD:** This method repeats and encodes each element of the projection matrix as plaintext and multiplies with SIMD encoded query vectors. The result is a single ciphertext for each dimension of the result. This method is best suited for large n.

3.3.4 Approximate Normalization

Recent biometric representations (e.g., DeepPrint [17], ArcFace [15]) are typically projected to the surface of a unit ℓ_2-ball[2]. Formally, $\hat{u} = \frac{u}{||u||_2}$ where $||u||_2 = \sqrt{\sum_{i=1}^{d} u_i^2}$ for $u \in \mathbb{R}^d$. This normalization allows for computing cosine similarity simply through a dot-product between a pair of vectors. Such a normalization operation, however, cannot be performed directly on the ciphertexts since FHE schemes do not support non-arithmetic operations such as square root and division in the encrypted domain. Although it is possible to approximate each of these operations individually [14, 4], the computational efficiency can be significantly improved by directly approximating the inverse square root operation. Panda [33] showed it is possible to approximate inverse square root through the iterative Goldschmidt's Algorithm [12, 31] but is impractical for our purposes due to its high multiplicative depth.

We adopt a composite polynomial of the form $f(x) =$

[2] Other norms like ℓ_1 or ℓ_∞ can also be supported by HEFT if desired.

Encoding	Operation	Time Complexity					Space Complexity
		Additions	Plain-Cipher Mult.	Cipher-Cipher Mult.	Mult. Depth	Rotations	
Dense	Concatenation	$\lceil \frac{n}{l} \rceil$	0	0	0	$\lceil \frac{n}{l} \rceil$	$O(p\lceil \frac{n}{l} \rceil)$
	Projection	$(\gamma + log(\delta) - log(\gamma) - 2)\lceil \frac{n}{l} \rceil$	$\gamma\lceil \frac{n}{l} \rceil$	0	1	$(\gamma + log(\delta) - log(\gamma) - 1)\lceil \frac{n}{l} \rceil$	$O(\gamma p + p\lceil \frac{n}{l} \rceil)$
	Normalization	$log(\gamma)\lceil \frac{n}{l} \rceil$	$d\lceil \frac{n}{l} \rceil$	$2\lceil \frac{n}{l} \rceil$	$2 + d$	$log(\gamma)\lceil \frac{n}{l} \rceil$	$O(p\lceil \frac{n}{l} \rceil)$
	Preprocessing	$\lceil \frac{n}{l} \rceil - \lceil \frac{n\gamma}{m} \rceil$	$\lceil \frac{n}{l} \rceil$	0	1	$\lceil \frac{n}{l} \rceil - \lceil \frac{n\gamma}{m} \rceil$	$O(p\lceil \frac{n}{l} \rceil + p\lceil \frac{n\gamma}{m} \rceil)$
	Matching	$log(\gamma)\lceil \frac{n\gamma}{m} \rceil$	0	$\lceil \frac{n\gamma}{m} \rceil$	1	$log(\gamma)\lceil \frac{n\gamma}{m} \rceil$	$O(p\lceil \frac{n\gamma}{m} \rceil)$
SIMD	Concatenation	-	-	-	-	-	-
	Projection	$\gamma(\delta - 1)\lceil \frac{n}{m} \rceil$	$\delta\gamma\lceil \frac{n}{m} \rceil$	0	1	0	$O(\delta\gamma p + \gamma p\lceil \frac{n}{m} \rceil)$
	Normalization	$(\gamma - 1)\lceil \frac{n}{m} \rceil$	$d\lceil \frac{n}{m} \rceil$	$2\gamma\lceil \frac{n}{m} \rceil$	$2 + d$	0	$O(\gamma p\lceil \frac{n}{m} \rceil)$
	Preprocessing	-	-	-	-	-	-
	Matching	$(log(\gamma) - 1)\lceil \frac{n}{m} \rceil$	0	$\gamma\lceil \frac{n}{m} \rceil$	1	0	$O(\gamma p\lceil \frac{n}{m} \rceil)$

Table 1: Time and memory complexity comparison of the Dense and SIMD encoding schemes for template fusion and matching. A preprocessing step is used in the Dense encoding scheme to reduce the number of ciphertexts in the gallery to enable faster matching. γ is the output dimensionality of the resultant vector. δ is the dimensionality of the query vector. For m slots available in a single ciphertext, we define $l = \lfloor \frac{m}{2\delta} \rfloor$. Depending on the encoding scheme, to process n samples, we must perform each operation $\lceil \frac{n}{l} \rceil$ or $\lceil \frac{n}{m} \rceil$ times ($\lceil \frac{n\gamma}{m} \rceil$ times to perform matching in the Dense scheme). p denotes the amount of space a single ciphertext occupies in memory.

$(g_k \circ g_{k-1} \circ \cdots \circ g_1)(x)$, where each $g_i(x)$ is a low-degree polynomial, to approximate the inverse square root function in a desired interval of x i.e., $\frac{1}{\sqrt{x}} \approx f(x) \forall x \in [a, b]$[3]. The number of composite functions k and the degree of each g_i determine the homomorphic multiplicative depth of the operation. Higher degree polynomials offer a better approximation of this function, but also increase the multiplicative depth of the circuit. Hence, there is a trade-off between accuracy of the approximation and computational efficiency.

3.3.5 Computational Complexity

Table 1 shows an analytical comparison of the time and space complexity of the end-to-end pipeline for both the Dense and SIMD encoding schemes. We show the required number of atomic operations for each stage of the pipeline including concatenation, projection, normalization, preprocessing (that is necessary for matching), and matching.

3.4. FHE Aware Learning of Projection Matrix

Having described the inference process, we now turn our attention to learning the optimal linear projection matrix $\boldsymbol{P}$ for template fusion. We posit that $\boldsymbol{P}$ can be learned in the unencrypted domain using biometric templates that are already available and hence do not suffer from privacy concerns. And, once learned, it can be employed for fusing the templates from private data for inference.

The projection matrix should map vectors of the same class close together for a given distance metric, while those of different classes should be far apart. To realize this goal, we adapt the concept of the maximum-margin loss function introduced by Siena et al. [37] for learning $\boldsymbol{P}$. The loss function minimizes the distance between samples of the same class and uses a hinge loss on triplets of samples involving a similar and dissimilar pair. We build upon this concept and adapt it in several ways to satisfy the unique combination of constraints imposed by the multi-modal fusion of features from deep neural networks and those of normalization approximations induced by FHE computations at inference.

Firstly, we adapt the loss function for multimodal feature-level fusion. Specifically, unlike Siena et al. [37] who seek to learn a pair of projection matrices with Euclidean distance-based metric, we learn a single projection matrix with cosine similarity[4] based metric. Given a concatenated dataset $\tilde{\boldsymbol{X}}$, the loss function is defined as:

$$\begin{aligned}\mathcal{L} = \lambda \frac{\sum_M d(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j)}{|M|} + \\ (1 - \lambda) \frac{\sum_V [m + d(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j) - d(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_k)]_+}{|V|}\end{aligned} \tag{1}$$

where $d(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j) = 1 - \frac{(\boldsymbol{P}\tilde{\boldsymbol{x}}_i)^T(\boldsymbol{P}\tilde{\boldsymbol{x}}_j)}{\|\boldsymbol{P}\tilde{\boldsymbol{x}}_i\|\|\boldsymbol{P}\tilde{\boldsymbol{x}}_j\|}$, $[x]_+ = max(0, x)$, M is the set of all pairs in $\tilde{\boldsymbol{X}}$ belonging to the same class, V denotes the set of all triplets $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k)$ such that $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ belong to the same class and $(\boldsymbol{x}_i, \boldsymbol{x}_k)$ belong to different classes, λ is a hyperparameter that weighs the "push" and "pull" terms' influences on the loss, m is the margin hyperparameter that determines the desired margin of separation between samples belonging to the same class and those belonging to different classes. The margin hyperparameter used in the triplet hinge loss can appropriately take on any value in the range $\left[0, \frac{c}{c-1}\right]$ for c classes [49].

Secondly, we note that the loss function in (1) is defined with exact normalization, while at the inference stage HEFT can only perform approximate normalization as described in Sec. 3.3.4. For instance, Fig. 5 shows a comparison between the exact inverse square root function and polynomial approximations of degrees 2 and 6. The mismatch between the unencrypted training and encrypted inference objectives observed here leads to performance degradation,

[3] See supplementary material for discussion on choosing the interval.

[4] Note that HEFT can also optimize for Euclidean distance if desired.

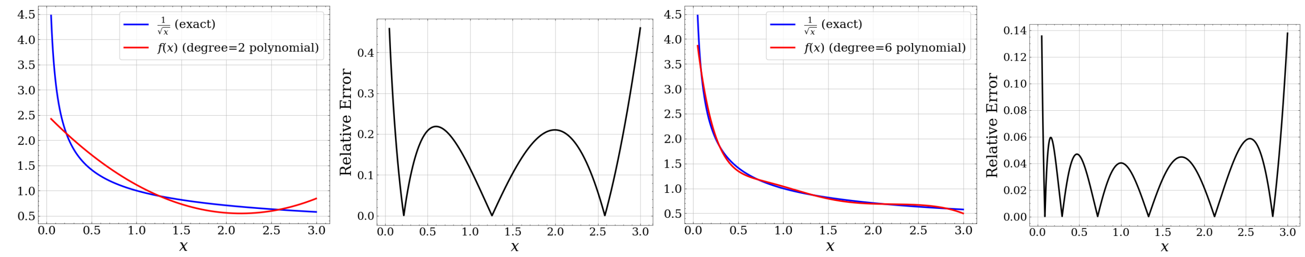


Figure 5: Polynomial approximations of inverse square root over the interval [0.05, 3.0] for polynomials of degree 2 and 6. Relative error $\left(\left|f(x) - \frac{1}{\sqrt{x}}\right| / \left|\frac{1}{\sqrt{x}}\right|\right)$ of the approximations are shown to the right of each plot.

as we demonstrate in Section 4. To mitigate this loss and recover the matching performance in the unencrypted domain, we incorporate the approximate normalization into the distance metric as,

$$d(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j) = 1 - (\boldsymbol{P}\tilde{\boldsymbol{x}}_i \odot f(\boldsymbol{P}\tilde{\boldsymbol{x}}_i))^T(\boldsymbol{P}\tilde{\boldsymbol{x}}_j \odot f(\boldsymbol{P}\tilde{\boldsymbol{x}}_j)) \quad (2)$$

where $\odot$ is the Hadamard product, and $f(\cdot)$ is the composite polynomial defined in Sec 3.3.4 and can be computed efficiently in the encrypted domain. Substituting the distance metric in (1) by (2) allows the learned projection matrix to compensate for the approximate normalization to a large extent, if not fully eliminate it.

4. Experiments

We evaluate the effectiveness of HEFT and analyze the effect of our design choices, both in terms of matching accuracy and computational complexity.

Implementation Details: To learn the projection matrix, we use the Adam [25] optimizer with a learning rate of 5×10^{-3}, a weight decay of 1×10^{-4} and train for 1000 epochs. Our encrypted inference is based on the CKKS scheme implemented in Microsoft's SEAL [1] library. Depending on the multiplicative depth of our approximate normalization method, we either use a polynomial modulus degree (N) of 16,384 or 32,768 along with a chain of very large prime numbers totaling 420, 580 or 860 bits as the coefficient modulus (q).

4.1. Evaluation Datasets

Google Speech Commands: This dataset comprises spoken single-word commands from many speakers. We use 5380 samples over 188 classes. We extract 512-dimensional feature vectors with the Deep Speaker [28] model, which is trained on the train-clean-360 portion of the LibriSpeech [32] dataset using a publicly available implementation.

CPLFW [52]: This benchmark face dataset is a harder version of LFW that incorporates cross-posed faces. We extract 512-dimensional feature vectors from a pre-trained VGG16 model trained on VGGFace[40, 34].

We pair 2 samples of 188 identities from CPLFW with those in the Google Speech Commands Dataset to create a multimodal dataset. This results in 10,760 samples over 188 classes as our dataset. Of these, 20% of the classes are used for testing, 20% for validation, and 60% for training. This yields a test set of 1028 samples.

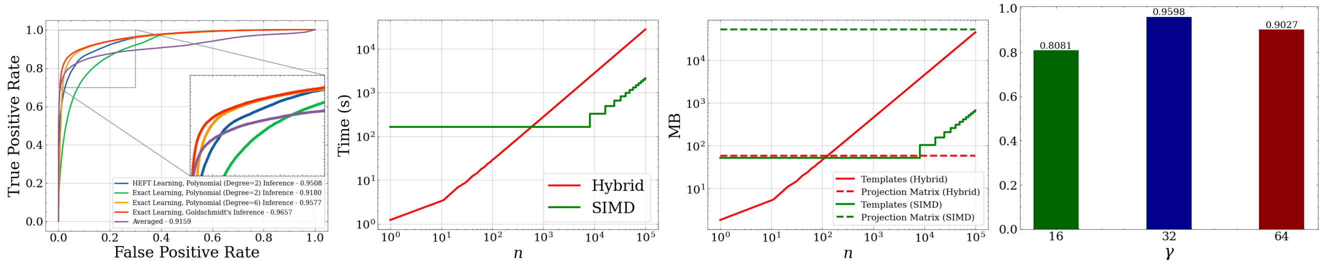
4.2. Comparison and Selection of Encoding Scheme

As discussed in Sec. 3.3.1 there are two encoding schemes, each with different computational properties. To select the one that is appropriate for our purposes, we first numerically compare them. The time and space complexity for the end-to-end pipeline, i.e., concatenation, projection, approximate normalization, and match score computation, of each encoding scheme are shown in Figs. 6b and 6c respectively. To compute the numerical values from the theoretical expressions in Table 1, we compute the runtime of each atomic operation in SEAL by averaging over 1,000 operations with the appropriate encryption parameters. Similarly, space is calculated by examining the size of a single ciphertext. As expected, we observe a cross-over point between the two, with SIMD being more efficient in terms of latency for $n > 1000$ and in terms of memory for $n > 10000$. Furthermore, for our dataset of size 1028, while the latency between the two is comparable, the dense encoding scheme has lower memory requirements. Therefore, we use the dense encoding scheme for all further experiments.

4.3. Evaluation Metrics and Results

In HEFT to compute the cosine similarity of feature vectors, we apply the appropriate normalization method (exact or approximate) on each vector and then take their dot product. Finally, we use the AUROC metric to evaluate the template fusion methods. The metric is computed in the unencrypted domain after decrypting the match scores.

Matching Performance: To evaluate the performance of HEFT we compare it against the following baselines, i) the unibiometric templates, ii) a simple concatenation of the unibiometric features, i.e., $\tilde{\boldsymbol{X}}$, iii) training using exact normalization, and iv) the feature averaging fusion technique



(a)

(b)

(c)

(d)

Figure 6: (a) ROC comparison of HEFT against baselines. (b) and (c) Comparison of theoretical runtimes and memory requirements for Hybrid and SIMD encoding schemes with $\delta = 1024$ and $\gamma = 32$. (d) Ablation study on γ, where 32 performs the best in our case.

Index	Data	Domain	Normalization		Dimensionality	AUROC
			Inference	Learning		
1	CPLFW	Unencrypted	Exact	-	512	0.8401
2	GSC	Unencrypted	Exact	-	512	0.8550
3	Average [16]	Encrypted	Exact	-	512	0.9159
4	Concatenation	Unencrypted	Exact	-	1024	0.9253
5	Learned	Unencrypted	Exact	Exact	32	0.9755
6	Learned	Encrypted	Poly (Deg=6)	Exact	32	0.9577
7	Learned	Encrypted	Poly (Deg=2)	Exact	32	0.9180
8	Learned	Encrypted	Goldschmidt's	Exact	32	0.9657
9	Learned	Unencrypted	Exact	HEFT (Deg=2)	32	0.9598
10	Learned	Encrypted	Poly (Deg=2)	HEFT (Deg=2)	32	0.9508

Table 2: AUROC comparison of HEFT versus baselines

Protocol	Enc. Norm. Method	Concatenation	Projection	Normalization	Preprocessing	Fusion Total	Score Comp.
Enrollment	Poly (Deg=2)	5.68	244.89	31.40	3.41	285.38	-
	Poly (Deg=6)	11.17	470.86	83.32	3.62	568.97	-
	Goldschmidt's	23.22	954.03	380.28	2.31	1,359.84	-
Authentication	Poly (Deg=2)	22.72	979.54	125.59	-	1,127.85	4.87
	Poly (Deg=6)	89.05	3,752.24	663.95	-	4,505.24	5.21
	Goldschmidt's	185.00	7,602.64	3,030.47	-	10,818.11	2.75

Table 3: Time (milliseconds) breakdown for each step in enrollment and authentication for a single sample. For comparison the same operations in message-space takes 0.62, 1.02, 11.75, and 4.51 μs respectively for concatenation, projection, normalization, and score computation per sample/match.

introduced in [16]. Table 2 compares the performance of HEFT with the baselines. We make the following observations, i) all fusion techniques with exact normalization (rows 1-5) namely averaging, concatenation and learned projection improve biometric matching performance with the latter providing the best performance, ii) approximate normalization at inference leads to drop in performance (rows 5 & 6, 5 & 7) ii) higher degree polynomial for approximate normalization performs better than the lower degree counterpart (row 6 & 7), iv) learning the projection matrix by taking the approximate normalization into account helps recover performance (rows 7 & 10), v) approximate normalization through Goldschmidt's algorithm is more accurate than that using polynomials (rows 6 & 8, 7& 8) and vi) computing the match score in the encrypted domain entails a slight loss in performance (rows 9 & 10). Overall, HEFT improves AUROC by 11.07% and 9.58% over CPLFW and Google Speech Commands, respectively.

Computational Complexity: The efficiency of homomorphic operations is critically dependent on the chosen encryption parameters. We select these parameters based on the multiplicative depth needed for end-to-end fusion and matching. Table 3 shows the latency of each individual component of HEFT. First, we observe a trade-off between performance and time complexity, with the 2^{nd}-degree polynomial being 2× faster than the 6^{th}-degree polynomial for enrollment. Although Goldschmidt's algorithm performs the best, it is 4.8× and 9.6× slower than HEFT with degree two approximation for enrollment and authentication respectively.

Ablation Study: We study the effect of γ, the dimensionality of the fused templates. Noting that γ should be a power of two to enable efficient match score computation, we compare three choices. Figure 6d shows that $\gamma = 32$ yields the best performance, with 64 being slightly better than 16.

5. Conclusions

In this paper, we proposed HEFT, the first non-interactive end-to-end homomorphically encrypted multimodal feature-level fusion and matching system. From an inference perspective, we carefully analyzed different data encoding and linear projection schemes and introduced approximate scale normalization through composite polynomial. From a learning perspective, we introduced FHE-Aware learning that explicitly accounts for the inherent limitations of FHE, namely the inability to perform exact normalization. Experimental results show that HEFT can overcome the performance losses due to approximations induced by FHE constraints and improve performance over the unibiometric features by 11.07% and 9.58% AUROC respectively while being practically feasible, taking 884 ms for fusing a pair of 512-dimensional vectors and matching against a gallery of 1024 templates.

Acknowledgments: This material is based upon work supported by the Center for Identification Technology Research and the National Science Foundation under Grant No. 1841517.

References

[1] Microsoft SEAL (release 4.0). https://github.com/Microsoft/SEAL, 2022. 7

[2] N. Alay and H. H. Al-Baity. Deep learning approach for multimodal biometric recognition system based on fusion of iris, face, and finger vein traits. *Sensors*, 20(19):5523, 2020. 3

[3] M. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, et al. Homomorphic encryption standard. In *Protecting Privacy through Homomorphic Encryption*. 2021. 3

[4] M. Babenko and E. Golimblevskaia. Euclidean division method for the homomorphic scheme ckks. In *IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*, 2021. 5

[5] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. D. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, A. Piva, et al. A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates. In *IEEE International Conference on Biometrics: Theory, Applications, and Systems (BTAS)*, 2010. 2

[6] M. Barni, G. Droandi, and R. Lazzeretti. Privacy protection in biometric-based recognition systems: A marriage between cryptography and signal processing. *IEEE Signal Processing Magazine*, 32(5):66–76, 2015. 2

[7] E. Bartuzi, K. Roszczewska, M. Trokielewicz, and R. Białobrzeski. Mobibits: Multimodal mobile biometric database. In *International Conference of the Biometrics Special Interest Group (BIOSIG)*, 2018. 3

[8] M. Blanton and P. Gasti. Secure and efficient protocols for iris and fingerprint identification. In *European Symposium on Research in Computer Security*, 2011. 2

[9] V. N. Boddeti. Secure face matching using fully homomorphic encryption. In *IEEE International Conference on Biometrics Theory, Applications, and Systems (BTAS)*, 2018. 1, 2

[10] Z. Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *Annual Cryptology Conference*, 2012. 1

[11] J. Bringer, H. Chabanne, M. Izabachene, D. Pointcheval, Q. Tang, and S. Zimmer. An application of the goldwasser-micali cryptosystem to biometric authentication. In *Australasian Conference on Information Security and Privacy*, 2007. 2

[12] G. S. Cetin, Y. Doroz, B. Sunar, and W. J. Martin. Arithmetic using word-wise homomorphic encryption. *Cryptology ePrint Archive*, 2015. 5

[13] J. H. Cheon, A. Kim, M. Kim, and Y. Song. Homomorphic encryption for arithmetic of approximate numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*, 2017. 1, 3

[14] J. H. Cheon, D. Kim, D. Kim, H. H. Lee, and K. Lee. Numerical method for comparison on homomorphically encrypted numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*, 2019. 5

[15] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5

[16] P. Drozdowski, F. Stockhardt, C. Rathgeb, D. Osorio-Roig, and C. Busch. Feature fusion methods for indexing and retrieval of biometric data: Application to face recognition with privacy protection. *IEEE Access*, 9:139361–139378, 2021. 3, 7, 8

[17] J. J. Engelsma, K. Cao, and A. K. Jain. Learning a fixed-length fingerprint representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(6):1981–1997, 2019. 5

[18] J. J. Engelsma, A. K. Jain, and V. N. Boddeti. HERS: Homomorphically encrypted representation search. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 4(3):349–360, 2022. 1, 2, 5

[19] J. Fan and F. Vercauteren. Somewhat practical fully homomorphic encryption. *Cryptology ePrint Archive*, 2012. 1

[20] C. Gentry, S. Halevi, and N. P. Smart. Fully homomorphic encryption with polylog overhead. In *International Conference on the Theory and Applications of Cryptographic Techniques*, 2012. 1, 2

[21] O. Goldreich. *Foundations of cryptography: volume 2, basic applications*. Cambridge University Press, 2009. 2

[22] M. Gomez-Barrero, E. Maiorana, J. Galbally, P. Campisi, and J. Fierrez. Multi-biometric template protection based on homomorphic encryption. *Pattern Recognition*, 67:149–163, 2017. 2

[23] A. Juels and M. Sudan. A fuzzy vault scheme. *Designs, Codes and Cryptography*, 38(2):237–257, 2006. 2

[24] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan. GAZELLE: A low latency framework for secure neural network inference. In *USENIX Security Symposium*, 2018. 5

[25] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[26] Y. J. Lee, K. R. Park, S. J. Lee, K. Bae, and J. Kim. A new method for generating an invariant iris private key based on the fuzzy vault system. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(5):1302–1313, 2008. 2

[27] M. Leghari, S. Memon, L. D. Dhomeja, A. H. Jalbani, and A. A. Chandio. Deep feature fusion of fingerprint and online signature for multimodal biometrics. *Computers*, 10(2):21, 2021. 3

[28] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu. Deep speaker: an end-to-end neural speaker embedding system. *arXiv preprint arXiv:1705.02304*, 2017. 7

[29] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 1

[30] G. Mai, K. Cao, P. C. Yuen, and A. K. Jain. On the reconstruction of face images from deep face templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(5):1188–1202, 2018. 1

[31] P. Markstein. Software division and square root using goldschmidt's algorithms. In *Conference on Real Numbers and Computers (RNC)*, 2004. 5

[32] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015. 7

[33] S. Panda. Principal component analysis using ckks homomorphic encryption scheme. *Cryptology ePrint Archive*, 2021. 5

[34] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. 2015. 7

[35] A. A. Ross and R. Govindarajan. Feature level fusion of hand and face biometrics. In *Biometric Technology for Human Identification II*. SPIE, 2005. 3

[36] P. P. Sarangi, D. R. Nayak, M. Panda, and B. Majhi. A feature-level fusion based improved multimodal biometric recognition system using ear and profile face. *Journal of Ambient Intelligence and Humanized Computing*, 13(4):1867–1898, 2022. 3

[37] S. Siena, V. N. Boddeti, and B. V. Kumar. Maximum-margin coupled mappings for cross-domain matching. In *IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, 2013. 3, 6

[38] S. Siena, V. N. Boddeti, and B. Vijaya Kumar. Coupled marginal fisher analysis for low-resolution face recognition. In *European Conference on Computer Vision Workshops (ECCVW)*, 2012. 3

[39] P. H. Silva, E. Luz, L. A. Zanlorensi, D. Menotti, and G. Moreira. Multimodal feature level fusion based on particle swarm optimization with deep transfer learning. In *IEEE Congress on Evolutionary Computation (CEC)*, 2018. 3

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 7

[41] C. Soutar, D. Roberge, A. Stoianov, R. Gilroy, and B. V. Kumar. Biometric encryption using image processing. In *Optical Security and Counterfeit Deterrence Techniques II*, volume 3314, pages 178–188, 1998. 2

[42] C. Soutar, D. Roberge, A. Stoianov, R. Gilroy, and B. V. Kumar. Biometric encryption. In *ICSA guide to Cryptography*, volume 22, page 649. McGraw-Hill New York, 1999. 2

[43] V. Talreja, M. C. Valenti, and N. M. Nasrabadi. Multibiometric secure system based on deep learning. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2017. 3

[44] L. C. O. Tiong, S. T. Kim, and Y. M. Ro. Multimodal facial biometrics recognition: Dual-stream convolutional neural networks with multi-feature fusion layers. *Image and Vision Computing*, 102:103977, 2020. 3

[45] J. R. Troncoso-Pastoriza, D. Gonzalez-Jimenez, and F. Perez-Gonzalez. Fully private noninteractive face verification. *IEEE Transactions on Information Forensics and Security*, 8(7):1101–1114, 2013. 2

[46] U. Uludag, S. Pankanti, and A. K. Jain. Fuzzy vault for fingerprints. In *International Conference on Audio-and Video-Based Biometric Person Authentication*, 2005. 2

[47] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. Jawahar. Efficient biometric verification in encrypted domain. In *International Conference on Biometrics (ICB)*, 2009. 2

[48] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. Jawahar. Blind authentication: a secure crypto-biometric verification protocol. *IEEE Transactions Information Forensics and Security*, 5(2):255–268, 2010. 2

[49] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[50] Y. Xin, L. Kong, Z. Liu, C. Wang, H. Zhu, M. Gao, C. Zhao, and X. Xu. Multimodal feature-level fusion for biometrics identification system on iomt platform. *IEEE Access*, 6:21418–21426, 2018. 3

[51] Q. Zhang, H. Li, Z. Sun, and T. Tan. Deep feature fusion for iris and periocular biometrics on mobile devices. *IEEE Transactions on Information Forensics and Security*, 13(11):2897–2912, 2018. 3

[52] T. Zheng and W. Deng. Cross-pose lfw: A database for studying cross-pose face recognition in unconstrained environments. Technical report, Beijing University of Posts and Telecommunications, 2018. 7